Increased autonomy for quadrocopter systems: trajectory generation, fail-safe strategies and state estimation

Diss. ETH No. 23137

INCREASED AUTONOMY FOR QUADROCOPTER SYSTEMS: TRAJECTORY GENERATION, FAIL-SAFE STRATEGIES, AND STATE ESTIMATION

A thesis submitted to attain the degree of
DOCTOR OF SCIENCES of ETH Zurich

(Dr. sc. ETH Zurich)

presented by

Mark Wilfried Müller
M.Sc. ETH in Mechanical Engineering

born on 22 August 1986
citizen of South Africa

accepted on the recommendation of

Prof. Dr. Raffaello D'Andrea, examiner
Prof. Dr. Manfred Morari, co-examiner

2016

Increased autonomy for quadrocopter systems: trajectory generation, fail-safe strategies, and state estimation

Mark Wilfried Müller

Institute for Dynamic Systems and Control
ETH Zurich
2016

Institute for Dynamic Systems and Control
ETH Zurich
Switzerland

Abstract

This thesis investigates three specific topics aimed at increasing the autonomy of quadrocopter systems. Such systems have become popular as they provide relatively easy and cheap access to the sky, allowing e.g. novel aerial perspectives, or the transportation of goods. Their potential value to industry is derived in a large part by their autonomy: their ability to safely and reliably execute tasks with minimal human intervention. The three topics investigated in this thesis are trajectory generation, fail-safe strategies and novel vehicles, and state estimation.

The trajectory generation algorithms investigated focus on achieving agile manoeuvres at low computational cost. Two approaches are presented, of which the first can generate and verify a candidate trajectory in approximately one microsecond on a normal laptop, achieved by only verifying feasibility after the generation of the motion. The second approach transforms the trajectory generation task into a convex optimisation problem, and then uses modern, highly-optimised methods to solve for the trajectory, at the cost of using conservative convex approximations of the feasible inputs.

The thesis investigates component failures that would break the global feedback control loop in an indoor aerial robotics testbed, and suggests strategies to mitigate the effects of such a failure. In addition, a detailed analysis into the attitude dynamics of multicopters is presented, from which follow generic conditions for a multicopter to maintain a "relaxed hover" (that is, remaining substantially at one place). This analysis is used to derive controllers with which a quadrocopter can maintain flight despite the complete loss of all but one of its propellers. Similarly, novel vehicles are designed that have fewer than four propellers.

The research into state estimation presents two generic methods for state estimation of a dynamic system, where the state includes an attitude. The two methods are extensions of the extended and unscented Kalman filters, and build on existing methods that encode the attitude to be estimated with a redundant representation (the system state vector includes a three element representation, in addition to a separate "reference" attitude). It is shown that, in contrast to existing methods, the methods are shown to keep track of the covariance correctly.

Based on the desire to have cheap, reliable, and flexible localisation for flying vehicles, work is also presented that fuses ultra-wideband ranging radios with the inertial sensors on a quadrocopter to estimate the quadrocopter's position, velocity, attitude, and angular velocity. This estimator is computationally light-weight enough to be run on a standard microcontroller, and the additional sensors are light and small enough to not significantly impact the performance of the quadrocopter to which they are attached.

Kurzfassung

Die vorliegende Arbeit untersucht drei verschiedene Aspekte, die dazu dienen die Autonomie von Quadrokoptersystemen zu verbessern. Solche Systeme haben an Popularität gewonnen, da sie einen relativ einfachen und günstigen Zugang zum Luftraum bieten. Ihr potentieller Wert für die Industrie liegt grösstenteils in ihren autonomen Eigenschaften, welche es ermöglichen Aufgaben sicher und zuverlässig mit einem Minimum an menschlicher Unterstützung auszuführen. In dieser Arbeit werden speziell die drei folgenden Aspekte untersucht: Generierung von Trajektorien, Sicherheitsstrategien und neue Vehikel, und Zustandsschätzung.

Die entwickelten Algorithmen für die Trajektoriengenerierung haben das Ziel agile Manöver mit möglichst geringem Rechenaufwand auszuführen. Dafür werden zwei Ansätze präsentiert, wovon der erste in ungefähr einer Mikrosekunde auf einem Laptop eine Trajektorie generieren und verifizieren kann. Dies wird dadurch ermöglicht, dass die Durchführbarkeit der Trajektorie erst nach der Generierung überprüft wird. Der zweite Ansatz transformiert die Trajektoriengenerierung in ein konvexes Optimierungsproblem und benutzt moderne, hoch-optimierte Methoden um die Trajektorien zu lösen, was den Nachteil hat, dass der Definitionsbereich der Systemeingänge nur durch konservative konvexe Annäherungen beschrieben werden kann.

Der zweite Teil dieser Arbeit untersucht das Versagen einzelner Komponenten, die die globale Rückkopplungsschleife eines Quadrokopter in einer geschlossen Testumgebung unterbrechen würden und schlägt Strategien vor um die Effekte eines solchen Versagens zu mindern. Weiterhin wird eine detaillierte Analyse von der Orientierungsdynamik von Multikoptern präsentiert, woraus allgemeine Bedingungen folgen unter denen ein Multikopter im weiteren Sinn schweben kann (d.h. ungefähr an einer Stelle bleiben). Diese Analyse wird dazu benutzt einen Regler zu entwickeln mit dem ein Quadrokopter trotz des Verlustes von bis zu drei Propellern fliegen kann. In einer ähnlichen Weise werden neue Flugvehikel entworfen, die weniger als vier Propeller besitzen.

Der letzte Teil dieser Arbeit behandelt die Zustandsschätzung und präsentiert zwei generische Methoden um den Zustand von einem dynamischen System zu schätzen, der als Zustand eine Orientierung beinhaltet. Diese Methoden sind Erweiterungen des sogenannten “extended” und “unscented Kalman filter” und bauen auf existierende Methoden, die es ermöglichen eine Orientierung zu encodieren (der Zustandsvektor enthält drei Orientierungselemente und zusätzlich eine “Referenzorientierung”). Es wird bewiesen, dass diese Methoden im Gegensatz zu existierenden Methoden die Kovarianz des Fehlers richtig abschätzen.

Generell ist eine günstige, zuverlässige und flexible Lokalisierung für Fluggeräte er-

strebenswert. In dieser Arbeit wird ein Quadrokopter präsentiert, der mit einem Ultrabreitbandradio ausgestattet ist, das (im Zusammenspiel mit Inertialsensoren) es ermöglicht die Position, Geschwindigkeit, Orientierung, und Winkelgeschwindigkeit von Quadrokoptern zu schätzen. Dieser Zustandsschätzer braucht wenig Rechenleistung und läuft auf einem Standardmikrokontroller. Das zusätzliche Radio ist so leicht und klein, dass es ohne grossen Einfluss auf die Dynamik eines Quadrokopter montiert werden kann.

[This page intentionally left blank for this notice.]

[Diese Seite wurde absichtlich für diesen Satz leer gelassen.]

Acknowledgement

There are many people to whom I am grateful after these PhD years. Firstly, I would like to thank Raff for his support, guidance, and enthusiasm. I have learnt much while at the IDSC, I can hardly imagine a better environment to have been in. Thank you for the trust you placed in me, and the opportunities you offered. I hope to pay it forward.

Secondly, I'd like to thank Prof. Morari for his willingness to act as co-examiner, for his kind feedback, and support.

Throughout my time at the IDSC, I have had the pleasure of being surrounded by smart and interesting people. I've gotten to be a co-author with the following folks: Markus (H), Robin, Mike, Weixuan, Federico, Sergei, Angela, and Markus (W). I'd like to thank the 'old hands', who helped me find my way at the beginning of my PhD (Sergei, Angela, & Sebastian) and who continue to give me inputs and advice (Markus & Angela).

In 2014 we went to Cape Town to do a public lecture, thanks Robin, Dario, and Fede for joining on this adventure. Working with you guys was always fun, it was a great team to be part of.

Markus W.: thanks for all the chocolate; for teaching me that often "it's just work", and I should stop complaining, sit down, and get it done; and for all the endless discussions on the meaning of life.

I had the great pleasure of sharing offices with Max for over four years – thank you for the (dubious?) German expressions you taught me, and for acting as translator on my abstract (you said "wow, I didn't know google translate was this bad" after you read my attempt at a Kurzfassung).

Finally, the encouragement, support, love, and friendship of Elena, my family, and my friends were invaluable.

Financial Support

This work was supported by the Swiss National Science Foundation (SNSF).

Contents

Foreword

This thesis documents the research carried out by the author during his doctoral studies under the supervision of Professor Raffaello D'Andrea at the Institute for Dynamic Systems and Control (IDSC) at ETH Zurich between September 2011 and October 2015.

The work is presented in the form of a cumulative thesis: its main content consists of six self-contained research articles (of which three are journal articles and three are conference contributions) that have been published or submitted for publication during the doctoral studies.

The work is divided into three parts: the generation of computationally light-weight trajectories for quadrocopters (Part A), quadrocopter failsafe strategies and novel vehicles (Part B), and state estimation (Part C).

The articles are put into context by three introductory chapters, which are structured as follows: Chapter 1 introduces and motivates this work, including the problems considered, related work, and the approaches used. Chapter 2 describes the key contributions of the research papers included in this thesis and how the individual papers relate to each other. Chapter 3 then provides a discussion of potential extensions and new directions of this research.

1

Introduction

> Basic research is what I am doing
> when I don't know what I am doing
>
> Wernher von Braun

Multicopters, and specifically quadrocopters, have evolved into versatile and popular flight platforms. They offer a good combination of agility and mechanical simplicity, due to having few moving parts (only the four motors for a quadrocopter). The outward mounting of the motors (and the resulting large lever arm for the motor forces) results in the ability to produce large torques and thus also large angular accelerations. Combining this with typically large thrust-to-weight ratios gives the quadrocopter its exceptional agility. Having only four moving parts means that mechanical maintenance and repair of a quadrocopter is much easier than e.g. that of a helicopter (with its intricate swashplate mechanism). These advantages have led to multicopters finding uses in a variety of fields, notably as platforms to deliver packages [1], for whale monitoring [2], for weed research [3], calibrating a radio telescope antenna [4], and construction of tensile structures [5].

Given the use of relatively low cost, off-the-shelf components, such aerial systems become increasingly economically attractive the more autonomous they become. Ideally, a flying vehicle can be used safely and reliably, with minimum human interaction, to achieve a given goal. This autonomy should ideally not reduce or limit the system's capabilities, or the vehicle's agility.

Multicopters and quadrocopters are also popular amongst researchers. Their dynamics are well approximated by elegant, nonlinear, differential equations, such that they motivate research into nonlinear control strategies. This has led to the creation of many controls-focused indoor testbeds, typically equipped with motion capture systems allowing for precise state estimation. Some examples of such systems are MIT Raven [6], Stanford/Berkeley STARMAC [7], the University of Pennsylvania GRASP multiple micro-UAV testbed [8], and the ETH Zurich Flying Machine Arena (FMA) [R1].

The measurement precision available in an indoor testbed such as the FMA, coupled with the controlled nature of the environment, allows for isolating individual research questions related to the performance of autonomous systems. In one extreme, using the motion capture system's measurement for the closed-loop state estimation means that the system's performance is determined primarily by the performance of the control

algorithms, and the physical capabilities of the experimental platform. On the other extreme, if an alternative sensing modality is used, the motion capture system provides accurate and reliable ground-truth data, and the controlled environment means that (non-repeatable) external disturbances are kept to a minimum.

The research in this thesis is divided into three parts. In Part A strategies are presented for generating quadrocopter trajectories, with a strong emphasis on algorithms with low computational cost. Part B discusses failsafe strategies and novel vehicles: the failsafe strategies comprise control strategies allowing a quadrocopter to fly despite actuator failure, and strategies allowing the system to deal with component failure (e.g. central motion capture system). This part also discusses some novel vehicle designs which are based on the same underlying theory as the actuator failsafe. State estimation is discussed in Part C, including theory related to applying Kalman filtering to systems whose state includes an attitude, and the development of an estimator specifically for a quadrocopter equipped with time-of-flight ultra-wideband ranging sensors. The context for each part is presented below, while the contributions made in the thesis (and specifically the contributions of the papers in this thesis) are discussed in Chapter 2.

1.1 Trajectory generation

A key requirement for the autonomous operation of a quadrocopter system is the ability to reliably plan motions. In the simplest case, such motions consist of quasi-statically moving a position setpoint, and using a near-hover controller to track this. Such a strategy is however feasible in only very few cases, exploits the dynamic capabilities of quadrocopters only to a very limited extent, and may be very energy inefficient. An ideal trajectory generation algorithm can compute a trajectory to achieve some high-level goal in a short amount of time. This trajectory must satisfy motion constraints (such as avoiding obstacles) and input constraints, and ideally the trajectory can be objectively justified, e.g. by optimising for a useful quantity (such as expended energy).

Much literature exists on trajectory generation techniques for quadrocopters. Trajectory generation schemes may be roughly grouped into two categories: the first category comprises those that decouple geometric and temporal planning, where in a first step a (geometric) path is generated; in a second step a temporal profile is associated to the path in order to guarantee feasibility with respect to the quadrocopter dynamics. Examples in this category include using lines [9], polynomials [10], or splines [11].

The second group of algorithms exploits the differential flatness of the quadrocopter dynamics in order to derive constraints on the trajectory, and then solve an optimization problem. Examples of optimisation criteria used are minimum snap [12], minimum time [13] and [14], shortest path under uncertain conditions [15], or combinations of position derivatives [16]. In [17] a search over parameters is proposed for quadrocopter motion planning, including trajectories where the position is described by polynomials in time.

This thesis investigates interception trajectories for quadrocopters, that is trajectories with a fixed execution time, and with a (potentially only partially) defined end state. Such trajectories are required whenever the system's high-level goal is time-varying, for example landing a quadrocopter on a moving platform, or catching a falling object.

A strong emphasis is placed on computational complexity. The time-varying examples mentioned above are often subject to significant uncertainty, for example due to imperfect measurements and modelling – the ability to recompute trajectories in real time, in response to new information, therefore allows the system to interact much more responsively with its environment. Two complementary approaches that attempt to solve for such trajectories are presented. Both approaches rely on the same underlying model of the quadrocopter, where the dynamics are simplified by assuming that the vehicle angular velocity can instantaneously track a given command. This simplification is physically motivated by a quadrocopter's ability to produce large torques (due to the outward mounting of the propellers) combined with low angular inertia, such that the quadrocopter can produce large angular accelerations. The model used for the path planning then treats the angular velocity of the vehicle as an input, and an inner controller tracks these desired angular velocities by commanding motor forces. This decoupling is described in more detail in [R1]. Exploiting the differential flatness of the quadrocopter dynamics, it can then be shown that the position needs to be differentiated only three times (that is, to the jerk) until the inputs (angular velocity and total thrust) appear. The two approaches also share the same optimisation goal: to minimize the jerk squared along the trajectory. This is shown to be related to the inputs of the quadrocopter, such that a trajectory that is optimal in this sense is also likely to have "nice" inputs; it is also amenable to analysis and allows for a variety of simplifications.

The two presented approaches differ primarily in how they deal with constraints. The first approach computes trajectories without regard to constraints in a first step, and then in a second step tests the trajectory for feasibility. The second approach uses conservative convex approximations of the constraints, and then poses the trajectory generation problem as a convex optimisation problem. Recent advances in creating specialised solvers for convex optimisation problems (see e.g. [18]–[20]) mean that trajectories with time horizons spanning up to 200 time steps can be solved for in real time on a laptop computer, where "real time" is here taken to be on the order of 10 ms.

The first approach is computationally much more light-weight, and trajectories can be computed and evaluated approximately ten thousand times faster than with the second approach. This means that the first approach is especially useful if many potential trajectories exist which solve a given problem, such that one may simply evaluate a large number of them in the hope that at least some will test to be feasible. The first approach is also able to test feasibility over the entire thrust feasible space, in contrast to the second approach that requires a convex approximation thereof. However, because the second approach uses the feasibility criteria in the generation stage, it may be able to compute trajectories where the first fails.

Both approaches may be executed quickly enough to be used as implicit feedback laws,

similar to what is done in model predictive control, where an entire trajectory is planned at every controller update step, and the initial inputs of the trajectory are applied to the system.

1.2 Failsafe strategies and novel vehicles

A typical strategy to improve the safety of aerial systems is redundancy, where multiple independent subsystems exist which can execute the same tasks. Examples of such redundant systems in multicopters are the use of a backup safety pilot during autonomous vehicle experiments (where the safety pilot can remotely take over control of the vehicle if an anomaly is detected, see e.g. [21]), and the use of hexa- and octocopters (with respectively six or eight propellers) that can fly with a minimal loss of capabilities after the loss of one of their actuators (see e.g. [22], [23]).

Such redundancy often comes at significant cost: to be effective, safety pilots must be highly skilled, and one pilot can be expected to monitor and control at most one aerial vehicle, making their use for long-duration, multi-vehicle experiments expensive. The use of mechanical redundancy, as in the case of the hexa- and octocopters, may also be problematic: adding the redundant components (which are by definition not needed in normal operation) increases the mass of the vehicle (and thus also its power requirements in operation). Although this mechanical redundancy reduces the probability of catastrophic failure due to e.g. an actuator failure, it may increase the severity of a different type of failure (e.g. a software glitch that causes the vehicle to fly into the ground).

This thesis investigates actuator failsafe strategies, which allow a quadrocopter to maintain controlled flight despite the complete loss of one or more actuators. After such a failure occurs, the quadrocopter can no longer maintain hover in the conventional sense, but can instead enter a "relaxed hover" condition, where specifically the vehicle's angular velocity is allowed to be non-zero. Effectively, this means that the vehicle will rapidly rotate about an axis, but can still fully control its position in space.

These relaxed hover solutions are also applied to the creation of novel, increasingly under-actuated flying vehicles. As an example, it is shown that a controllable flying vehicle can be constructed with only a single moving part (the rotation of an attached propeller), and having only a single input (the propeller thrust) which is used to stabilise the vehicle. Notably, the vehicle requires no passively stabilising aerodynamic surfaces.

The thesis also investigates system failsafe strategies for indoor testbeds such as the Flying Machine Arena. For such systems, the failure of the central motion capture system or the failure of the command radio system effectively breaks the global feedback control loop. These systems are expected to be lightweight: they must be easily adapted to allow for novel research. Thus strategies are investigated where the vehicles use their existing onboard sensors to maintain a partial state estimate, allowing the vehicles to mitigate the effect of a failure in the global feedback control loop.

1.3 State estimation

State estimation is important in autonomous systems for at least two reasons [24]: the system state may be desirable for its own sake, for example when monitoring a system's performance or compliance. Alternatively, the state estimate may be used in closed loop by a controller to compute desired control inputs. When used in a feedback control setting, the system's achievable performance is often dictated by the quality of the state estimate.

The state estimation task for many systems, especially in robotics, is often complicated by the presence of non-linearities. In such cases it is common to use non-linear extensions of the Kalman filter, for example the extended Kalman filter (EKF) or the unscented Kalman filter (UKF) [24].

A typical source of such non-linearity is attitude dependence in the system: for example, a quadrocopter's translational motion is primarily determined by the total thrust force it produces, and gravity. The direction of the total thrust force is determined by the quadrocopter's attitude, which is itself a dynamic state. The estimation of attitudes is however complicated by the inherent difficulties in their parametrisation, as an attitude has three degrees of freedom, but no three dimensional parametrisation exists that is both global and without singular points [25].

One approach is to have a three parameter "attitude error" parametrisation in the Kalman filter state, coupled with an additional, redundant, "reference" attitude parametrisation. The attitude error in the filter state is to be kept small, and thus far away from singularities in the representation, while the reference attitude may be large. This attempts to have the best of both worlds: the minimal error parametrisation in the filter reduces computational cost when compared to higher order representations, and avoids the difficulties associated with applying nonlinear constraints to the filter state. Nonetheless, the use of a reference attitude to keep the attitude error small means that singularities may be avoided. Such approaches date back to at least 1970, and may be found in e.g. [26], [27].

These approaches require a periodic "reset", where information is moved from the filter's attitude error state to the reference attitude, and the attitude error is reset to zero. The difficulty here is that it is not clear how the attitude covariance must be adapted during the reset. In Part C of this thesis it is shown that the approach common in the literature (e.g. [27]–[31]) does not keep track of the covariance correctly to first order during the reset step. Two covariance corrections are suggested, one which is shown to keep track of the covariance correctly to first order, and another (based on the unscented transform [32]) that is more accurate, but comes at a higher computational cost.

This research was motivated by efforts to create a localisation system for quadrocopters that does not rely on the centralised motion capture system. The goal was to use low-cost and low-mass components, which do not yield as much information as the motion capture system but may be more flexible. Since such measurements may be expected to be less informative, greater care must be taken to correctly track the estimate covariance.

A popular alternative sensing modality uses cameras on the vehicle itself, as in e.g. [33],

[34]. Such on-board vision systems have the advantage of allowing the autonomous system to be fully contained in the quadrocopter, but they may be more sensitive to the environment (e.g. lighting, sufficient environmental texture).

Instead, in this thesis, time-of-flight ultra-wideband radios are used, that allow the quadrocopter to measure its distance to a set of anchors fixed in the world. These radios are very low-cost, have low mass, and therefore offer a cost-effective and flexible alternative to centralised motion capture systems, as e.g. currently used in the Flying Machine Arena. Additionally, the radio technology should be robust to external disturbances, such as external radio interference. This makes such a system attractive in safety critical situations.

The following chapter presents the research done in this thesis, with a specific focus on the six papers that make up this thesis.

2

Contributions

This chapter summarizes the scientific contributions for each of the papers that constitute this thesis. In total, three journal publications, and three peer-reviewed conference proceedings, are discussed. Furthermore, a list of other contributions such as results from unpublished student projects and outreach activities are provided in this chapter.

2.1 Trajectory generation

[P1] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation", *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015

Quadrocopters offer exceptional agility and can perform complex and highly dynamic tasks. To exploit this agility in situations where the exact task or environment is not known in advance requires the ability to plan agile motions in real time. In [P1] an algorithm is presented allowing for the rapid generation and feasibility verification of motion primitives for quadrocopters. These primitives are defined by the quadrocopter's initial state, and any combination of the quadrocopter's desired state at a specified goal time. By exploiting the problem structure, including the differential flatness of quadrocopter dynamics, a computationally efficient algorithm is defined, allowing the generation and evaluation of approximately one million motion primitives per second on a standard laptop computer. The speed at which the primitives may be computed means that the method may be applied especially in situations where there is considerable freedom in the desired end state – in such a situation, the user may sample over a large number of primitives which solve a given high-level goal, and then retain the 'best' that achieves the goal and is input feasible.

Contribution The paper presents a quadrocopter motion primitive, based on a first-principles model of the quadrocopter dynamics. Using the Pontryagin minimum principle, an analytic solution is found for the translational motion that minimizes the integral of the jerk squared along the quadrocopter's trajectory (where the jerk is the third temporal derivative of position). These trajectories are solved under the simplifying assumption

Figure 2.1 To validate the method of [P1], a quadrocopter with an attached net is tasked to catch a ball thrown by a person.

that the quadrocopter's angular dynamics are sufficiently fast that they may be neglected compared to its translational dynamics. The optimal control problem is decoupled along the three spatial axes, and solved in closed form for each axis as a function of the initial state, and desired final state. It is shown that the decoupled solutions are also optimal for the original, coupled problem. These closed form solutions are then used to derive computationally efficient feasibility tests, allowing to test a given primitive for input feasibility, and the violation of position box constraints. For the special case of rest-to-rest trajectories, these solutions are considerably simplified. The approach is experimentally validated by tasking a quadrocopter with an attached net to catch a thrown ball, as shown in Fig. 2.1.

[P2] M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception", in *European Control Conference (ECC)*, 2013, pp. 1383–1389

The computational power offered by modern computers, coupled with recent advances in the numerical solution of convex optimisation problems, means that it has become feasible to solve model predictive control problems in real time for highly dynamic systems. In [P2] a highly optimized interior-point solver (based on FORCES [19]) is used to create interception trajectories based on a discrete-time model of the quadrocopter dynamics. Similarly to [P1], the trajectories are defined by the quadrocopter's initial state, a desired final state, and a desired trajectory duration.

The method is significantly slower at generating trajectories than that presented in [P1], but is able to actively take constraints into account at the generation stage, unlike [P1] where the primitive is first generated and subsequently tested for feasibility. However, because the constraints must be decoupled per spatial axis and approximated by convex constraints, the constraints that can be used for planning tend to be very conservative. This means that, in practise, the method of [P1] appears to be preferable to that of [P2].

Figure 2.2 Novel flying vehicles developed in this thesis,, from left to right: the mono-, bi-, tri-, and quadspinner. The monospinner is described in [R2], the remainder in [P3].

Contribution The paper presents a first-principles dynamic model of a quadrocopter, and constructs a per-axis optimal control problem for trajectory generation. This cost function is similar to that used in [P1], with the major difference as to how the constraints are encoded. This paper encodes the quadrocopter input constraints (taken to be angular velocity and total thrust limits) using conservative per-axis box constraints. This means that each axis may be solved for independently, and the resulting trajectory is guaranteed to be input feasible (however, the solution may no longer be optimal for the original, coupled problem). The problem is discretised in time, using timesteps equivalent to 50Hz, and then numerically solved. The algorithm is then applied in experiment in the Flying Machine Arena.

2.2 Failsafe strategies and novel vehicles

[P3] M. W. Mueller and R. D'Andrea, "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles", *International Journal of Robotics Research*, 2015

In paper [P3] an approach is developed to solve for "relaxed hover" equilibria of a multicopter. A conventional hover is for the multicopter to have a constant position (and thus zero translational velocity), with constant attitude (and thus also zero angular velocity). For a relaxed hover solution, the only requirement is that the vehicle's position is approximately constant, and that the solution may be described with constant parameters in a body-fixed frame. Thus, it is allowed for the vehicle to rotate during flight. Such solutions can be used to describe flight conditions for vehicles with as few as one propeller, allowing for the design of novel vehicles (see Fig. 2.2), or to create fail-safe solutions for existing multicopters (see Fig. 2.3 and 2.4).

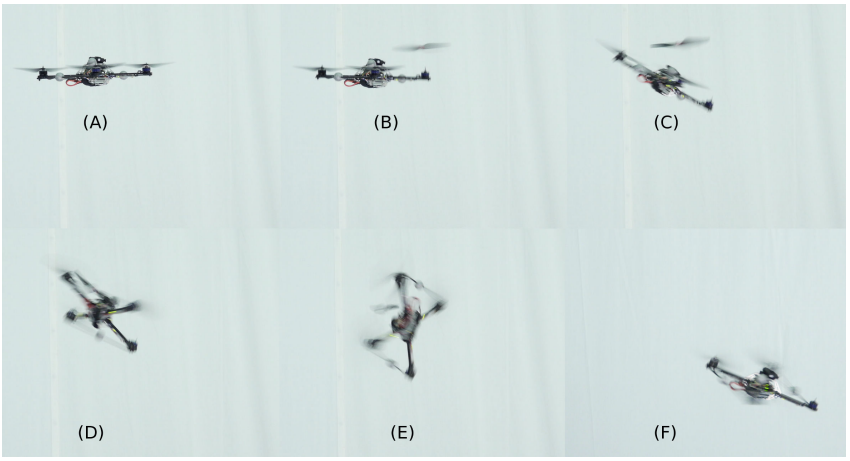

Figure 2.3 A sequence of photos showing a quadrocopter experiencing the loss of a propeller, and automatically recovering. (A) shows the quadrocopter in normal operation. In (B) the propeller detaches due to vibrations, and the quadrocopter starts pitching over in (C)–(E). In (F) the vehicle has regained control, and is flying stably. The stills are taken from the video at `www.mwm.im/l/VidFailsafe1`.

Contribution The paper presents a detailed derivation of the effects acting on a multicopter at low translational speeds. This model is then used to derive general conditions on hover solutions of multicopters where all propellers have a common thrust direction, without assumptions on the number of propellers, their locations or the vehicle's mass distribution. A general framework for establishing whether the attitude of the multicopter is controllable about a relaxed hover solution is then derived, which may then also be used to compute a linear, time-invariant controller. These results are then applied to create failsafe strategies for quadrocopters experiencing the loss of one or more propellers, and to design novel vehicles with fewer than four propellers. It is furthermore shown that the results can be used to control a quadrocopter with very large centre of mass offsets.

The theory developed has subsequently been applied to create a single moving part, single input, controllable flying vehicle, shown in flight in Fig. 2.5 and described in [R2]. This is potentially the mechanically simplest, controllable, flying vehicle that exists, since it conceptually consists of only a single propeller attached to a body.

Two patent applications follow from this work: [Pat1], [Pat2].

[P4] M. W. Mueller and R. D'Andrea, "Critical subsystem failure mitigation in an indoor uav testbed", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 780–785

Many visitors attend demonstrations of the Flying Machine Arena in Zurich, and the system has also been exhibited abroad, across four continents. This demanding demonstration schedule, coupled with the complexity of the system, motivates the development of failure mitigation strategies. In Paper [P4] a strategy is developed to allow for the mitigation of the effects of the failure of the Arena's central measurement system, or the failure of the radio system used to send commands to the vehicles. The centralised

Figure 2.4 A quadrocopter being flown by hand on the Züriberg after the simulated complete failure of one propeller. This is a still taken from the video at `www.mwm.im/l/VidFailsafe2`.

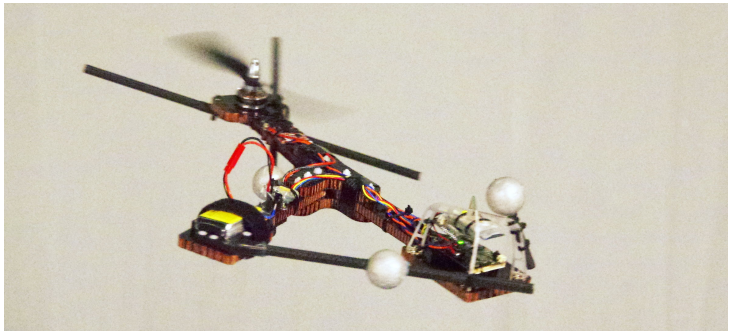

Figure 2.5 The monospinner in flight [R2]. The monospinner is a controllable flying vehicle with a single moving part (the rotating propeller), and a single input.

nature of the Flying Machine Arena, with a centralised measurement system capturing the state of the vehicles', and a single command radio broadcasting commands to all vehicles (see [R1] for details) means that such failures effectively cut the high-level feedback control loop, as the vehicles no longer have access to measurements of their position and orientation.

Contribution The paper analyses the most relevant potential causes of failure in the Flying Machine Arena, and presents algorithms with low computational cost that allow the vehicles greater autonomy. The fundamental strategy is to periodically transmit to the vehicle its current velocity and orientation, and then use the on-board rate gyroscope to predict this state forward in time. If the motion capture system then fails (or the command communication channel is cut), the vehicle can then attempt to control this state to zero. If the fault is not repaired within a predefined period (after which the open-loop state estimate may no longer be accurate), the vehicle attempts a soft crash-landing. Experimental results are given, showing the successful reduction of the vehicle's kinetic energy after a failure, thus improving the safety of the system.

Figure 2.6 Some FMA events at which the failsafe system of [P4] was deployed, from left to right: the Flight Assembled Architecture (where quadrocopters built a 6 m structure [R3]), Prof. Raffaello D'Andrea giving a talk at TED (see `www.ted.com/talks/raffaello_d_andrea_the_astounding_athletic_power_of_quadcopters`), and the filming of SPARKED (a short film, see `youtu.be/6C8OJsHfmpI`). Photos by Markus Waibel, James Duncan Davison, and the Sparked Team, respectively.

Remark The system currently implemented in the Flying Machine Arena is closely based on [P4]. However, due to improvements in available onboard computational power, a more detailed dynamic model and a more sophisticated estimator are used, which include aerodynamic effects and accelerometer measurements. The aerodynamic effects are described in more detail in [P6].

2.3 State estimation

[P5] M. W. Mueller, M. Hehn, and R. D'Andrea, "Kalman filtering with an attitude", Zurich, Tech. Rep., 2016

A key requirement for autonomous operation is an accurate state estimate. In robotics applications, it is common that the state to be estimated includes an attitude, or orientation: system dynamics that include attitudes are typically nonlinear, and thus require the use of nonlinear estimation techniques, such as the extended Kalman filter or the unscented Kalman filter. The task is further complicated by the difficulties inherent in representing attitudes: although an attitude has three degrees of freedom, no three element parametrisation exists that is both global and without singularities. This chapter presents two algorithms to estimate the state of a generic dynamic system, where the state includes an attitude. The algorithms are based on the extended and unscented Kalman filters.

Contribution The algorithms presented follow an approach similar to the widely used "Multiplicative Extended Kalman Filter" [27] which uses a redundant attitude representation consisting of a three element attitude in the state vector, and a reference attitude. The three element attitude is kept small (and thus far away from its singularities) by

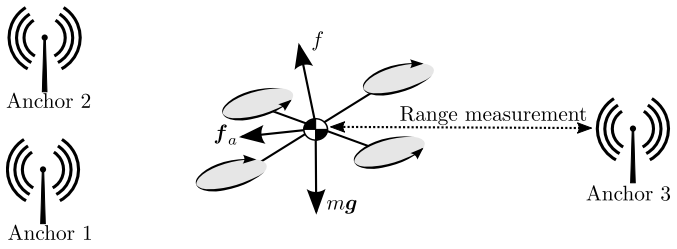

Figure 2.7 A quadrocopter uses measurements from a ranging radio to measure the distance to a set of stationary radio anchors [P6]. An accelerometer effectively measures the static thrust f and other aerodynamic effects $\boldsymbol{f}_a$, thereby providing information on the quadrocopter's translational velocity. This allows the quadrocopter to estimate its state, using low-cost and flexible infrastructure.

periodic reset operations, where the reference attitude is updated. In the chapter it is shown that the reset method used in the literature fails to correctly track the estimated covariance. Corrections are derived based on a first-order analysis, and based on the unscented transform, where the latter is shown to be significantly more accurate at the cost of increased computational complexity. The algorithms are furthermore extended, such that they may be easily applied to arbitrary systems (for which the usual assumptions needed for extended/unscented Kalman filtering apply).

[P6] M. W. Mueller, M. Hamer, and R. D'Andrea, "Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015

The centralised motion capture system in the Flying Machine Arena allows the system to determine the position of the vehicles in the space. Such a localisation is an important requirement for autonomous operations, but motion capture systems are expensive, centralised (and thus represent a single point of failure), and are typically constrained to small workspaces. There is thus considerable interest in alternative localisation systems. Ultra-wideband radios are readily available at low cost, add little mass to a vehicle, and therefore are promising candidates for creating low-cost localisation systems.

Contribution In this paper an estimation strategy is described that fuses accelerometer and rate gyroscope measurements with distance measurements from a set of ultra-wideband radio anchors, while exploiting a model of the quadrocopter's aerodynamics. These anchors have fixed, known positions in the world, and the range measurements are fused using an extended Kalman filter, using a typical microcontroller. The system layout is shown in Fig. 2.7.

Remark This work was done before [P5], and thus does not include the covariance correction. Difficulties with tuning the Kalman filter (selecting noise variances) led to a

closer investigation of the assumptions made in the Kalman filter, and thus led to the work described in [P5].

Remark The paper includes substantial contributions from the second author, Michael Hamer: an algorithm to compute the range between two radios using simple messages, as well as discussions about systematic delays in the radio system, and how to estimate and mitigate them.

2.4 List of publications

Publications in this Thesis

[P1] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation", *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[P2] M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception", in *European Control Conference (ECC)*, 2013, pp. 1383–1389.

[P3] M. W. Mueller and R. D'Andrea, "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles", *International Journal of Robotics Research*, 2015.

[P4] M. W. Mueller and R. D'Andrea, "Critical subsystem failure mitigation in an indoor uav testbed", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 780–785.

[P5] M. W. Mueller, M. Hehn, and R. D'Andrea, "Kalman filtering with an attitude", Zurich, Tech. Rep., 2016.

[P6] M. W. Mueller, M. Hamer, and R. D'Andrea, "Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

Related publications

[R1] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena", *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[R2] W. Zhang, M. W. Mueller, and R. D'Andrea, "A controllable flying vehicle with a single moving part", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[R3] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines", *Control Systems, IEEE*, vol. 34, no. 4, pp. 46–64, 2014.

[R4] M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 45–52.

[R5] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3480–3486.

[R6] R. Ritz, M. W. Mueller, M. Hehn, and R. D'Andrea, "Cooperative quadrocopter ball throwing and catching", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 4972–4978.

[R7] M. W. Mueller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 5113–5120.

Patent applications

[Pat1] M. W. Mueller, S. Lupashin, R. D'Andrea, and M. Waibel, "Controlled flight of a multicopter experiencing a failure affecting an effector", *Patent pending, WO 2014/198641 A1*, 2014.

[Pat2] M. W. Mueller, S. Lupashin, R. D'Andrea, and M. Waibel, "Volitant vehicle rotating about an axis and method for controlling the same", *Patent pending, WO 2014/198642 A1*, 2014.

2.5 Student supervision

Masters thesis

The masters thesis is a six-month, full-time project.

[MT1] R. Ritz, "Cooperative quadrocopter ball play", Masters thesis, ETH Zurich, 2011.

[MT2] S. Berger, "State estimation using ultra-wide band ranging radios and inertial sensors", Masters thesis, ETH Zurich, 2013.

[MT3] T. Kaegi, "Design, modelling, and control of a novel flying vehicle", Masters thesis, ETH Zurich, 2013.

[MT4] W. Zhang, "Design, modelling, and control of a single propeller vehicle", Masters thesis, ETH Zurich, 2015.

Semester project

The semester project is a semester-long, part-time project.

[SP1] P. Petit, "Integrating fixed-wing UAVs into the FMA", Semester project, ETH Zurich, 2012.

[SP2] S. Hubacher, "Improving the quadrocopter blind hover", Semester project, ETH Zurich, 2012.

[SP3] S. Berger, "Fault detection and user interface for the FMA", Semester project, ETH Zurich, 2012.

[SP4] A. Zanelli, "Catching rings on a quadrocopter", Semester project, ETH Zurich, 2013.

[SP5] W. Zhang, "Randomised trajectory generation", Semester project, ETH Zurich, 2014.

Bachelors thesis

The bachelors thesis is a three-month, full-time project.

[BT1] P. Puentener, "Improved filter for ball tracking/prediction", Bachelors thesis, ETH Zurich, 2011.

[BT2] M. Wermelinger, "Quadrocopter ball juggling optimization", Bachelors thesis, ETH Zurich, 2013.

[BT3] E. Kaufmann, "Using magnetometer during indoor flight", Bachelors thesis, ETH Zurich, 2014.

[BT4] L. Huber, "The tricoptercopter and state estimation", Bachelors thesis, ETH Zurich, 2014.

Studies on mechatronics

The Studies on Mechatronics is a literature review, comprising approximately eighteen full work days. Note that the bachelors theses [BT1]–[BT3] also contained Studies on Mechatronics: these are not listed here.

[SoM1] F. Metzler, "A survey on distance sensors", Studies on mechatronics, ETH Zurich, 2013.

Internship

[In1] A. Wilkinson, "Design of novel flying vehicles", Internship, ETH Zurich, 2015.

Figure 2.8 "Quadrocopter dynamics: a demonstration", presented at the 19th World Congress of the International Federation for Automatic Control, in Cape Town 2014, in front of an audience of 500 people. Photo by Mark Rosenberg.

2.6 Outreach

Talks

Note that the talks at scientific conferences corresponding to the publications [P2], [P4], [P6], [R4], [R5], [R7] are not listed. The talk at the IFAC World Congress, in August 2014, is shown in Fig. 2.8.

Mar. 2015	*Systems Control Seminar* (University of Toronto).
Feb. 2015	*Lunch talk* (California Institute of Technology).
Jan. 2015	*Drones: From Technology to Policy, Security to Ethics* (Zurich).
Jan. 2015	*Future UAS Technologies Workshop* (Thun).
Nov. 2014	*Swiss Drone Day* (Bern).
Oct. 2014	*TEDxAirbus* (Toulouse).
Aug. 2014	*Public lecture, 19th World Congress of the International Federation of Automatic Control* (Cape Town), `www.mwm.im/l/IFACTalk`.
June 2014	*Frontiers of Robotics and Autonomous Systems* (Hong Kong University of Science and Technology).
Nov. 2013	*Swiss-Kyoto Symposium* (Zurich).
Oct. 2013	*Computer Science Seminar, University of Witwatersrand* (University of Johannesburg).
May 2013	*Cooperation Forum: Advanced Driver Assistance Systems, Bayern Innovativ* (Aschaffenburg).
May 2012	*General assembly, Swiss Association of Aeronautical Sciences* (Zurich).

Exhibitions

During the period of this thesis, the Flying Machine Arena was shown at the following exhibitions. In all cases, to varying degrees, work contained in this thesis was used.

August 2014	Cape Town	Public lecture, IFAC 2014
June 2013	Edinburgh	TED Global
December 2012	Zurich	Zurich.Minds
June 2012	San Francisco	Google I/O after hours
April 2012	Hannover	Hannover Messe
December 2011	Orléans	Flight Assembled Architecture

In addition to the above, smaller demonstrations were also conducted during some invited talks, where a vehicle was flown by a joystick whilst demonstrating the ability to maintain controlled flight after the loss of one or two actuators.

Lab demonstrations

The Flying Machine Arena regularly hosts visitors, ranging from groups of primary school students to distinguished professors. During such visits, different demonstrations are presented. Demonstrations stemming from, or including, work done in this thesis include:

- a quadrocopter with a rigidly attached badminton racket returning a ball thrown by a person,
- a quadrocopter maintaining controlled flight after the complete removal of first one, then two, of its propellers,
- the trispinner in flight, following a setpoint command given by a user with a pointer (this demonstration, and the vehicle, were developed by Alex Wilkinson [In1]).

Youtube videos

The following videos were created for consumption by the general public, demonstrating some of the research results.

[V1] R. Ritz, M. W. Mueller, M. Hehn, and R. D'Andrea, *Cooperative quadrocopter ball throwing and catching*, Sep. 2012. [Online]. Available: `https://youtu.be/hyGJBV1xnJI`.

[V2] M. W. Mueller, M. Hehn, and R. D'Andrea, *Rapid trajectory generation for quadrocopters*, Oct. 2013. [Online]. Available: `https://youtu.be/R8nX-cg-WIO`.

[V3] M. W. Mueller, M. Waibel, S. Lupashin, and R. D'Andrea, *Quadrocopter failsafe algorithm: Recovery after propeller loss*, Dec. 2013. [Online]. Available: `https://youtu.be/bsHryqnvyYA`.

[V4] M. W. Mueller, S. Berger, and R. D'Andrea, *Onboard quadrocopter failsafe: Flight after actuator failure*, Mar. 2014. [Online]. Available: `https://youtu.be/ek0FrCaogcs`.

[V5] M. W. Mueller and R. D'Andrea, *The bicoptercopter: A preview*, Oct. 2014. [Online]. Available: `https://youtu.bbe/csfAOCdkCdU`.

Software resources

The following software resources were developed during the thesis, and are released under the GPL license [35].

- `www.github.com/markwmuller/RapidQuadrocopterTrajectories` An implementation of the motion primitive algorithm described in [P1].
- `www.github.com/markwmuller/controlpy` A Python library implementing some controls analysis and synthesis tools, e.g. computing controllability Gramian, system H_2 or H_∞ norms, and synthesizing LQR controllers.

Selected media coverage

- "Quadrocopters — Tech Report", television interview, *ENCA South Africa*, August 2014.
- "Quadrocopters; redefining the future of unmanned aerial vehicles", television interview, *TechBusters, CNBC Africa*, August 2014.
- "Quadcopters fascinate crowd in city", Jan Cronje, *Weekend Argus*, August 2014.
- "When the drones go dancing", Arthur Goldstuck, *Mail and Guardian*, August 2014.
- "Die tollkühnen Schweizer mit ihren fliegenden Drohnen", Eva Raisig, *Deutschland Radio*, March 2014.
- "Every quadrotor needs this amazing failsafe software", Evan Ackerman, *IEEE Spectrum*, March 2014.
- "Amazing quadcopter prop-loss recovery", Chris Anderson, *3D robotics*, March 2014.
- Live television interview on *Dagbreek*, October 2013.
- "Perfekte Teamarbeit in der Luft", Frankfurter Allgemeine Zeitung, October 2012.

In addition to the above, the research carried out in the FMA has been widely publicized. Demonstrations of the system have been shown on television by SRF, Tele Züri, RTS, RSI, BBC, Discovery Channel, Fuji TV, NBC, RTL Aktuell, CNN, Canal+, Nippon Television Network, France 2, Daily Planet, Deutsche Welle, CNBC Africa, TV Chosun,

Nine Network Australia, and CBC. Articles about our research have appeared in print (Sonntagszeitung, c't Magazine, Technology Review, engine, LOOP Magazine, CHIP, Science et Vie, ICON Magazine, Globe, ETH life, Giornale del Popolo, Corriere del Ticino, 20 Minuti, Blick am Abend, Zürichsee-Zeitung, New Scientist, Tracé, Die Burger, Weekend Argus, and Wired Magazine).

3

Future work

This chapter provides an overview of potential future work based on the research presented in this thesis.

Randomised motion planning

The low computational cost of the motion primitive of [P1] makes it attractive for use in randomised path planners, such as in a rapidly exploring random tree (RRT) or a probabilistic roadmap (PRMs) [36]. Such methods, however, rely on the ability to rapidly evaluate whether a motion is collision-free in a non-convex environment. The method of [P1] allows to very quickly evaluate collisions with planes in space, such that it would need to be extended for non-convex flight spaces.

Initial work [SP5] used the rest-to-rest trajectories of [P1] as primitives for an RRT* search through a cluttered environment. Those trajectories appear to offer a good compromise between fast computation (the resulting trajectories are composed of straight lines, and thus collision checks are greatly simplified) and fast execution. However, the vehicle comes to rest during each trajectory segment. This could perhaps be improved by adding an additional step to the algorithm, taking the result of [SP5] and attempting to smooth the points where the quadrocopter comes to rest.

Libraries of evasive/alternative manoeuvres

The trajectories developed in [P1] may be evaluated in microseconds, and thus lend themselves to situations where the goal is to search over large sets of potential motions. The experimental section of [P1] exploits this to interact with an unpredictable environment, where specifically the task of catching a thrown ball implies that trajectories to catch the ball need to be generated on time scales much lower than the flight time of a ball.

An alternative use of such computationally inexpensive algorithms is to generate a comprehensive library of trajectories before the execution of a task, and then execute motions from this library during task execution. As an example of this, consider flying a quadrocopter through a cluttered but only partially mapped environment. Using a method such as RRT* [36] a nominal trajectory may be computed that guides the quadrocopter from an initial to a final goal. Once the optimal trajectory has been found that successfully avoids the known obstacles, a library of evasive manoeuvres and detours can then be computed, for example computing a tree of evasive manoeuvres for every

point along a discretised version of the nominal trajectory.

In flight, when executing the nominal trajectory, and when an obstacle is detected that was not known before, the vehicle may then look in its library of evasive manoeuvres for a trajectory that successfully avoids the newly detected obstacle, and execute this. Because looking data up from the stored data may be much faster than generating the trajectories in the first place (especially the collision detection is computationally expensive), such an approach may allow for the vehicle to fly with lower on-board computational requirements, reducing power and mass requirements and thus allowing for longer range and increased agility.

Such an approach lends itself directly to cloud computing and cloud robotics [37], where the quadrocopter may upload, for example, images of its flight space to the cloud, which then creates a rough map, computes the nominal trajectory, and generates the library of evasive manoeuvres. The nominal trajectory and the library are then downloaded to the vehicle, which proceeds to execute the trajectory.

Although such a library would not be generated in flight in real time, computational complexity is still important: the faster an individual trajectory may be computed and added to the library, the more detailed the library will be and thus also the more likely that a successful trajectory has been generated in advance. Furthermore, if the quadrocopter must wait for the generation of the library to accomplish the task, it is clear that the faster the library may be generated, the faster the task may be accomplished.

State estimation for rotating vehicles

The rapidly rotating vehicles introduced in [P3] potentially offer some unique opportunities for state estimation. As an example, consider the use of the magnetometer on a flying vehicle. Experience has shown that a magnetometer experiences significant and hard-to-predict disturbances when mounted near the vehicle's power electronics, battery, or motors. In such cases, use of the magnetometer in estimation is extremely problematic. If, in contrast, the vehicle is rotating rapidly, such body-fixed disturbances may be easily discriminated from world-fixed magnetic fields, from which the vehicle's yaw angle may be inferred.

Additionally, a position sensor attached to an extremity of such a vehicle would effectively measure points along a disk, centred about the vehicle's centre of rotation. The orientation of this disk of point measurements would allow to infer the vehicle's tilt (the Euler roll and pitch angles), while the phase allows to infer the vehicle's yaw angle.

The vehicle may also be used as a rotating sensor platform: for example, by sweeping a line-scanning camera the vehicle could be turned into an omnidirectional camera. However, in such a situation, the camera would have to be capable of low shutter speeds to minimise the effects of motion blur.

Extending the capabilities of the ultra-wideband system

Many directions exist for future work when combining the ultra-wideband radios with flying vehicles. If such a system were to be used outdoors, the no-wind assumption made

in [P6] would no longer apply and an important aspect would be the investigation of the effects of wind on the model, and methods for compensating for wind. One approach would be to include the wind speed as an additional state in the estimator, where a likely difficulty will be determining a suitable dynamic model for the wind (taking e.g. gusts into account).

An alternative method would be to forego the aerodynamic model used in [P6] completely, and instead use a model based solely on the integration of the accelerometers and rate gyroscopes. Such an estimator would be much more generic, and its performance would not be influenced by physical interactions with the environment (as these will be measured by the inertial sensors), at the cost of not exploiting the available model knowledge.

Attitude discontinuous trajectories

The trajectories generated in [P1], [P2] are continuous in the vehicle's attitude, but not in its angular velocity. This is motivated by the angular agility of the vehicles, such that the angular dynamics may be neglected to yield a simplified motion planning problem.

For certain tasks, and vehicle configurations, it could be interesting to extend this, and to assume that the vehicle's attitude can also be instantaneously changed, such that the vehicle's acceleration becomes a planning input (instead of the vehicle's jerk). Although clearly physically impossible, such trajectories could be much more computationally lightweight than those proposed in [P1]. As such, they may be interesting to use as a first step in randomised path planning, such as RRT* [36]. Once such a trajectory has been generated, one could search for a physically feasible trajectory that closely approximates it.

Trajectories for vehicles with alternative dynamics

The computational speed of the trajectories presented in [P1] is primarily due to exploitation of structure in the quadrocopter's dynamics, and the two step approach (generate a motion without regards for constraints, then test for feasibility in a second step). It would be interesting to investigate whether a similar approach could be useful to alternative vehicles, for example ground-based vehicles, or aerial vehicles with different dynamics.

References for Chapters 1-3

[1] R. D'Andrea, "Can drones deliver?", *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 3, pp. 647–648, 2014.

[2] W. Selby, P. Corke, and D. Rus, "Autonomous aerial navigation and tracking of marine animals", in *Proceedings of the 2011 Australasian Conference on Robotics and Automation*, Australian Robotics & Automation Association, 2011, pp. 1–7.

[3] J. Rasmussen, J. Nielsen, F. Garcia-Ruiz, S. Christensen, and J. C. Streibig, "Potential uses of small unmanned aircraft systems (UAS) in weed research", *Weed Research*, vol. 53, no. 4, pp. 242–248, 2013.

[4] J. R. Hörandel, S. Buitink, A. Corstanje, J. E. Enriquez, and H. Falcke, "The LOFAR radio telescope as a cosmic ray detector", in *International Cosmic Ray Conference*, 2013.

[5] F. Augugliaro, A. Mirjan, F. Gramazio, M. Kohler, and R. D'Andrea, "Building tensile structures with flying machines", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2013, pp. 3487–3492.

[6] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment", *Control Systems, IEEE*, vol. 28, no. 2, pp. 51–64, Apr. 2008.

[7] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Precision flight control for a multi-vehicle quadrotor helicopter testbed", *Control Engineering Practice*, vol. 19, no. 9, pp. 1023–1036, 2011.

[8] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed", *IEEE Robotics Automation Magazine*, vol. 17, no. 3, pp. 56–65, Sep. 2010.

[9] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter trajectory tracking control", in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, USA, Aug. 2008, pp. 1–14.

[10] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "A prototype of an autonomous controller for a quadrotor UAV", in *European Control Conference (ECC)*, Kos, Greece, 2007, pp. 1–8.

[11] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter", in *Mediterranean Conference on Control and Automation*, Jun. 2008, pp. 1258–1263.

[12] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors", in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011, pp. 2520–2525.

[13] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control", in *IFAC World Congress*, vol. 18, 2011, pp. 1485–1491.

[14] A. Boeuf, J. Cortes, R. Alami, and T. Siméon, "Planning agile motions for quadrotors in constrained environments", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2014, pp. 218–223.

[15] M. P. Vitus, W. Zhang, and C. J. Tomlin, "A hierarchical method for stochastic motion planning in uncertain environments", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Villamoura, Portugal, 2012, pp. 2263–2268.

[16] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for quadrotor flight", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[17] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "Direct method based control system for an autonomous quadrotor", *Journal of Intelligent & Robotic Systems*, vol. 60, no. 2, pp. 285–316, 2010.

[18] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization", *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[19] A. Domahidi, A. Zgraggen, M. Zeilinger, M. Morari, and C. Jones, "Efficient interior point methods for multistage horizons arising in receding horizon control", in *IEEE Conference on Decision and Control (CDC)*, 2012, pp. 668–674.

[20] S. Richter, "Computational complexity certification of gradient methods for real-time model predictive control", PhD thesis, ETH Zurich, 2012.

[21] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Autonomous navigation and exploration of a quadrotor helicopter in GPS-denied indoor environments", in *Robotics: Science and Systems Conference*, Jun. 2008.

[22] M. C. Achtelik, K.-M. Doth, D. Gurdan, and J. Stumpf, "Design of a multi rotor MAV with regard to efficiency, dynamics and redundancy", in *AIAA Guidance, Navigation, and Control Conference*, 2012, pp. 1–17.

[23] A. Marks, J. F. Whidborne, and I. Yamamoto, "Control allocation for fault tolerant control of a VTOL octorotor", in *UKACC International Conference on Control*, IEEE, 2012, pp. 357–362.

[24] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[25] J. Stuelpnagel, "On the parametrization of the three-dimensional rotation group", *SIAM review*, vol. 6, no. 4, pp. 422–430, 1964.

[26] J. L. Farrell, "Attitude determination by Kalman filtering", *Automatica*, vol. 6, no. 3, pp. 419–430, 1970.

[27] F. L. Markley, "Attitude error representations for Kalman filtering", *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[28] J. L. Crassidis and F. L. Markley, "Unscented filtering for spacecraft attitude estimation", *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 4, pp. 536–542, 2003.

[29] F. L. Markley, J. L. Crassidis, and Y. Cheng, "Nonlinear attitude filtering methods", in *AIAA Guidance, Navigation, and Control Conference*, 2005, pp. 15–18.

[30] J. L. Crassidis, F. L. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods", *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[31] F. L. Markley and J. L. Crassidis, "Filtering for attitude estimation and calibration", in *Fundamentals of Spacecraft Attitude Determination and Control*, Springer, 2014, pp. 235–285.

[32] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation", *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[33] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: A compendium", *Journal of Field Robotics*, vol. 30, no. 5, pp. 803–831, 2013.

[34] L. Meier, P. Tanskanen, L. Heng, G. Lee, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision", English, *Autonomous Robots*, vol. 33, no. 1-2, pp. 21–39, 2012.

[35] *GNU general public license*, version 3, Free Software Foundation, Jun. 29, 2007. [Online]. Available: `http://www.gnu.org/licenses/gpl.html`.

[36] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning", *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[37] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schiessle, M. Tenorth, O. Zweigle, and R. van de Molengraft, "RoboEarth: A world wide web for robots", *IEEE Robotics & Automation Magazine*, vol. 18, no. 2, pp. 69–82, 2011.

Part A
Trajectory generation

Paper P1

A computationally efficient motion primitive for quadrocopter trajectory generation

Mark W. Mueller, Markus Hehn, and Raffaello D'Andrea

Abstract

A method is presented for the rapid generation and feasibility verification of motion primitives for quadrocopters and similar multirotor vehicles. The motion primitives are defined by the quadrocopter's initial state, the desired motion duration, and any combination of components of the quadrocopter's position, velocity and acceleration at the motion's end. Closed form solutions for the primitives are given, which minimize a cost function related to input aggressiveness. Computationally efficient tests are presented to allow for rapid feasibility verification. Conditions are given under which the existence of feasible primitives can be guaranteed a priori. The algorithm may be incorporated in a high-level trajectory generator, which can then rapidly search over a large number of motion primitives which would achieve some given high-level goal. It is shown that a million motion primitives may be evaluated and compared per second on a standard laptop computer. The motion primitive generation algorithm is experimentally demonstrated by tasking a quadrocopter with an attached net to catch a thrown ball, evaluating thousands of different possible motions to catch the ball.

Accepted for publication in *IEEE Transactions on Robotics*.

1. Introduction

Quadrocopters offer exceptional agility, with typically high thrust-to-weight ratios, and large potential for angular acceleration due to the outward mounting of the propellers. This allows them to perform complex and highly dynamic tasks, for example aerial manipulation [1.1] and cooperative aerial acrobatics [1.2]. The ability to hover, and the safety offered by small rotors storing relatively little energy [1.3], make quadrocopters attractive platforms for aerial robotic tasks that involve the navigation of tight, cluttered environments (for example, [1.4], [1.5]).

A key feature required for the use of these vehicles under complex conditions is a trajectory generator. The trajectory generator is tasked with computing flight paths that achieve the task objective, while respecting the quadrocopter dynamics. The trajectory must also be collision-free, and there could be additional requirements imposed on the motion by the sensing modalities (for example, limits on the quadrocopter velocity imposed by an onboard camera). This trajectory planning problem is complicated by the underactuated and nonlinear nature of the quadrocopter dynamics, as well as potentially complex task constraints that may consist of variable manoeuvre durations, partially constrained final states, and non-convex state constraints. In addition, dynamic environments or large disturbances may require the re-computation or adaptation of trajectories in real time, thus limiting the time available for computation.

Active research in this field has yielded numerous trajectory generation algorithms, focusing on different trade-offs between computational complexity, the agility of the possible motions, the level of detail in which manoeuvre constraints can be specified, and the ability to handle complex environments.

Broadly speaking, a first group of algorithms handles the trajectory generation problem by decoupling geometric and temporal planning: in a first step, a geometric trajectory without time information is constructed, for example using lines [1.6], polynomials [1.7], or splines [1.8]. In a second step, the geometric trajectory is parametrised in time in order to guarantee feasibility with respect to the dynamics of quadrocopters.

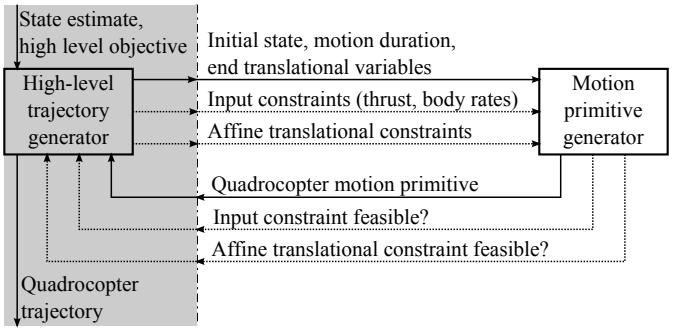

Figure 1.1. The presented algorithm aims to provide computationally inexpensive motion primitives, which may then be incorporated by a high-level trajectory generator. The focus of this paper is on the right-hand-side (unshaded) part of this diagram.

A second group of algorithms exploits the differential flatness of the quadrocopter dynamics in order to derive constraints on the trajectory, and then solves an optimization problem over a class of trajectories, for example minimum snap [1.9], minimum time [1.10], shortest path under uncertain conditions [1.11], or combinations of position derivatives [1.5]. In [1.12] a search over parameters is proposed for quadrocopter motion planning, including trajectories where the position is described by polynomials in time. A dynamic inversion based controller is proposed in [1.13] to directly control a quadrocopter's position and orientation, and this controller is exploited in [1.14]. For a broader discussion of differential flatness see e.g. [1.15], [1.16], and e.g. [1.17], [1.18] for generic trajectory generation methods for differentially flat systems.

A common property of these methods is that they impose a rigid structure on the end state, for example fixing the final state and allowing a fixed or varying manoeuvre duration, or by specifying the goal state with convex inequalities. Many quadrocopter applications do not, however, impose such structured constraints; instead the set of states that achieve the application might be non-convex, or even disjoint. Furthermore, the conditions on the final state necessary to achieve a task may be time-dependent (for example when the task objective involves interaction with a dynamic environment). Methods relying on convex optimisation furthermore require the construction of (conservative) convex approximations of constraints, potentially significantly reducing the space of feasible trajectories.

This paper attempts a different approach to multicopter trajectory generation, where the constraints are not explicitly encoded at the planning stage. Instead, the focus is on developing a computationally light-weight and easy-to-implement motion primitive, which can be easily tested for constraints violation, and allows significant flexibility in defining initial and final conditions for the quadrocopter. The low computational cost may then be exploited by searching over a multitude of primitives to achieve some goal. Each primitive is characterised by the quadrocopter's initial state, a duration, and a set of constraints on the quadrocopter's position, velocity, and/or acceleration at the end of the primitive.

The approach is illustrated in Fig. 1.1. The high-level trajectory generator is tasked with evaluating motion primitives, and specifically with defining the constraints on the motion primitives to solve the given high-level goal. The high-level trajectory generator must also encode the behaviour for dealing with infeasible motion primitives. As an example, the high-level trajectory generator may generate a large number of motion primitives, with varying durations and end variables, to increase the probability of finding a feasible motion primitive. These motion primitives are generated in a two-step approach: First, a state-to-state planner is used to generate a motion while disregarding feasibility constraints. In the second step, this trajectory is checked for feasibility. The first step is solved for in closed form, while a computationally efficient recursive test is designed for the feasibility tests of the second step.

The state-to-state motion primitives generated in the first step are closely related to other algorithms exploiting the differential flatness of the quadrocopter dynamics to plan

position trajectories that are polynomials in time (e.g. [1.5], [1.9], [1.12]). In this paper an optimal control problem is solved, whose objective function is related to minimizing an upper bound of the product of the quadrocopter inputs, to yield position trajectories characterised by fifth order polynomials in time. A key property that is then exploited is that the specific polynomials allow for the rapid verification of constraints on the system's inputs, and constraints on the position, velocity, and/or acceleration.

The benefits of this approach are twofold: first, a unified framework is given to generate trajectories for arbitrary manoeuvre duration and end state constraints, resulting in an algorithm which can be easily implemented across a large range of trajectory generation problems. Secondly, the computational cost of the approach is shown to be very low, such that on the order of one million motion primitives per second can be generated and tested for feasibility on a laptop computer with an unoptimised implementation.

The algorithm therefore lends itself to problems with significant freedom in the end state. In this situation, the designer can apply the presented approach to rapidly search over the space of end states and trajectory durations which would achieve the high level goal. This ability to quickly evaluate a vast number of candidate trajectories is achieved at the expense of not directly considering the feasibility constraints in the trajectory generation phase, but rather verifying feasibility a posteriori.

For certain classes of trajectories, explicit guarantees can be given on the existence of feasible motion primitives as a function of the problem data. Specifically, for rest-to-rest manoeuvres, a bound on the motion primitive duration is explicitly calculated as a function of the distance to be translated and the system's input constraints. Furthermore, bounds on the velocity during the rest-to-rest manoeuvre are explicitly calculated, and it is shown that the position feasibility of rest-to-rest trajectories can be directly asserted if the allowable flight space is convex.

An experimental demonstration of the algorithm is presented where the motion primitive generator is encapsulated in a high-level trajectory generation algorithm. The goal is for a quadrocopter with an attached net to catch a thrown ball. The catching trajectories must be generated in real time, because the ball's flight is not repeatable. Furthermore, for a given ball trajectory, the ball can be caught in many different ways (quadrocopter positions and orientations). The computational efficiency of the presented approach is exploited to do an exhaustive search over these possibilities in real time.

An implementation of the algorithm presented in this paper in both C++ and Python is made available at [1.19].

This paper follows on previous work presented at conferences [1.20], [1.21]. A related cost function, the same dynamics model, and a related decoupled planning approach were presented in [1.20]. Preliminary results of the fundamental state-to-state motion primitive generation algorithm were presented in [1.21]. This paper extends these previous results by presenting

- conditions under which primitives are guaranteed to be feasible;
- an investigation into the completeness of the approach, by comparing rest-to-rest

trajectories to the time optimal rest-to-rest trajectories; and

- presenting a challenging novel demonstration to show the capabilities of the approach.

The remainder of this paper is organised as follows: the quadrocopter model and problem statement are given in Section 2, with the motion primitive generation scheme presented in Section 3. A computationally efficient algorithm to determine feasibility of generated trajectories is presented in Section 4. The choice of coordinate system is discussed in Section 5. In Section 6 classes of problems are discussed where the existence of feasible trajectories can be guaranteed. The performance of the presented approach is compared to the system's physical limits in Section 7. The computational cost of the algorithm is measured in Section 8, and the demonstration of catching a ball is presented in Section 9. A conclusion is given in Section 10.

2. System dynamics and problem statement

This section describes the dynamic model used to describe the quadrocopter's motion, including constraints on the quadrocopter inputs. A formal statement of the problem to be solved in this paper is then given, followed by an overview of the solution approach.

2.1 Quadrocopter dynamic model

The quadrocopter is modelled as a rigid body with six degrees of freedom: linear translation along the orthogonal inertial axes, $\boldsymbol{x} = (x_1, x_2, x_3)$, and three degrees of freedom describing the rotation of the frame attached to the body with respect to the inertial frame, defined by the proper orthogonal matrix $\mathbf{R}$. Note that the notation $\boldsymbol{x} = (x_1, x_2, x_3)$ is used throughout this paper to compactly express the elements of a column vector.

The control inputs to the system are taken as the scalar total thrust produced f, for simplicity normalised by the vehicle mass and thus having units of acceleration; and the body rates expressed in the body-fixed frame as $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)$, as are illustrated in Fig. 1.2. It is assumed that high bandwidth controllers are used to track these angular rate commands. Then by separation of timescales, because of the vehicles' low rotational inertia and their ability to produce large torques, it is assumed that the angular rate commands are tracked perfectly and that angular dynamics may be neglected. The quadrocopter's state is thus nine dimensional, and consists of the position, velocity, and orientation.

Although more complex quadrocopter models exist that incorporate, for example, aerodynamic drag [1.22] or propeller speeds [1.23], the preceding model captures the most relevant dynamics, and yields tractable solutions to the trajectory generation problem. Furthermore, in many applications (for example, model predictive control) such a simple model is sufficient, with continuous replanning compensating for modelling inaccuracies.

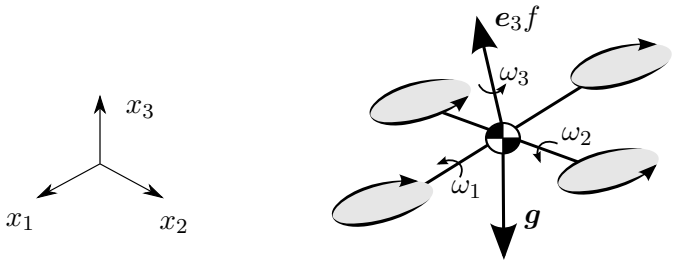

Figure 1.2. Dynamic model of a quadrocopter, acted upon by gravity $\boldsymbol{g}$, a thrust force f pointing along the (body-fixed) axis $\boldsymbol{e}_3$; and rotating with angular rate $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)$, with its position in inertial space given as (x_1, x_2, x_3).

The differential equations governing the flight of the quadrocopter are now taken as those of a rigid body [1.24]

$$\ddot{\boldsymbol{x}} = \mathbf{R}\, \boldsymbol{e}_3 f + \boldsymbol{g} \tag{1.1}$$

$$\dot{\mathbf{R}} = \mathbf{R}\, [\![\boldsymbol{\omega}\times]\!] \tag{1.2}$$

with $\boldsymbol{g}$ the acceleration due to gravity as expressed in the inertial coordinate frame, $\boldsymbol{e}_3 = (0, 0, 1)$ a constant vector in the body-fixed frame, as illustrated in Fig. 1.2. Finally, $[\![\boldsymbol{\omega}\times]\!]$ is the skew-symmetric matrix form of the vector cross product such that

$$[\![\boldsymbol{\omega}\times]\!] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{1.3}$$

It should be noted that the preceding model may also be applied to other multirotor vehicles, such as hexa- and octocopters. This is because the high-bandwidth angular rate controller that maps angular velocity errors to motor forces effectively hides the number and locations of the propellers.

1) Feasible inputs The achievable thrust f produced by the vehicle lies in the range

$$0 \leq f_{\min} \leq f \leq f_{\max} \tag{1.4}$$

where $f_{\min}$ is non-negative because of the fixed sense of rotation of the fixed-pitch propellers. The magnitude of the angular velocity command is taken to be constrained to lie within a ball:

$$\|\boldsymbol{\omega}\| \leq \omega_{\max} \tag{1.5}$$

with $\|\cdot\|$ the Euclidean norm. This limit could be due, for example, to saturation limits of the rate gyroscopes, or motion limits of cameras mounted on the vehicle. Alternatively, a designer may use this value as a tuning factor, to encode the dynamic regime where the low-order dynamics of (1.1)-(1.2) effectively describe the true quadrocopter. The Euclidean norm is chosen for computational convenience, specifically invariance under rotation.

2.2 Problem statement

Define $\sigma(t)$ to be translational variables of the quadrocopter, consisting of its position, velocity and acceleration, such that

$$\sigma(t) = (\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t)) \in \mathbb{R}^9. \tag{1.6}$$

Let T be the goal duration of the motion, and let $\hat{\sigma}_i$ be components of desired translational variables at the end of the motion, for some $i \in \mathcal{I} \subseteq \{1, 2, \ldots, 9\}$. Let the trajectory goal be achieved if

$$\sigma_i(T) = \hat{\sigma}_i \; \forall i \in \mathcal{I}. \tag{1.7}$$

Furthermore, the quadrocopter is subject to N_c translational constraints of the form

$$a_j^T \sigma(t) \leq b_j \text{ for } t \in [0, T]\,, \; j = 1, 2, \ldots, N_c. \tag{1.8}$$

An interpretation of these translational constraints is given at the end of this section.

The problem addressed in this paper is to find quadrocopter inputs $f(t), \boldsymbol{\omega}(t)$ over $[0, T]$, for a quadrocopter starting at some initial state consisting of position, velocity, and orientation, at time $t = 0$ to an end state at time T satisfying (1.7), while satisfying throughout the trajectory:

- the vehicle dynamics (1.1)-(1.2),
- the input constraints (1.4)-(1.5), and
- the N_c affine translational constraints (1.8).

Discussion It should be noted that the nine quadrocopter state variables as described in Section 2.1 (three each for position, velocity and orientation) are not the nine translational variables. Only eight components of the state are encoded in the translational variables, consisting of the quadrocopter's position, its velocity, and two components of the orientation (which are encoded through the acceleration). These two orientation components are those that determine the direction of the quadrocopter's thrust vector – given an acceleration value, the quadrocopter's thrust direction $\mathbf{R}\boldsymbol{e}_3$ can be recovered through (1.1). These are the same components encoded by the Euler roll and pitch angles.

The translational variables do not encode the quadrocopter's rotation about the thrust axis (the Euler yaw angle).

These variables are chosen because they are computationally convenient, whilst still being useful to encode many realistic problems. Two example problems are given below:

(i) To-rest manoeuvre: let the goal be to arrive at some point in space, at rest, at time T. Then all nine translational variables will be specified in (1.7), with specifically the components corresponding to velocity and acceleration set to zero. Similarly, if the goal is simply to arrive at a point in space, but the final velocity and acceleration do not matter, only the first three components of $\hat{\sigma}$ are specified.

(ii) Landing on a moving platform: to land the quadrocopter on a moving, possibly slanted platform, the goal end position and velocity are set equal to those of the platform at time T. The end acceleration is specified to be such that the quadrocopter's thrust vector e_3 is parallel to the normal vector of the landing platform. Then the quadrocopter will arrive with zero relative speed at the platform, and touch down flatly with respect to the platform.

The affine translational constraints of (1.8) are also chosen for computational convenience. They allow to encode, for example, that the position may not intersect a plane (such as the ground, or a wall), or specify a box constraint on the vehicle velocity. Constraints on the acceleration, in conjunction with (1.4), can be interpreted as limiting the tilt of the quadrocopter, by (1.1).

3. Motion primitive generation

Given an end time T and goal translational variables $\hat{\sigma}_i$, the dynamic model of Section 2.1 is used to generate motion primitives guiding the quadrocopter from some initial state to a state achieving the goal translational variables. The input constraints, and the affine state constraints, are ignored at this stage, and the motion primitives are generated in the quadrocopter's jerk. Each of the three spatial axes is solved for independently. The generated motion primitive minimises a cost value for each axis independently of the other axes, but the total cost is shown to be representative of the aggressiveness of the true system inputs. Constraints on the input and state are considered in Sections 4 and 6.

3.1 Formulating the dynamics in jerk

We follow [1.10] in planning the motion of the quadrocopter in terms of the jerk along each of the axes, allowing the system to be considered as a triple integrator in each axis. By ignoring the input constraints, the axes can be decoupled, and motions generated for each axis separately. These decoupled axes are then recombined in later sections when considering feasibility. This subsection will describe how to recover the thrust and body rates inputs from such a thrice differentiable trajectory.

Given a thrice differentiable motion $\boldsymbol{x}(t)$, the jerk is written as $\boldsymbol{j} = \dddot{\boldsymbol{x}} = (\dddot{x}_1, \dddot{x}_2, \dddot{x}_3)$.

The input thrust f is computed by applying the Euclidean norm to (1.1):

$$f = \|\ddot{\boldsymbol{x}} - \boldsymbol{g}\|. \tag{1.9}$$

Taking the first derivative of (1.1) and (1.9), yields

$$\boldsymbol{j} = \mathbf{R}[\![\boldsymbol{\omega}\times]\!]\boldsymbol{e}_3 f + \mathbf{R}\boldsymbol{e}_3\dot{f} \tag{1.10}$$

$$\dot{f} = \boldsymbol{e}_3^T\mathbf{R}^{-1}\boldsymbol{j}. \tag{1.11}$$

After substitution, and evaluating the product $[\![\boldsymbol{\omega}\times]\!]\boldsymbol{e}_3$, it can be seen that the jerk $\boldsymbol{j}$ and thrust f fix two components of the body rates:

$$\begin{bmatrix} \omega_2 \\ -\omega_1 \\ 0 \end{bmatrix} = \frac{1}{f}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{R}^{-1}\boldsymbol{j}. \tag{1.12}$$

Note that ω_3 does not affect the linear motion, and is therefore not specified. Throughout the rest of the paper, it will be taken as $\omega_3 = 0$.

3.2 Cost function

The goal of the motion primitive generator is to compute a thrice differentiable trajectory which guides the quadrocopter from an initial state to a (possibly only partially defined) final state in a final time T, while minimizing the cost function

$$J_\Sigma = \frac{1}{T}\int_0^T \|\boldsymbol{j}(t)\|^2\, dt. \tag{1.13}$$

This cost function has an interpretation as an upper bound on the average of a product of the inputs to the (nonlinear, coupled) quadrocopter system: rewriting (1.12), and taking $\omega_3 = 0$ gives

$$\begin{aligned} f^2\|\boldsymbol{\omega}\|^2 &= \left\| f\begin{bmatrix} \omega_2 \\ -\omega_1 \\ 0 \end{bmatrix}\right\|^2 = \left\|\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{R}^{-1}\boldsymbol{j}\right\|^2 \\ &\leq \|\boldsymbol{j}\|^2 \end{aligned} \tag{1.14}$$

so that:

$$\frac{1}{T}\int_0^T f(t)^2\|\boldsymbol{\omega}(t)\|^2\, dt \leq J_\Sigma. \tag{1.15}$$

If multiple motion primitives exist that all achieve some high-level goal, this cost function may thus be used to rank the input aggressiveness of the primitives. The cost function is also computationally convenient, and closed form solutions for the optimal trajectories are given below.

3.3 Axes decoupling and trajectory generation

The nonlinear trajectory generation problem is simplified by decoupling the dynamics into three orthogonal inertial axes, and treating each axis as a triple integrator with jerk used as control input. The true control inputs f and $\boldsymbol{\omega}$ are then recovered from the jerk inputs using (1.9) and (1.12). The final state is determined from the goal end state components $\hat{\sigma}_i$ relevant to each axis, and the duration T is given.

The cost function of the three dimensional motion, J_Σ, is decoupled into a per-axis cost J_k by expanding the integrand in (1.13).

$$J_\Sigma = \sum_{k=1}^{3} J_k, \quad \text{where } J_k = \frac{1}{T}\int_0^T j_k(t)^2\,dt \tag{1.16}$$

For each axis k, the state $s_k = (p_k, v_k, a_k)$ is introduced, consisting of the scalars position, velocity, and acceleration. The jerk j_k is taken as input, such that

$$\dot{s}_k = f_s(s_k, j_k) = (v_k, a_k, j_k)\,. \tag{1.17}$$

Note again that the input constraints of Section 2.1 are not considered here, during the planning stage, but are deferred to Sections 4 and 6.

For the sake of readability, the axis subscript k will be discarded for the remainder of this section where only a single axis is considered. The time argument t will similarly be neglected where it is considered unambiguous.

The optimal state trajectory can be solved straightforwardly with Pontryagin's minimum principle (see e.g. [1.25]) by introducing the costate $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ and defining the Hamiltonian function $H(s, j, \lambda)$:

$$\begin{aligned} H(s, j, \lambda) &= \frac{1}{T}j^2 + \lambda^T f_s(s, j) \\ &= \frac{1}{T}j^2 + \lambda_1 v + \lambda_2 a + \lambda_3 j \qquad (1.18) \\ \dot{\lambda} &= -\nabla_s H(s^*, j^*, \lambda) = (0, -\lambda_1, -\lambda_2) \qquad (1.19) \end{aligned}$$

where j_k^* and s_k^* represent the optimal input and state trajectories, respectively.

The costate differential equation (1.19) is easily solved, and for later convenience the

solution is written in the constants α, β and γ, such that

$$\lambda(t) = \frac{1}{T}\begin{bmatrix} -2\alpha \\ 2\alpha t + 2\beta \\ -\alpha t^2 - 2\beta t - 2\gamma \end{bmatrix}. \tag{1.20}$$

The optimal input trajectory is solved for as

$$\begin{aligned} j^*(t) &= \arg\min_{j(t)} H(s^*(t), j(t), \lambda(t)) \\ &= \frac{1}{2}\alpha t^2 + \beta t + \gamma \end{aligned} \tag{1.21}$$

from which the optimal state trajectory follows from integration of (1.17):

$$s^*(t) = \begin{bmatrix} \frac{\alpha}{120}t^5 + \frac{\beta}{24}t^4 + \frac{\gamma}{6}t^3 + \frac{a_0}{2}t^2 + v_0 t + p_0 \\ \frac{\alpha}{24}t^4 + \frac{\beta}{6}t^3 + \frac{\gamma}{2}t^2 + a_0 t + v_0 \\ \frac{\alpha}{6}t^3 + \frac{\beta}{2}t^2 + \gamma t + a_0 \end{bmatrix} \tag{1.22}$$

with the integration constants set to satisfy the initial condition $s(0) = (p_0, v_0, a_0)$.

The remaining unknowns α, β and γ are solved for as a function of the desired end translational variable components $\hat{\sigma}_i$ as given in (1.7).

1) Fully defined end translational state Let the desired end position, velocity, and acceleration along this axis be $s(T) = (p_f, v_f, a_f)$, given by the components $\hat{\sigma}_i$. Then the unknowns α, β and γ are isolated by reordering (1.22):

$$\begin{bmatrix} \frac{1}{120}T^5 & \frac{1}{24}T^4 & \frac{1}{6}T^3 \\ \frac{1}{24}T^4 & \frac{1}{6}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} \Delta p \\ \Delta v \\ \Delta a \end{bmatrix} \tag{1.23}$$

where

$$\begin{bmatrix} \Delta p \\ \Delta v \\ \Delta a \end{bmatrix} = \begin{bmatrix} p_f - p_0 - v_0 T - \frac{1}{2}a_0 T^2 \\ v_f - v_0 - a_0 T \\ a_f - a_0 \end{bmatrix}. \tag{1.24}$$

Solving for the unknown coefficients yields

$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \frac{1}{T^5}\begin{bmatrix} 720 & -360T & 60T^2 \\ -360T & 168T^2 & -24T^3 \\ 60T^2 & -24T^3 & 3T^4 \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta v \\ \Delta a \end{bmatrix}. \tag{1.25}$$

Thus, generating a motion primitive only requires evaluating the above matrix multiplica-

tion for each axis, after which the state along the primitive is found by evaluating (1.22).

2) Partially defined end translational state Components of the final state may be left free by $\hat{\sigma}$. These states may correspondingly be specified as free when solving the optimal input trajectory, by noting that the corresponding costates must equal zero at the end time [1.25]. The closed form solutions to the six different combinations of partially defined end states are given in Appendix A – in each case solving the coefficients reduces to evaluating a matrix product.

3) Motion primitive cost The per-axis cost value of (1.16) can be explicitly calculated as below. This is useful if multiple different candidate motion primitives are evaluated to achieve some higher-level goal. In this case, the primitive with the lowest cost can be taken as the 'least aggressive' in the sense of (1.14).

$$J = \gamma^2 + \beta\gamma T + \frac{1}{3}\beta^2 T^2 + \frac{1}{3}\alpha\gamma T^2 + \frac{1}{4}\alpha\beta T^3 + \frac{1}{20}\alpha^2 T^4 \tag{1.26}$$

Note that this cost holds for all combinations of end translational variables.

4. Determining feasibility

The motion primitives generated in the previous section did not take the input feasibility constraints of Section 1 into account – this section revisits these and provides computationally inexpensive tests for the feasibility/infeasibility of a given motion primitive with respect to the input constraints of (1.4) and (1.5). This section also provides a computationally inexpensive method to calculate the extrema of an affine combination of the translational variables along the primitive, allowing to test constraints of the form (1.8). In Section 6 conditions are given under which feasible motion primitives are guaranteed to exist.

4.1 Input feasibility

Given some time interval $\mathcal{T} = [\tau_1, \tau_2] \subseteq [0, T]$ and three triple integrator trajectories of the form (1.22) with their corresponding jerk inputs $j_k(t)$, the goal is to determine whether corresponding inputs to the true system f and $\boldsymbol{\omega}$, as used in (1.1) and (1.2), satisfy feasibility requirements of Section 1. The choice of $\mathcal{T}$ is revisited when describing the recursive implementation, below. The tests are designed with a focus on computational speed, and provide sufficient, but not necessary, conditions for both feasibility and infeasibility – meaning that some motion primitives will be indeterminable with respect to these tests.

1) Thrust The interval $\mathcal{T}$ is feasible with respect to the thrust limits (1.4) if and only if

$$\max_{t\in\mathcal{T}} f(t)^2 \leq f_{\max}^2 \text{ and} \tag{1.27}$$

$$\min_{t\in\mathcal{T}} f(t)^2 \geq f_{\min}^2. \tag{1.28}$$

Squaring (1.9) yields

$$f^2 = \|\ddot{\boldsymbol{x}} - \boldsymbol{g}\|^2 = \sum_{k=1}^{3} (\ddot{x}_k - g_k)^2 \tag{1.29}$$

where g_k is the component of gravity in axis k. Combining (1.27)-(1.29), the thrust constraints can be interpreted as spherical constraints on the acceleration.

By taking the per-axis extrema of (1.29) the below bounds follow.

$$\max_{t\in\mathcal{T}} (\ddot{x}_k(t) - g_k)^2 \leq \max_{t\in\mathcal{T}} f(t)^2 \text{ for } k \in \{1,2,3\} \tag{1.30}$$

$$\max_{t\in\mathcal{T}} f(t)^2 \leq \sum_{k=1}^{3} \max_{t\in\mathcal{T}} (\ddot{x}_k(t) - g_k)^2 \tag{1.31}$$

$$\min_{t\in\mathcal{T}} f(t)^2 \geq \sum_{k=1}^{3} \min_{t\in\mathcal{T}} (\ddot{x}_k(t) - g_k)^2 \tag{1.32}$$

These bounds will be used as follows: if the left hand side of (1.30) is greater than $f_{\max}$, the interval is definitely infeasible, while if both the right hand side of (1.31) is less than $f_{\max}$ and the right hand side of (1.32) is greater than $f_{\min}$, the interval is definitely feasible with respect to the thrust constraints.

Note that the value $\ddot{x}_k - g_k$ as given by (1.22) is a third order polynomial in time, meaning that its maximum and minimum (denoted $\bar{\ddot{x}}_k$ and $\underline{\ddot{x}}_k$, respectively) can be found in closed form by solving for the roots of a quadratic and evaluating $\ddot{x}_k - g_k$ at at most two points strictly inside $\mathcal{T}$, and at the boundaries of $\mathcal{T}$. The extrema of $(\ddot{x}_k - g_k)^2$ then follow as

$$\max_{t\in\mathcal{T}} (\ddot{x}_k(t) - g_k)^2 = \max\{\bar{\ddot{x}}_k^2, \underline{\ddot{x}}_k^2\} \tag{1.33}$$

$$\min_{t\in\mathcal{T}} (\ddot{x}_k(t) - g_k)^2 = \begin{cases} \min\{\bar{\ddot{x}}_k^2, \underline{\ddot{x}}_k^2\} & \text{if } \bar{\ddot{x}}_k \cdot \underline{\ddot{x}}_k \geq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{1.34}$$

where $\bar{\ddot{x}}_k \cdot \underline{\ddot{x}}_k < 0$ implies a sign change (and thus a zero crossing) of $\ddot{x}_k(t) - g_k$ in $\mathcal{T}$.

Thus, from (1.30), a sufficient criterion for input infeasibility of the section is if

$$\max\left\{\bar{\ddot{x}}_k^2, \underline{\ddot{x}}_k^2\right\} > f_{\max}. \tag{1.35}$$

Similarly, from (1.31)-(1.32), a sufficient criterion for feasibility is if both

$$\sum_{k=1}^{3} \max\left\{\bar{\ddot{x}}_k^2, \underline{\ddot{x}}_k^2\right\} \leq f_{\max} \quad \text{and} \tag{1.36}$$

$$\sum_{k=1}^{3} \min\left\{\bar{\ddot{x}}_k^2, \underline{\ddot{x}}_k^2\right\} \geq f_{\min} \tag{1.37}$$

hold. If neither criterion (1.35) nor (1.36)-(1.37) applies, the section is marked as indeterminate with respect to thrust feasibility.

2) Body rates The magnitude of the body rates can be bounded as a function of the jerk and thrust as below:

$$\omega_1^2 + \omega_2^2 \leq \frac{1}{f^2}\left\|j\right\|^2. \tag{1.38}$$

This follows from squaring (1.12), and using the following induced norm:

$$\left\| \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right\| \leq 1. \tag{1.39}$$

The right hand side of (1.38) can be bounded from above by $\bar{\omega}^2$, defined as below, which then also provides an upper bound for the sum $\omega_1^2 + \omega_2^2$. The terms in the denominator are evaluated as in (1.34).

$$\omega_1^2 + \omega_2^2 \leq \frac{1}{f^2}\left\|j\right\|^2 \leq \bar{\omega}^2 = \frac{\sum_{k=1}^{3} \max_{t\in\mathcal{T}} j_k(t)^2}{\sum_{k=1}^{3} \min_{t\in\mathcal{T}} \left(\ddot{x}_k(t) - g_k\right)^2} \tag{1.40}$$

Using the above equation, and assuming $\omega_3 = 0$, the time interval $\mathcal{T}$ can be marked as feasible w.r.t. the body rate input if $\bar{\omega}^2 \leq \omega_{\max}^2$, otherwise the section is marked as indeterminate.

3) Recursive implementation The feasibility of a given time interval $\mathcal{T} \subseteq [0, T]$ is tested by applying the above two tests on $\mathcal{T}$. If both tests return feasible, $\mathcal{T}$ is input feasible and the algorithm terminates; if one of the tests returns infeasible, the algorithm terminates

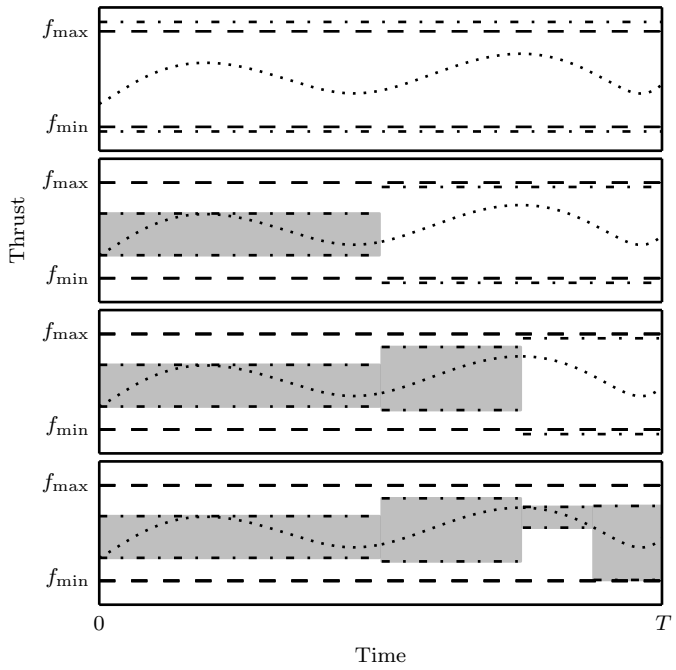

Figure 1.3. Visualisation of the recursive implementation of the thrust feasibility test: The vehicle thrust limits $f_{\min}$ and $f_{\max}$ are shown as dashed lines, and the thrust trajectory is the dotted curve. First, the minimum and maximum bounds of (1.36) and (1.37) are calculated for the entire motion primitive, shown as the dash-dotted lines in the top plot. Because these bounds exceed the vehicle limits, the section is halved, and the tests are applied to each half (second plot). The left hand side section's bounds are now within the limits, so this section is marked feasible with respect to the thrust bounds (shown shaded in the plot), while the second half is again indeterminate. This is halved again, in the third plot, and a section is yet again halved in the fourth plot. Now, all sections are feasible with respect to the thrust limits, and the test terminates. Note that, in the implementation, the thrust infeasibility test of (1.35) and the body rate feasibility test (1.40) will be done simultaneously.

with the motion over $\mathcal{T}$ marked as input infeasible. Otherwise, the section is divided in half, such that

$$\tau_{\frac{1}{2}} = \frac{\tau_1 + \tau_2}{2} \tag{1.41}$$

$$\mathcal{T}_1 = [\tau_1, \tau_{\frac{1}{2}}], \quad \mathcal{T}_2 = [\tau_{\frac{1}{2}}, \tau_2]. \tag{1.42}$$

If the time interval $\tau_{\frac{1}{2}} - \tau_1$ is smaller than some user defined minimum $\underline{\Delta\tau}$, the algorithm terminates with the primitive marked indeterminable. Otherwise, the algorithm is recursively applied first to $\mathcal{T}_1$: if the result is feasible, the algorithm is recursively applied to $\mathcal{T}_2$. If $\mathcal{T}_2$ also terminates as a feasible section, the entire primitive can be marked as feasible, otherwise the primitive is rejected as infeasible/indeterminable. Thus, the test returns one of three outcomes:

- the inputs are provably feasible,

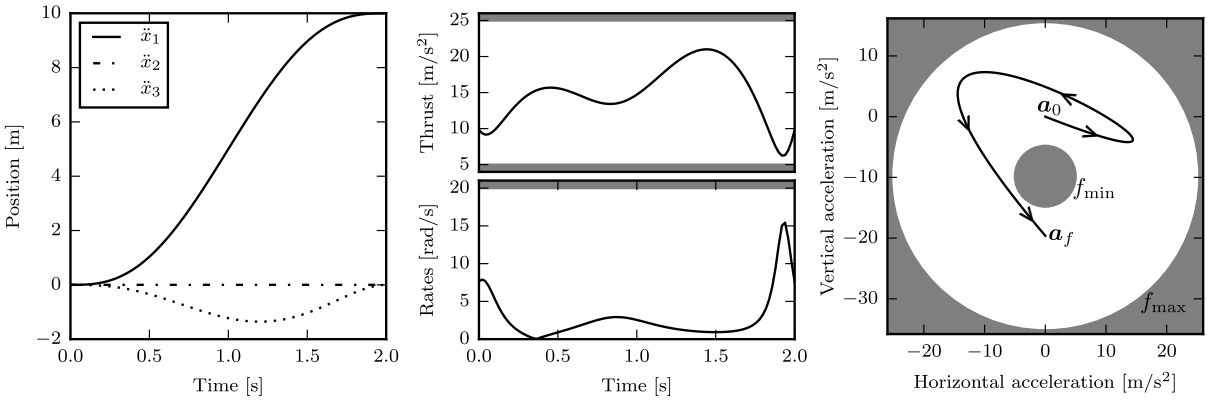

Figure 1.4. Example trajectory demonstrating non-convexity of the feasible acceleration space. The trajectory moves an initially stationary quadrocopter 10 m horizontally, ending at zero velocity and with final acceleration $\boldsymbol{a}_f = -2\boldsymbol{g}$ in 2 s (i.e. the quadrocopter is upside down at the end). The left-most plot shows the position trajectory of the vehicle, the middle plots show the inputs during the trajectory (with the shaded regions being infeasible). The right-hand side plot shows the acceleration trajectory in the acceleration space, where the two concentric circles represent the minimum and maximum thrust limits (and are centred at $\boldsymbol{g}$) for $\ddot{x}_2 = 0$. The axes are chosen such that x_3 points opposite to gravity, and there is no motion along x_2. Note the non-convexity of the feasible acceleration space.

- the inputs are provably infeasible, or
- the tests were indeterminate, feasibility could not be established.

Note that the interval upper bound of (1.33) is monotonically nonincreasing with decreasing length of the interval $\mathcal{T}$, and similarly the lower bound of (1.34) is monotonically nondecreasing.

Furthermore, note that as $\tau_{\frac{1}{2}} - \tau_1$ tends to zero the right hand sides of (1.31)-(1.32) converge and the thrust feasibility test becomes exact. This does not, however, apply to the body rate feasibility test due to the induced norm in (1.39).

The recursive implementation of the sufficient thrust feasibility tests of (1.36)-(1.37) is visualised for an example motion primitive in Fig. 1.3.

4) Remark on convex approximations of thrust-feasible region The feasible acceleration space of the vehicle follows from (1.27)-(1.29), and is non-convex due to the positive minimum thrust value. This is visualised in Fig. 1.4. Nonetheless, in the limit, the presented recursive thrust input test allows for testing feasibility over the entire thrust feasible space. This is an advantage when compared to approaches that require the construction of convex approximations of the feasible space (e.g. [1.20]). Consider, for example, a trajectory that begins with zero acceleration and ends with a final acceleration of $-2\boldsymbol{g}$ (i.e. the quadrocopter is upside down at the end of the manoeuvre) – such an example is shown in Fig. 1.4. The straight line connecting the initial and final accelerations crosses through the acceleration infeasible zone due to minimum thrust. Therefore no convex approximation can be constructed in the acceleration space in which to evaluate this

trajectory.

4.2 Affine translational variable constraints

Referring to (1.22), it can be seen that calculating the range of some linear combination of the system's translational variables $\sigma(t)$ (of the form (1.8)) can be done by solving for the extrema of a polynomial of order at most five. This involves finding the roots of its derivative (a polynomial of order at most four) for which closed form solutions exist [1.26].

This is useful, for example, to verify that the quadrocopter does not fly into the ground, or that the position of the quadrocopter remains within some allowable flight space. Specifically, any planar position constraint can be specified by specifying that the inner product of the quadrocopter's position with the normal of the plane is greater than some constant value. It can also be used to calculate a bound on the vehicle's maximum velocity, which could be useful in some computer vision applications (e.g. [1.27]). Furthermore, an affine bound on the vehicle's acceleration can be interpreted as a bound on the tilt of the quadrocopter's $\boldsymbol{e}_3$ axis, through (1.1).

5. Choice of coordinate system

Because the Euclidean norm used in the cost (1.13) is invariant under rotation, the optimal primitive for some given problem will be the same when the problem is posed in any inertial frame, despite the axes being solved for independently of one another.

The Euclidean norm is also used in the feasibility tests of Section 4. In the limit, as the length of the time interval $\mathcal{T}$ tends to zero, both the thrust and body rates feasibility tests become invariant under rotation, and thus independent of the choice of coordinate system. The affine constraints of Section 4.2 can be trivially transformed from one coordinate system to another, such that there exists no preferred coordinate system for a set of constraints. This allows the designer the freedom to pose the motion primitive generation problem in the most convenient inertial coordinate system.

6. Guaranteeing feasibility

For some classes of trajectory, the existence of a feasible motion primitive can be guaranteed a priori, without running the tests described in the preceding section.

For the specific case of primitives starting and ending at rest (zero velocity, zero acceleration, and a given end point), a bound on the end time T will be explicitly calculated in dependence of the translation distance, such that any end time larger than this bound is guaranteed to be feasible with respect to the input constraints. Furthermore, the position trajectory for such rest-to-rest primitives is shown to remain on the straight line segment between the initial and final positions. Thus, given a convex allowable flight space, all

primitives that start and end at rest and within the allowable flight space will remain within the allowable flight space for the entire trajectory.

The input feasibility of general motion primitives, with arbitrary initial and final accelerations and velocities, is also briefly discussed. It should be noted that those motion primitives which can be guaranteed to be feasible a priori will be a conservative subset of the primitives which can be shown to be feasible a posteriori using the recursive tests described in Section 4. Further discussion is provided in Section 7.

6.1 Rest-to-rest primitives

First, an upper bound on the lowest end time T at which a rest-to-rest primitive becomes feasible with respect to the input constraints is calculated in dependence of the translation distance. Without loss of generality it is assumed that the quadrocopter's initial position coincides with the origin, and the problem is posed in a reference frame such that the final position is given as $(p_f, 0, 0)$, with p_f being the distance from the origin to the final location, such that $p_f \geq 0$.

From (1.25) it is clear that the optimal position trajectory along x_2 and x_3 has zero position for the duration of the primitive, and therefore also zero acceleration and velocity.

1) Input feasibility The acceleration trajectory along axis 1 is calculated by solving the motion coefficients with (1.25), and then substituting into (1.22), such that

$$\ddot{x}_1(t) = 60\frac{t}{T^3}p_f - 180\frac{t^2}{T^4}p_f + 120\frac{t^3}{T^5}p_f. \tag{1.43}$$

Introducing the variable $\xi \in [0, 1]$ such that $t = \xi T$, and substituting for the above yields

$$\ddot{x}_1(\xi T) = \frac{60p_f}{T^2}\left(\xi - 3\xi^2 + 2\xi^3\right) \tag{1.44}$$

for which the extrema lie at

$$\xi^* = \frac{1}{2} \pm \frac{\sqrt{3}}{6} \tag{1.45}$$

$$\ddot{x}_1(\xi^* T) = \mp\frac{10\sqrt{3}p_f}{3T^2}. \tag{1.46}$$

Exploiting the observation that $|\ddot{x}(\xi^* T)|$ decreases monotonically with increasing T, an upper bound $T_{f_{\min}}$ for the end time at which such a primitive becomes feasible with respect to the minimum thrust constraint can be calculated. This is done by substituting (1.46) for the sufficient criterion of (1.32), under the worst case assumption that the motion is aligned with gravity, i.e. the acceleration in the directions perpendicular to

gravity are zero. Then

$$T_{f_{\min}} = \sqrt{\frac{10p_f}{\sqrt{3}\left(\|\boldsymbol{g}\| - f_{\min}\right)}}. \tag{1.47}$$

Similarly, an upper bound $T_{f_{\max}}$ for the end time at which the primitive becomes feasible with respect to the maximum thrust constraint can be calculated by substituting the maximum acceleration bound of (1.46) into the sufficient criterion of (1.31). Again, the worst case occurs when the motion is aligned with gravity, and the final time can be calculated as

$$T_{f_{\max}} = \sqrt{\frac{10p_f}{\sqrt{3}\left(f_{\max} - \|\boldsymbol{g}\|\right)}}. \tag{1.48}$$

Finally, an upper bound $T_{\omega_{\max}}$ for the end time at which the primitive satisfies the body rates constraint is calculated as follows. First, the jerk along axis 1 is solved for with (1.21), and $t = \xi T$ is substituted as before, to give

$$j_1(\xi T) = \frac{60p_f}{T^3}\left(6\xi^2 - 6\xi + 1\right). \tag{1.49}$$

This has extrema at $\xi \in \{0, \frac{1}{2}, 1\}$, and the maximum of $|j_1(\xi T)|$ occurs at $\xi^* \in \{0, 1\}$, so that

$$\max_{t\in[0,T]} |j_i(t)| = |j_i(\xi^* T)| = \frac{60p_f}{T^3}. \tag{1.50}$$

Again, this maximum is monotonically decreasing for increasing end time T.

This value is then substituted for the numerator of (1.40), and it is assumed that the primitive satisfies the minimum thrust constraint, so that $f_{\min}$ can be substituted for the denominator. This makes the conservative assumption that the maximum jerk value is achieved at the same time as the minimum thrust value. Equating the result to $\omega_{\max}$, and solving for $T_{\omega_{\max}}$ yields

$$T_{\omega_{\max}} = \sqrt[3]{\frac{60p_f}{\omega_{\max} f_{\min}}}. \tag{1.51}$$

Note that this requires $f_{\min}$ to be strictly positive (instead of non-negative as specified in (1.4)).

Combining (1.47), (1.48), and (1.51), any rest-to-rest primitive within a ball of radius p_f is guaranteed to be feasible with respect to the input constraints of Section 1 if

the end time T is chosen to satisfy the below bound.

$$T \geq \max\{T_{f_{\min}}, T_{f_{\max}}, T_{\omega_{\max}}\}. \tag{1.52}$$

The conservatism of this bound is investigated in Section 7.

2) Position feasibility It will be shown that the position along a rest-to-rest primitive remains on the straight line segment between the initial and final positions, independent of the end time T. Solving the full position trajectory as given in Section 1 and substituting $p_0 = 0$, $v_0 = v_f = 0$ and $a_0 = a_f = 0$, the position trajectory in axis 1 is given by

$$x_1(t) = p_f \left(6\frac{t^5}{T^5} - 15\frac{t^4}{T^4} + 10\frac{t^3}{T^3} \right) \tag{1.53}$$

with $p_f \geq 0$ the desired displacement along axis 1. Substituting again for $\xi = t/T$ such that $\xi \in [0, 1]$, the position trajectory can be rewritten as

$$x_1(\xi T) = p_f \left(6\xi^5 - 15\xi^4 + 10\xi^3 \right) \tag{1.54}$$

It is now straight forward to show that the extrema of $x_1(\xi T)$ are at $\xi^* \in \{0, 1\}$, and specifically that

$$x_1(t) \in [0, p_f]\,. \tag{1.55}$$

Axes 2 and 3 will remain at zero, so that the rest-to-rest primitive will travel along the straight line segment connecting the initial and final positions in three dimensional space. Therefore, given a convex allowable flight space, if the initial and final position are in the allowable flight space, all rest-to-rest primitives will remain within the allowable flight space.

3) Maximum velocity In some applications it is desirable to limit the maximum velocity of the vehicle along the motion, most notably where the quadrocopter's pose is estimated with onboard vision [1.27]. For rest-to-rest primitives of duration T, the maximum speed occurs at $t = T/2$, and equals

$$\max_{t\in[0,T]} \|\dot{\boldsymbol{x}}(t)\| = \max_{t\in[0,T]} |\dot{x}_1(t)| = \frac{15}{8}\,\frac{p_f}{T}. \tag{1.56}$$

Thus, given some maximum allowable velocity magnitude and a distance to translate, the primitive end time T at which this maximum velocity will not be exceeded can be readily calculated from the above.

6.2 Guarantees for general primitives

For general primitives, with nonzero initial and/or final conditions, and with possibly unconstrained end states, it is harder to provide conditions under which feasible primitives can be guaranteed. Indeed, cases can be constructed which are input feasible for some specific end times, but become infeasible if the time is extended. It can however be stated for a motion primitive along an axis k (with arbitrary initial and final conditions and with any combination of final translational variable constraints in the goal state) that as the end time T tends to infinity:

- the magnitude of the jerk trajectory tends to zero, and
- the magnitude of the acceleration trajectory becomes independent of both initial and final position and velocity constraints, and can be bounded as

$$\lim_{T\to\infty} \max_{t\in[0,T]} \ddot{x}_k(t) \leq \max\{|a_{k0}|, |a_{kf}|\} \tag{1.57}$$

$$\lim_{T\to\infty} \min_{t\in[0,T]} \ddot{x}_k(t) \geq -\max\{|a_{k0}|, |a_{kf}|\}. \tag{1.58}$$

If the final acceleration is not specified, the acceleration bounds are

$$\lim_{T\to\infty} \max_{t\in[0,T]} \ddot{x}_k(t) \leq |a_{k0}| \tag{1.59}$$

$$\lim_{T\to\infty} \min_{t\in[0,T]} \ddot{x}_k(t) \geq -|a_{k0}|. \tag{1.60}$$

The calculations to show this can be found in Appendix B.

This knowledge can then be combined with the input feasibility constraints (similar to the rest-to-rest primitives, above), to guarantee the existence of an input feasible motion primitive based on the values of $|a_0|$ and $|a_f|$, at least as the end time is extended to infinity. Furthermore, by expanding the acceleration trajectory from (1.22) and applying the triangle inequality, the magnitude of the acceleration for finite end times can also be bounded. This bound can then be used to calculate an upper bound on the end time T at which the primitive will be feasible with respect to the inputs, however this bound will typically be very conservative.

Algorithm overview

The focus in the three preceding sections is on the generation and feasibility verification for a quadrocopter motion primitive, with an arbitrary initial state, to a set of goal end translational variables $\hat{\sigma}_i$ in a given time T. The resulting acceleration trajectory, along

any axis, is a cubic polynomial in time. These trajectories minimize an upper bound representative of a product of the inputs, and computationally convenient methods are presented to test whether these trajectories are feasible with respect to input constraints, and with respect to bounds on linear combinations of the translational variables. Guarantees on feasible trajectories are given, with a specific focus on rest to rest trajectories.

In the following section the set end times and goal end translational variables for which feasible trajectories can be found with the presented approach is compared to the total set of feasible trajectories, for the class of rest to rest trajectories. The computational cost of the presented approach is investigated in Section 8. Section 9 describes an experimental demonstration, where the presented primitives are incorporated into a larger trajectory generator.

7. Conservatism

If the motion primitive computed with the presented approach is not feasible, it does not imply that no feasible trajectory is possible. There are two reasons for this:

- the trajectories generated in Section 3 are restricted to have accelerations described by cubic polynomials in time, and
- the feasibility verification of Section 4 is sufficient, but not necessary.

This section attempts to give an indication of the space of end times T and end translational variables $\hat{\sigma}_i$ for which a quadrocopter could fly a trajectory, but the presented method is unable to find a feasible motion primitive. This section will specifically consider rest-to-rest motions.

In [1.28] an algorithm is given to compute minimum time trajectories which satisfy Pontryagin's minimum principle, for the quadrocopter system dynamics and input constraints of Section 2.1. These trajectories represent the surface of the feasible region for quadrocopter rest-to-rest trajectories: given a desired final translation, a feasible trajectory exists with any end time larger than the time optimal end time (e.g. by executing the time optimal trajectory, and then simply waiting). By definition, no feasible trajectory exists for shorter end times.

Fig. 1.5 compares this feasible region to those trajectories that can be found with the methods of Section 3 and Section 4. The system limits were set as in [1.28], with $f_{\min} = 1\,\mathrm{m/s^2}$, $f_{\max} = 20\,\mathrm{m/s^2}$, $\omega_{\max} = 10\,\mathrm{rad/s}$. For a given translation distance d, the desired end translational variables are defined such that all components are zero, except the position in the direction of motion which is set to d.

For each distance d, the space of feasible end times was determined as follows. A set of end times was generated, starting at zero and with increments of 1 ms. For each end time, a motion primitive was generated, and the set of end distances and end times (T, d)

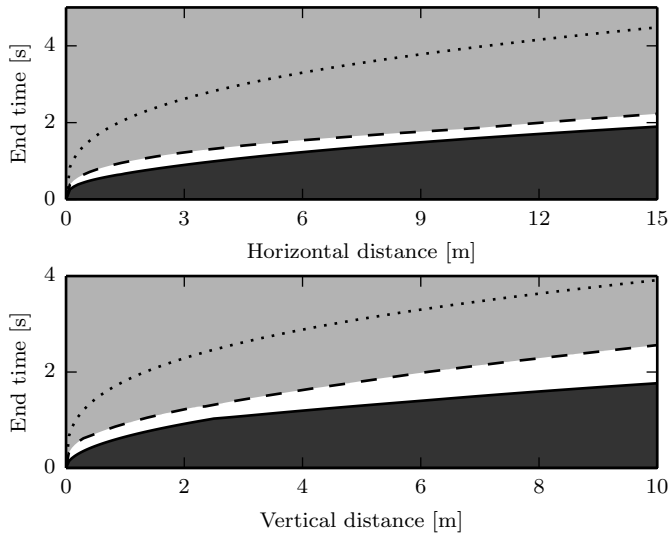

Figure 1.5. The set of horizontal/vertical final displacements for which the presented algorithm is able to find input feasible trajectories (the lightly shaded area above the dashed line), and the set of final displacements and end times for which no trajectories are possible (the darkly shaded area, with the boundary as given by the time optimal trajectories of [1.28]). The dotted line is the conservative end time guarantee as calculated in Section 6.1. The white area represents displacements that could be reached by the vehicle, but where the presented method cannot find a feasible motion primitive.

for which an input feasible trajectory was found is shown in Fig. 1.5. For the sake of comparison, the guaranteed feasible end time given in Section 6.1 is also plotted.

The fastest feasible manoeuvre found with the presented algorithm requires on the order of 50% longer than the time optimal trajectory of [1.28] when translating vertically, and on the order of 20% longer when translating horizontally. From the figure it can be seen that the guaranteed feasible end time of Section 6 is quite conservative, requiring the order of three times longer for the manoeuvre than the time optimal trajectory. However, these trajectories can be computed at very low cost, specifically requiring no iterations to determine feasibility.

8. Computation times

This section presents statistics for the computational cost of the presented algorithm when implemented on a standard laptop computer, and on a standard microcontroller. The algorithm was implemented in C++. Except for setting the compiler optimisation to maximum, no systematic attempt was made to optimise the code for speed. To evaluate the time required to compute the motion primitives described in this paper, primitives were generated for a quadrocopter starting at rest, at the origin, and translating to a final position chosen uniformly at random in a box with side length 4 m, centred at the origin. The target end velocity and acceleration were also chosen uniformly between $-2\,\mathrm{m/s}$

and 2 m/s, and $-2\,\mathrm{m/s^2}$ and $2\,\mathrm{m/s^2}$, respectively. The end time was chosen uniformly at random between 0.2 s and 10 s. The algorithm was implemented on a laptop computer with an Intel Core i7-3720QM CPU running at 2.6GHz, with 8GB of RAM, with the solver compiled with Microsoft Visual C++ 11.0. The solver was run as a single thread.

For one hundred million such motion primitives, the average computation time per primitive was measured to be 1.05 µs, or approximately 950'000 primitives per second. This includes

- generating the primitive,
- verifying that the inputs along the trajectory are feasible (with the recursive test of Section 3, for $f_{min} = 5\,\mathrm{m/s^2}$, $f_{max} = 25\,\mathrm{m/s^2}$, and $\omega_{max} = 20\,\mathrm{rad/s}$, and
- verifying that the position of the quadrocopter stays within a $4 \times 4 \times 4$m box centred at the origin (that is, six affine constraints of the form (1.8) as described in Section 4.2).

Of the primitives, 91.6% tested feasible with respect to the inputs, 6.4% infeasible, and the remaining 2.0% were indeterminate.

If the position feasibility test is not performed, the average computation time drops to 0.74 µs, again averaged over the generation of one hundred million random primitives, or approximately 1.3 million per second.

The algorithm was also implemented on the STM32-F4 microcontroller, which costs approximately EUR 10 and is, for example, used on the open-source PX4 autopilot system [1.29]. One million random primitives were generated, with the same parameters as above. The average execution time was 149 µs per primitive (or approximately 6'700 primitives per second), including trajectory generation, input feasibility, and position feasibility tests. If the position feasibility test is not performed, the average computation time drops to 95 µs per primitive, or approximately 10'500 per second.

9. Example application and experimental results

This section presents an example of a high level trajectory generator that uses the motion primitives to achieve a high level goal (as illustrated in Fig. 1.1). The high-level goal is for a quadrocopter, with an attached net, to catch a ball thrown by a person. This task was chosen due to its highly dynamic three-dimensional nature, the requirement for real-time trajectory generation, and the existence of a variety of trajectories which would achieve the goal of catching the ball. All the presented data is from actual flight experiments.

The algorithm is applied in a naïve brute force approach, to illustrate the ease with which a complex, highly dynamic, quadrocopter task may be encoded and solved. The multimedia attachment contains a video showing the quadrocopter catching the ball.

To catch the ball, the motion primitive generator is used in two different ways:

- to generate trajectories to a catching point, starting from the quadrocopter's current state and ending at a state and time at which the ball enters the attached net, and
- to generate stopping trajectories, which are used to bring the quadrocopter to rest after the ball enters the net.

The experiments were conducted in the Flying Machine Arena, a space equipped with an overhead motion capture system which tracks the pose of the quadrocopters and the position of the balls. A PC processes the motion capture data, and generates commands that are transmitted wirelessly to the vehicles at 50 Hz. More information on the system can be found in [1.30]. The quadrocopter used is a modified Ascending Technologies "Hummingbird" [1.31], with a net attached approximately 18 cm above the vehicle's centre of mass (see Fig. 1.6).

1) Catching trajectories The computational speed of the presented approach is exploited to evaluate many different ways of catching the ball. This is done with a naïve brute force approach, where thousands of different primitives are generated at every controller step. Each primitive encodes a different strategy to catch the ball, of which infeasible primitives are rejected and the 'best' is kept from the remainder. This is done in real-time, and used as an implicit feedback law, with one controller cycle typically involving the evaluation of thousands of candidate motion primitives. This task is related to that of hitting a ball with a racket attached to a quadrocopter, as was previously investigated in [1.32] and [1.21].

The catching task is encoded in the format of Section 2.2 by stipulating that the centre of the net must be at the same position as the ball, and the velocity of the quadrocopter perpendicular to the ball's flight must be zero. The requirement on the velocity reduces the impact of timing errors on the system. Because the centre of the net is not at the centre of the quadrocopter, the goal end state must also include the quadrocopter's normal

Figure 1.6. A quadrocopter with attached catching net, as used to demonstrate the algorithm. The centre of the net is mounted above the vehicle's centre of mass in the quadrocopter's $\boldsymbol{e}_3$ direction.

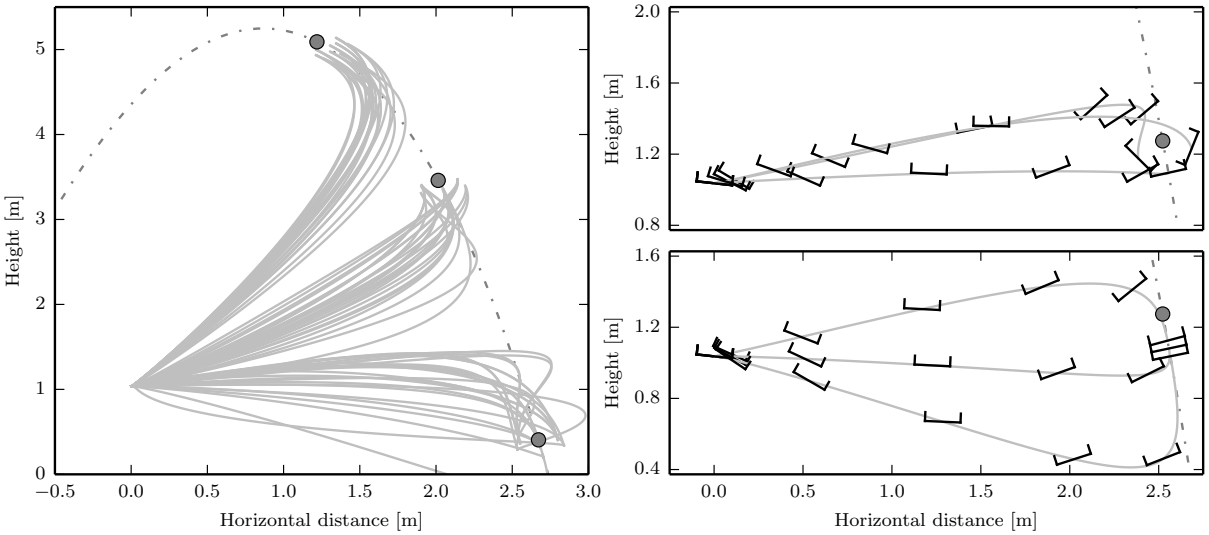

Figure 1.7. Sampled motion primitives at one time step of a catching manoeuvre: On the left is shown the quadrocopter's centre of mass for 81 candidate primitives to catch a thrown ball. The primitives are shown as solid lines, starting at the position $(0, 1)$m, and the ball's predicted trajectory is shown as a dash-dotted line, moving from left to right in the figure. The ball's position at each of three candidate catching instants is shown as a solid circle. Each candidate primitive places the centre of the net at the ball, with the quadrocopter's velocity at that instant parallel to the ball's velocity. Note that the final orientation is a parameter that is searched over, as is the thrust value at the catching instant. A total of 2'812 candidates were evaluated at this time instant, which required 3.05 ms of computation. The primitives shown passing through the ground (at 0 m height) are eliminated when the position boundaries are evaluated. The right most two plots show detail of some of these primitives, showing the quadrocopter's orientation along the primitives. At the top-right plot, three candidate primitives are shown with different end orientations (and only showing orientations lying in the plane of the plot). The lower right plot shows primitives to the same orientation, but with varying end thrusts.

direction $\boldsymbol{e}_3$ – given some ball position, there exists a family of quadrocopter positions and normal directions which will place the centre of the net at the ball's position. The desired end translational variables $\hat{\sigma}$ thus contains the quadrocopter's position, its normal direction (thus acceleration), and its velocity components perpendicular to the ball's velocity (thus specifying eight of the nine possible variables). The strategy for specifying these eight variables is described below.

The ball is modelled as a point mass with quadratic drag, and at every controller update step its trajectory is predicted until it hits the floor. This is discretized to generate a set of possible catching times $T^{(k)}$, with the discretization time step size chosen such that at most 20 end times are generated, and that the time step size is no smaller than the controller update period.

For each of these possible catching times $T^{(k)}$, a set of candidate desired end translational variables $\hat{\sigma}^{(j,k)}$ is generated as follows. The quadrocopter's goal normal direction is generated by generating 49 candidate end normals, centred around the direction opposing the ball's velocity at time $T^{(k)}$. To convert these candidate end normals to goal accelerations it is necessary to further specify an end thrust value for each. The goal acceleration

can then be calculated with (1.1). For each of the orientations generated, ten candidate final thrust values are used, spaced uniformly between $f_{\min}$ and $f_{\max}$.

Given an end normal direction, the required quadrocopter position at the catching instant can be calculated as that position placing the centre of the net at the ball (for the 49 different normal direction candidates, the end location of the vehicle centre of mass will be located on a sphere centred at the ball's position).

The quadrocopter's velocity is required to be zero in the directions perpendicular to the ball's velocity at the catching time, while the quadrocopter velocity component parallel to the ball's velocity is left free. Thus eight components of the end translational variables in (1.7) are specified.

For each of the candidate catching instants, there are 490 candidate end states to be evaluated. Because the number of end times is limited to 20, this means that there are at most 9'800 catching primitives of the form $\left(T^{(k)}, \hat{\sigma}^{(j,k)}\right)$ calculated at any controller update step.

Next the candidate primitive is tested for feasibility with respect to the inputs as described in Section 4.1, with the input limits set to $f_{min} = 5\,\mathrm{m/s^2}$, $f_{max} = 25\,\mathrm{m/s^2}$ and $\omega_{max} = 20\,\mathrm{rad/s}$. Then the candidate is tested for position feasibility, where the position trajectory is verified to remain inside a safe box as described in Section 4.2 – this is to remove trajectories that would either collide with the floor or the walls. If the primitive fails either of these tests, it is rejected.

Some such candidate motion primitives are visualised in Fig. 1.7.

For each candidate catching primitive remaining, a stopping trajectory will be searched for (described in more detail below). If no stopping trajectory for a candidate catching primitive is found that satisfies both the input feasibility constraints and position constraints, that catching candidate is removed from the set.

Now, each remaining candidate is feasible with respect to input and position constraints, both for catching the ball, and the stopping manoeuvre after the ball is caught. From this set, that candidate with the lowest cost J_Σ (as defined in Section 3) is selected as the best.

2) Stopping trajectories At the catching instant, a catching candidate trajectory will generally have a nonzero velocity and acceleration, making it necessary to generate trajectories from this state to rest. For these stopping trajectories, the goal end state translational variables specify that the velocity and acceleration must be zero, but leave the position unspecified. The primitive duration is sampled from a set of 6 possibilities, ranging from 2 down to 0.25 seconds. The search is terminated after the first stopping primitive is found that is feasible with respect to the inputs and the position constraints. Two such stopping primitives are shown in Fig. 1.8.

3) Closed loop control Each remaining candidate catching primitive is feasible with respect to the input constraints, the position box constraints, and has a feasible stopping primitive. From these, the catching candidate with the lowest cost value is then selected as

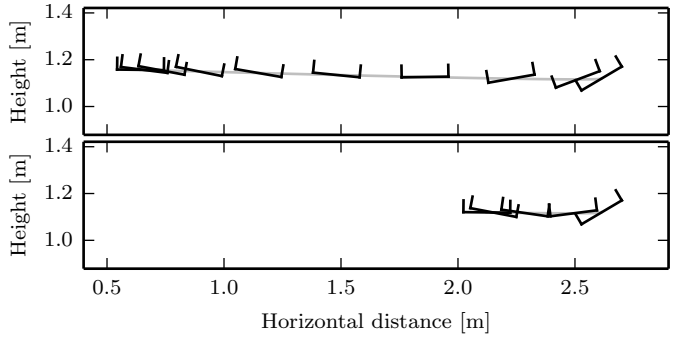

Figure 1.8. Sampled stopping motion primitives: Two candidate stopping primitives bringing the quadrocopter to rest, starting at the catching instant of the optimal catching primitive from Fig. 1.7. The quadrocopter moves from right to left in both figures. The upper stopping candidate brings the quadrocopter to rest in 2 s, the lower in 1 s.

the best. This algorithm is then applied as an implicit feedback law, as in model predictive control [1.33], such that the entire process is repeated at the controller frequency of 50 Hz – thus the high level trajectory generator must run in under 20 ms. This allows the system to implicitly update the trajectories as the prediction of the ball's flight becomes more precise, as well as compensate for disturbances acting on the quadrocopter.

If no candidate catching primitive remains, the last feasible trajectory is used as reference trajectory, tracked under feedback control. This typically happens at the end of the motion, as the end time goes to zero. After the ball is caught, the stopping primitive is executed. The stopping primitive is used as a reference trajectory tracked by the controller described in [1.30].

The completed catching trajectory corresponding to the candidates of Fig. 1.7 is shown in Fig. 1.9. The catching manoeuvre lasted 1.63 s, during which a total of 375'985 motion primitives were evaluated (including both catching and stopping manoeuvres). To catch the ball, the quadrocopter translated a distance of 2.93 m, having started at rest.

The attached video shows that the quadrocopter manages to catch thrown balls, validating the brute-force approach used to encode this problem. The video also shows the acrobatic nature of the resulting manoeuvres.

10. Conclusion

This paper presents a motion primitive that is computationally efficient both to generate, and to test for feasibility. The motion primitive starts at an arbitrary quadrocopter state, and generates a thrice differentiable position trajectory guiding the quadrocopter to a set of desired end translational variables (consisting of any combination of components of the vehicle's position, velocity, and acceleration). The acceleration allows to encode two components of the vehicle's attitude. The time to calculate a motion primitive and apply input and translational feasibility tests is shown to be on the order of a microsecond on

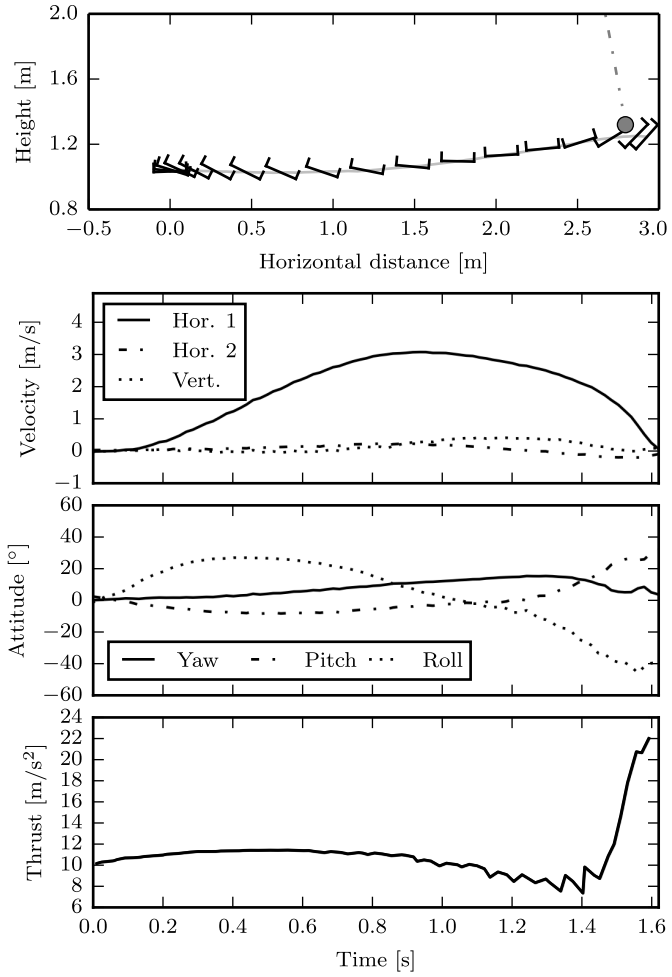

Figure 1.9. The actual trajectory flown to catch the ball shown in Fig. 1.7. The top-most plot shows the ball's flight shown as a dash-dotted line, and the ball's position at the catching instant shown as a solid circle. Note that the offset between the vehicle and the ball at the catching instant is due to the net's offset from the quadrocopter's centre of mass. The three lower plots show the manoeuvre history, until the catching instant, with (from top to bottom) the quadrocopter's velocity, attitude, and thrust commands. The attitude of the vehicle is shown using the conventional 3-2-1 Euler yaw-pitch-roll sequence [1.24]. The bottom plot shows the thrust command, which can be seen to be within the limits of 5 to 25 $\mathrm{m\,s^{-2}}$. It should be noted that the motion primitives are applied as an implicit feedback law, and thus the flown trajectory does not correspond to any single planned motion primitive.

a standard laptop computer.

The algorithm is experimentally demonstrated by catching a thrown ball with a quadrocopter, where it is used as part of an implicit feedback controller. In the application, thousands of candidate primitives are calculated and evaluated per controller update step, allowing the search over a large space of possible catching manoeuvres.

The algorithm appears well-suited to problems requiring to search over a large trajectory space, such as probabilistic planning algorithms [1.34], or the problem of planning for dynamic tasks with multiple vehicles in real time, similar to [1.35]. The algorithm

may be especially useful if the high-level goal is described by non-convex constraints.

The presented motion primitive is independent of the quadrocopter's rotation about it's thrust axis (the Euler yaw angle). This is reflected in that the resulting commands do not specify a yaw rate, ω_3. A useful extension would be to compute an input trajectory for the quadrocopter yaw rate to achieve a desired final yaw angle.

It is shown that constraints on affine combinations of the quadrocopter's position, velocity, and acceleration may be tested efficiently. An interesting extension would be to investigate efficient tests for alternative constraint sets, for example more general convex sets.

Implementations of the algorithm in both Python and C++ can be found at [1.19].

A. Solutions for different end states constraints

Here the solutions for each combination of fixed end state is given in closed form for one axis. The states can be recovered by evaluating (1.22), and the trajectory cost from (1.26). The values Δp, Δv and Δa are calculated by (1.24).

Fixed position, velocity, and acceleration

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T^5}\begin{bmatrix}720 & -360T & 60T^2\\ -360T & 168T^2 & -24T^3\\ 60T^2 & -24T^3 & 3T^4\end{bmatrix}\begin{bmatrix}\Delta p\\ \Delta v\\ \Delta a\end{bmatrix} \tag{1.61}$$

Fixed position and velocity

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T^5}\begin{bmatrix}320 & -120T\\ -200T & 72T^2\\ 40T^2 & -12T^3\end{bmatrix}\begin{bmatrix}\Delta p\\ \Delta v\end{bmatrix} \tag{1.62}$$

Fixed position and acceleration

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{2T^5}\begin{bmatrix}90 & -15T^2\\ -90T & 15T^3\\ 30T^2 & -3T^4\end{bmatrix}\begin{bmatrix}\Delta p\\ \Delta a\end{bmatrix} \tag{1.63}$$

Fixed velocity and acceleration

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T^3}\begin{bmatrix}0 & 0\\ -12 & 6T\\ 6T & -2T^2\end{bmatrix}\begin{bmatrix}\Delta v\\ \Delta a\end{bmatrix} \tag{1.64}$$

Fixed position

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T^5}\begin{bmatrix}20\\ -20T\\ 10T^2\end{bmatrix}\Delta p \tag{1.65}$$

Fixed velocity

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T^3}\begin{bmatrix}0\\ -3\\ 3T\end{bmatrix}\Delta v \tag{1.66}$$

Fixed acceleration

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \frac{1}{T}\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}\Delta a \tag{1.67}$$

B. Derivation of acceleration bounds

In Section 6.2 the claim is made that the acceleration can be bounded as the end time T tends to infinity. This is shown here for each of the different combination of end translational variable constraints. The constraints will be divided into those that constrain the final acceleration, and those that do not.

B.1 Constrained final acceleration

If the final acceleration is specified, it will be shown that the acceleration can be bounded as in (1.57)-(1.58).

1) Fixed position, velocity, and acceleration Substituting for the parameters α, β, and γ from (1.61) into (1.22), the acceleration trajectory for a fully specified set of end translational variables is

$$\begin{aligned}\ddot{x}(t) =& a_0 - \frac{9a_0}{T}t + \frac{3a_f}{T}t + \frac{18a_0}{T^2}t^2 - \frac{12a_f}{T^2}t^2 - \frac{36t}{T^2}v_0 - \frac{24t}{T^2}v_f - \frac{10a_0}{T^3}t^3\\ &+ \frac{10a_f}{T^3}t^3 - \frac{60p_0}{T^3}t + \frac{60p_f}{T^3}t + \frac{96v_0}{T^3}t^2 + \frac{84v_f}{T^3}t^2 + \frac{180p_0}{T^4}t^2 - \frac{180p_f}{T^4}t^2\\ &- \frac{60v_0}{T^4}t^3 - \frac{60v_f}{T^4}t^3 - \frac{120p_0}{T^5}t^3 + \frac{120p_f}{T^5}t^3.\end{aligned} \tag{1.68}$$

Introducing the variable $\xi := t/T \in [0, 1]$ and letting $T \to \infty$, yields

$$\lim_{T\to\infty} \ddot{x}(\xi T) = -10a_0\xi^3 + 18a_0\xi^2 - 9a_0\xi + a_0 + 10a_f\xi^3 - 12a_f\xi^2 + 3a_f\xi. \tag{1.69}$$

The above does not contain the initial and final position or velocity, as was claimed in Section 6.2. This means that in the limit as the duration T tends to infinity, the acceleration trajectory becomes independent of the position and velocity, for a fully defined end state. Next, it will be shown that (1.57) and (1.58) hold.

To do this, three cases will be examined independently:

- Case 1: $a_0 = a_f = 0$,
- Case 2: $|a_f| \leq |a_0|$, and
- Case 3: $|a_f| > |a_0|$.

For Case I, trivially, the right hand side of (1.69) goes to zero, such that

$$\lim_{T\to\infty} \ddot{x}(\xi T) = 0. \tag{1.70}$$

For Case II, we define the ratio $\rho_2 = a_f/a_0 \in [-1, 1]$. It will be shown that the ratio between the maximum acceleration along the trajectory and the initial acceleration a_0 is in the range $[-1, 1]$. Substituting into (1.69), yields the acceleration ratio as a function $\phi_2(\xi, \rho_2)$ of two variables:

$$\begin{aligned}\phi_2(\xi, \rho_2) :=& \lim_{T\to\infty} \frac{\ddot{x}(\xi T)|_{a_f=\rho_2 a_0}}{a_0} \\ =& \xi^3 (10\rho_1 - 10) + \xi^2 (-12\rho_1 + 18) + 3\xi (\rho_1 - 3) + 1\end{aligned} \tag{1.71}$$

The extrema of ϕ_2 over $\xi \in [0, 1]$ and $\rho_2 \in [-1, 1]$ can be calculated straight-forwardly, and the minimum and maximum value of ϕ_2 calculated. From this can be calculated

$$\phi_2(\xi, \rho_2) \in [-1, 1] \tag{1.72}$$

from which then follows that $\forall \xi \in [0, 1], |a_f| \leq |a_0|$:

$$\lim_{T\to\infty} |\ddot{x}(\xi T)| \leq \max\{|a_0|, |a_f|\}. \tag{1.73}$$

For Case III, we define the ratio $\rho_3 = a_0/a_f \in (-1, 1)$. It will be shown that the ratio between the acceleration extrema along the trajectory and the acceleration a_f is in the range $[-1, 1]$.

Substituting into (1.69), again yields the acceleration ratio as a function of two vari-

ables $\phi_3(\xi, \rho_3)$:

$$\begin{aligned}\phi_3(\xi, \rho_3) :=& \lim_{T\to\infty} \frac{\ddot{x}(\xi T)|_{a_0=\rho_3 a_f}}{a_f} \\ =& \rho_2 + \xi^3(-10\rho_2 + 10) + \xi^2(18\rho_2 - 12) - 3\xi(3\rho_2 - 1)\end{aligned} \tag{1.74}$$

Similar to case 2 before, the extrema of ϕ_3 over $\xi \in [0, 1]$ and $\rho_3 \in [-1, 1]$ can be calculated. Note that the closed interval is used for ρ_3, for convenience. Then

$$\phi_3(\xi, \rho_3) \in [-1, 1] \tag{1.75}$$

from which follows that $\forall \xi \in [0, 1], |a_f| > |a_0|$:

$$\lim_{T\to\infty} |\ddot{x}(\xi T)| \leq \max\{|a_0|, |a_f|\}. \tag{1.76}$$

2) Fixed velocity and acceleration If only the velocity and acceleration are fixed, the same procedure can be used as above, specifically evaluating the same three cases. Analogously to (1.68) the acceleration trajectory is

$$\ddot{x}(t) = a_0 - \frac{4a_0}{T}t - \frac{2a_f}{T}t + \frac{3a_0}{T^2}t^2 + \frac{3a_f}{T^2}t^2 - \frac{6t}{T^2}v_0 + \frac{6t}{T^2}v_f + \frac{6v_0}{T^3}t^2 - \frac{6v_f}{T^3}t^2. \tag{1.77}$$

Analogously to (1.69) the limit can be taken, and the variable ξ introduced:

$$\lim_{T\to\infty} \ddot{x}(\xi T) = 3a_0\xi^2 - 4a_0\xi + a_0 + 3a_f\xi^2 - 2a_f\xi \tag{1.78}$$

Applying the same three cases as before, (1.57)-(1.58) follow.

3) Fixed acceleration If only the end acceleration is specified, the acceleration trajectory is

$$\ddot{x}(t) = a_0 - \frac{a_0 t}{T} + \frac{a_f t}{T}. \tag{1.79}$$

From this, and noting that $t/T \in [0, 1]$, (1.57)-(1.58) follow trivially.

B.2 Unconstrained final acceleration

If the final acceleration is not fixed, the procedure of Section 1 must be modified slightly.

1) Fixed position and velocity Analogously to (1.69) the acceleration in this case is

$$\begin{aligned}\ddot{x}(t) =& a_0 - \frac{8a_0}{T}t + \frac{14a_0}{T^2}t^2 - \frac{28t}{T^2}v_0 - \frac{12t}{T^2}v_f - \frac{20a_0t^3}{3T^3} \\ &- \frac{40p_0}{T^3}t + \frac{40p_f}{T^3}t + \frac{64v_0}{T^3}t^2 + \frac{36v_f}{T^3}t^2 + \frac{100p_0}{T^4}t^2 \\ &- \frac{100p_f}{T^4}t^2 - \frac{100t^3v_0}{3T^4} - \frac{20v_f}{T^4}t^3 - \frac{160p_0t^3}{3T^5} + \frac{160p_ft^3}{3T^5}.\end{aligned} \tag{1.80}$$

Introducing again the variable $\xi = t/T \in [0, 1]$ and letting $T \to \infty$, yields

$$\lim_{T\to\infty} \ddot{x}(\xi t) = -\frac{20a_0}{3}\xi^3 + 14a_0\xi^2 - 8a_0\xi + a_0 \tag{1.81}$$

For $a_0 = 0$, the right hand side is trivially zero. For $a_0 \neq 0$, the above can be refactored, and for $\xi \in [0, 1]$:

$$\lim_{T\to\infty} \frac{\ddot{x}(\xi T)}{a_0} = -\frac{20}{3}\xi^3 + 14\xi^2 - 8\xi + 1 \in \left[-\frac{29}{75}, 1\right] \tag{1.82}$$

From this follow (1.59)-(1.60).

2) Fixed position The acceleration trajectory in this case is

$$\begin{aligned}\ddot{x}(t) =& a_0 - \frac{5a_0}{T}t + \frac{5a_0}{T^2}t^2 - \frac{10t}{T^2}v_0 - \frac{5a_0t^3}{3T^3} - \frac{10p_0}{T^3}t + \frac{10p_f}{T^3}t \\ &+ \frac{10v_0}{T^3}t^2 + \frac{10p_0}{T^4}t^2 - \frac{10p_f}{T^4}t^2 - \frac{10t^3v_0}{3T^4} - \frac{10p_0t^3}{3T^5} + \frac{10p_ft^3}{3T^5}.\end{aligned} \tag{1.83}$$

Introducing again the variable $\xi = t/T \in [0, 1]$ and letting $T \to \infty$, yields

$$\lim_{T\to\infty} \ddot{x}(\xi T) = -\frac{5a_0}{3}\xi^3 + 5a_0\xi^2 - 5a_0\xi + a_0 \tag{1.84}$$

For $a_0 = 0$, the right hand side is trivially zero. For $a_0 \neq 0$, the above can be refactored, and for $\xi \in [0, 1]$:

$$\lim_{T\to\infty} \frac{\ddot{x}(\xi T)}{a_0} = -\frac{5}{3}\xi^3 + 5\xi^2 - 5\xi + 1 \in \left[-\frac{2}{3}, 1\right]. \tag{1.85}$$

From this follow (1.59)-(1.60).

3) Fixed velocity The acceleration trajectory in this case is

$$\ddot{x}(t) = a_0 - \frac{3a_0}{T}t + \frac{3a_0t^2}{2T^2} - \frac{3t}{T^2}v_0 + \frac{3t}{T^2}v_f + \frac{3t^2v_0}{2T^3} - \frac{3t^2v_f}{2T^3}. \tag{1.86}$$

Introducing again the variable $\xi = t/T \in [0, 1]$ and letting $T \to \infty$, yields

$$\lim_{T\to\infty} \ddot{x}(\xi T) = \frac{3a_0}{2}\xi^2 - 3a_0\xi + a_0. \tag{1.87}$$

For $a_0 = 0$, the right hand side is trivially zero. For $a_0 \neq 0$, the above can be refactored, and for $\xi \in [0, 1]$:

$$\lim_{T\to\infty} \frac{\ddot{x}(\xi T)}{a_0} = \frac{3}{2}\xi^2 - 3\xi + 1 \in \left[-\frac{1}{2}, 1\right]. \tag{1.88}$$

From this follow (1.59)-(1.60).

Acknowledgement

The Flying Machine Arena is the result of contributions of many people, a full list of which can be found at `http://flyingmachinearena.org/`.

This research was supported by the Swiss National Science Foundation (SNSF).

References

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

[1.2] R. Ritz, M. W. Mueller, M. Hehn, and R. D'Andrea, "Cooperative quadrocopter ball throwing and catching", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2012, pp. 4972–4978.

[1.3] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment", in *AIAA Guidance, Navigation, and Control Conference*, 2007, pp. 1–20.

[1.4] S. Grzonka, G. Grisetti, and W. Burgard, "A fully autonomous indoor quadrotor", *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90–100, 2012.

[1.5] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for quadrotor flight", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[1.6] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter trajectory tracking control", in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, USA, Aug. 2008, pp. 1–14.

[1.7] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "A prototype of an autonomous controller for a quadrotor UAV", in *European Control Conference (ECC)*, Kos, Greece, 2007, pp. 1–8.

[1.8] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter", in *Mediterranean Conference on Control and Automation*, Jun. 2008, pp. 1258–1263.

[1.9] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors", in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011, pp. 2520–2525.

[1.10] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control", in *IFAC World Congress*, vol. 18, 2011, pp. 1485–1491.

[1.11] M. P. Vitus, W. Zhang, and C. J. Tomlin, "A hierarchical method for stochastic motion planning in uncertain environments", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Villamoura, Portugal, 2012, pp. 2263–2268.

[1.12] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "Direct method based control system for an autonomous quadrotor", *Journal of Intelligent & Robotic Systems*, vol. 60, no. 2, pp. 285–316, 2010.

[1.13] J. Wang, T. Bierling, L. Höcht, F. Holzapfel, S. Klose, and A. Knoll, "Novel dynamic inversion architecture design for quadrocopter control", in *Advances in Aerospace Guidance, Navigation and Control*, Springer, 2011, pp. 261–272.

[1.14] M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Inversion based direct position control and trajectory following for micro aerial vehicles", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2013, pp. 2933–2939.

[1.15] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of nonlinear systems: Introductory theory and examples", *International journal of control*, vol. 61, no. 6, pp. 1327–1361, 1995.

[1.16] R. M. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems: A catalog of prototype systems", in *ASME International Mechanical Engineering Congress and Exposition*, 1995.

[1.17] N. Faiz, S. Agrawal, and R. Murray, "Differentially flat systems with inequality constraints: An approach to real-time feasible trajectory generation", *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 2, pp. 219–227, 2001.

[1.18] C. Louembet, F. Cazaurang, and A. Zolghadri, "Motion planning for flat systems using positive b-splines: An lmi approach", *Automatica*, vol. 46, no. 8, pp. 1305–1309, 2010.

[1.19] M. W. Mueller, "Quadrocopter trajectory generator", *(online)* `github.com/markwmuller/RapidQuadrocopterTrajectories`, 2015.

[1.20] M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception", in *European Control Conference*, 2013, pp. 1383–1389.

[1.21] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3480–3486.

[1.22] P. Martin and E. Salaün, "The true role of accelerometer feedback in quadrotor control", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 1623–1629.

[1.23] R. Mahony, V. Kumar, and P. Corke, "Aerial vehicles: Modeling, estimation, and control of quadrotor", *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[1.24] P. H. Zipfel, *Modeling and Simulation of Aerospace Vehicle Dynamics Second Edition*. AIAA, 2007.

[1.25] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. I*. Athena Scientific, 2005.

[1.26] P. Borwein and T. Erdélyi, *Polynomials and Polynomial Inequalities*, ser. Graduate Texts in Mathematics Series. Springer, 1995.

[1.27] M. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "Path planning for motion dependent state estimation on micro aerial vehicles", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 3926–3932.

[1.28] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control", *Autonomous Robots*, vol. 33, pp. 69–88, 1 2012.

[1.29] L. Meier, P. Tanskanen, L. Heng, G. Lee, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision", English, *Autonomous Robots*, vol. 33, no. 1-2, pp. 21–39, 2012.

[1.30] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena", *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[1.31] D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz", in *IEEE International Conference on Robotics and Automation (ICRA)*, Apr. 2007, pp. 361–366.

[1.32] M. W. Mueller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 5113–5120.

[1.33] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice–a survey", *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[1.34] E. Frazzoli, M. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles", *Journal of Guidance Control and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002.

[1.35] M. Sherback, O. Purwin, and R. D'Andrea, "Real-time motion planning and control in the 2005 cornell robocup system", English, in *Robot Motion and Control*, ser. Lecture Notes in Control and Information Sciences, K. Kozlowski, Ed., vol. 335, Springer London, 2006, pp. 245–263.

Paper P2

A model predictive controller for quadrocopter state interception

Mark W. Mueller and Raffaello D'Andrea

Abstract

This paper presents a method for generating quadrocopter trajectories in real time, from some initial state to a final state defined by position, velocity and acceleration in a specified amount of time. The end state captures the attitude to within a rotation about the thrust axis. Trajectory generation is done by formulating the trajectory of the quadrocopter in its jerk, in discrete time, and then solving a convex optimisation problem on each decoupled axis. Convex bounds are derived to include feasibility constraints with respect to the quadrocopter's total allowable thrust and angular rates.

Published in *Proceedings of the 2013 European Control Conference.*

1. Introduction

Quadrocopters are an active area of research, owing to their agility, mechanical simplicity and robustness. They are capable of diverse tasks, and have been used as platforms to study vision-based pose estimation [2.1], nonlinear control [2.2], and learning [2.3], for example. Furthermore, they are useful as tools for solving practical problems such as surveillance [2.4] and inspection [2.5].

Various trajectory generation strategies have been proposed, depending on the goal to be achieved. Trajectory generation is complicated by the underactuated and nonlinear nature of the quadrocopter dynamics, and by difficult-to-model aerodynamics (see, e.g. [2.6], [2.7] for discussions on aerodynamic effects and [2.3] for a learning-based compensation strategy).

In [2.8] a strategy for generating state interception trajectories is presented, where the trajectory is broken down into a sequence of five phases, each part of which is assigned a different controller. These phases consist of: precise hover; 3D path following; attitude control to desired attitude; attitude recovery to zero angles; and finally, soft hover control for recovery.

The strategy presented in [2.9] involves exploiting the differential flatness of the quadrocopter dynamics, and is used to generate trajectories between so-called "keyframes", which are defined as positions in space and yaw angles. The generated trajectories then pass through keyframes at specified times, while minimizing the snap (fourth derivative of position) squared. These trajectories are solved using either a normalised time, or distance, and are then scaled to the problem at hand.

Pontryagin's minimum principle is exploited in [2.10] and [2.11] to generate: 1) bang-singular to-rest quadrocopter trajectories, and 2) bang-singular position interception trajectories, where the goal is to reach a given position at a given time, while minimizing the time required to stop after the intercept. These methods are computationally fast, with solution times on the order of microseconds.

The differential flatness of the quadrocopter dynamics was also exploited in [2.2] to generate trajectories, where the authors make the simplification that the roll-pitch-yaw Euler angle accelerations are control inputs. The optimisation is done on a weighted sum of the fuel cost (approximated as average speed) and deviation from desired arrival time. Trajectories are then solved for by choosing accelerations as polynomials in time, with the degree of the polynomial being a function of the number of boundary conditions, and then solving for the polynomial coefficients.

In [2.12] collision-free trajectories are generated to guide a fleet of UAVs from initial states to final states, guaranteeing that the trajectories maintain a minimum distance whilst minimising the total thrust produced by the quadrocopters. The solutions are found using sequential convex programming, with solution times on the order of seconds.

A learning-based model predictive controller (MPC) is presented in [2.13], with the Euler angles taken as inputs; a cascaded MPC is designed for a quadrocopter in [2.14], with the dynamics of the quadrocopter captured by piecewise affine equations, separating

control of the attitude and planar motions; while MPC is combined with robust control in [2.15]. In each case, MPC is used to track a given state trajectory.

This paper builds on a previously presented scheme for generating trajectories for generating trajectories for a quadrocopter hitting a ball with an attached racket [2.16]. That method required that the vehicle maintain small pitch and roll angles, with the end state restricted to the position, one component of the velocity, and the direction of the racket normal.

Here, a scheme is presented for generating state interception trajectories for quadrocopters; that is, trajectories starting from an arbitrary state and achieving a (reduced) end state in a specified amount of time, whilst satisfying input constraints. The desired end state is specified as the position, velocity and acceleration of the vehicle, i.e. fixing the attitude of the vehicle to within a rotation about the vehicle's thrust axis. The total thrust and body rates are bounded by convex functions, allowing the problem to be written as a convex constrained optimisation problem, which can be solved in real time on a typical desktop computer, and which allows the trajectory generator to be used as a diminishing horizon model predictive controller.

This method extends the state of the art by calculating state interception trajectories in real time, using sophisticated optimisation techniques to explicitly include input constraints in the trajectory generation problem.

The quadrocopter model is presented in Section 2, with the trajectory generation scheme given in Section 3. Section 4 discusses implementation and experimental results, and an outlook is given in Section 5.

2. Dynamic model

The quadrocopter is modelled as a rigid body with six degrees of freedom: linear translation along the inertial x_1, x_2 and x_3 axes, and three degrees of freedom describing the rotation of the frame attached to the body with respect to the inertial frame, which is taken here to be the proper orthogonal matrix $\mathbf{R}$. The control inputs to the system are taken as the total thrust produced f, for simplicity normalised by the vehicle mass and thus having units of acceleration; and the body rates expressed in the body-fixed frame as $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)$. These are illustrated in Fig. 2.1.

The mixing of these inputs to individual motor thrust commands is done on board the vehicle, using feedback from gyroscopes. It is assumed that the time constant of the onboard controllers is low enough to have negligible influence on the algorithm presented here. Because of their low rotational inertia, quadrocopters can achieve extremely high rotational accelerations (on the order of $200\,\mathrm{rad\,s^{-2}}$ [2.10]) about the ω_1 and ω_2 axes, while it will be shown that the rotation about ω_3 is not needed for the trajectories considered here. For example, [2.17] presents a scheme for mixing these inputs to motor commands.

The differential equations governing the flight of the quadrocopter are now taken as

those of a rigid body [2.18]

$$\ddot{\boldsymbol{x}} = \mathbf{R}\, \boldsymbol{e}_3 f + \boldsymbol{g} \tag{2.1}$$

$$\dot{\mathbf{R}} = \mathbf{R}\, [\![\boldsymbol{\omega}\times]\!] \tag{2.2}$$

with $\boldsymbol{e}_3 = (0, 0, 1)$ and $[\![\boldsymbol{\omega}\times]\!]$ the skew-symmetric matrix form of the vector cross product such that

$$[\![\boldsymbol{\omega}\times]\!] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{2.3}$$

and $\boldsymbol{g} = (0, 0, -g)$ the acceleration due to gravity. Note the distinction between the vector $\boldsymbol{g}$ and scalar g.

2.1 Reformulation in jerk

We follow [2.10] in considering the trajectories of the quadrocopter in terms of the jerk of the axes, allowing the system to be considered as a triple integrator in each axis and simplifying the trajectory generation task.

It is assumed that a thrice differentiable trajectory $\boldsymbol{x}(t)$ is available, where the jerk is written as $\boldsymbol{j} = \dddot{\boldsymbol{x}} = (\dddot{x}_1, \dddot{x}_2, \dddot{x}_3)$. The input thrust f is then found by applying the Euclidean norm $\|\cdot\|$ to (2.1),

$$f = \|\ddot{\boldsymbol{x}} - \boldsymbol{g}\|\,. \tag{2.4}$$

Squaring the above, taking the derivative and substituting for (2.1) yields

$$2 f \dot{f} = 2\,(\ddot{\boldsymbol{x}} - \boldsymbol{g})^T\, \boldsymbol{j} = 2\,(\mathbf{R}\boldsymbol{e}_3 f)^T\, \boldsymbol{j} \tag{2.5}$$

$$\dot{f} = \boldsymbol{e}_3^T \mathbf{R}^T \boldsymbol{j}. \tag{2.6}$$

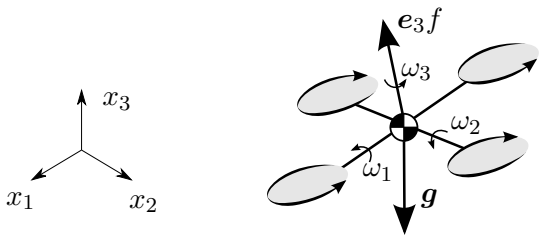

Figure 2.1. Dynamic model of a quadrocopter, acted upon by gravity $\boldsymbol{g}$, a thrust force f pointing along the (body-fixed) axis $\boldsymbol{e}_3$; and rotating with angular rate $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)$, with its position in inertial space given as (x_1, x_2, x_3).

Taking the first derivative of (2.1) yields

$$\boldsymbol{j} = \mathbf{R}[\![\boldsymbol{\omega}\times]\!]\boldsymbol{e}_3 f + \mathbf{R}\boldsymbol{e}_3\dot{f}. \tag{2.7}$$

After substitution, and evaluating the product $[\![\boldsymbol{\omega}\times]\!]\boldsymbol{e}_3$, it can be seen that the jerk $\boldsymbol{j}$ and thrust f values fix two components of the body rates:

$$\begin{bmatrix} \omega_2 \\ -\omega_1 \\ 0 \end{bmatrix} = \frac{1}{f}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{R}^T\boldsymbol{j}. \tag{2.8}$$

That the third component of body rates, ω_3, does not appear can be understood by noting that a rotation about the $\boldsymbol{e}_3$ axis does not affect the translational acceleration (2.1).

Using (2.4) and (2.8), the system inputs are given for a trajectory described in its jerk, with one remaining degree of freedom in ω_3. This could be fully specified if the full attitude of the quadrocopter were also known (specifically, the rotation about the thrust axis). For simplicity, here it will be assumed that $\omega_3 = 0$ and that this rotation is unimportant.

2.2 Feasibility constraints and decoupled axes

A quadrocopter trajectory described by (2.1) and (2.2) is considered to be feasible if the thrust and the magnitude of the body rates lie in some feasible set, defined as

$$0 < f_{\min} \leq f \leq f_{\max} \tag{2.9}$$

$$\|\boldsymbol{\omega}\| \leq \omega_{\max}. \tag{2.10}$$

Note that $f_{\min} > 0$ for fixed-pitch propellers with a fixed direction of rotation, and, specifically, that the requirement on the thrust input is non-convex. These limits are translated to limits on the jerk trajectory by squaring (2.4) and writing it in its components:

$$f_{\min}^2 \leq \ddot{x}_1^2 + \ddot{x}_2^2 + (\ddot{x}_3 + g)^2 \leq f_{\max}^2. \tag{2.11}$$

The following conservative box constraints are applied to yield convex constraints:

$$\ddot{x}_{\min\{1\}} = -\ddot{x}_{\max\{1\}} \leq \ddot{x}_1 \leq \ddot{x}_{\max\{1\}} \tag{2.12}$$

$$\ddot{x}_{\min\{2\}} = -\ddot{x}_{\max\{2\}} \leq \ddot{x}_2 \leq \ddot{x}_{\max\{2\}} \tag{2.13}$$

$$\ddot{x}_{\min\{3\}} = f_{\min} - g \leq \ddot{x}_3 \leq \ddot{x}_{\max\{3\}}. \tag{2.14}$$

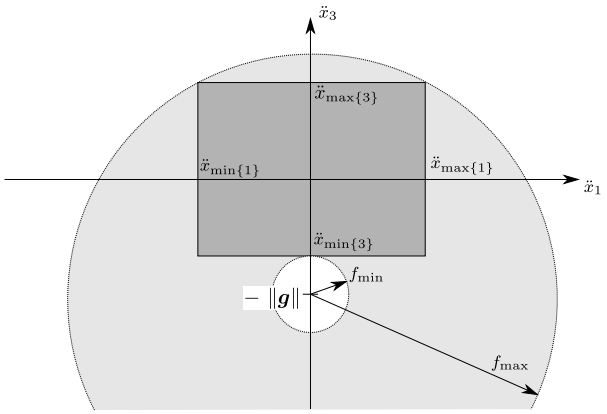

Figure 2.2. A cross-section of the feasible acceleration sets for a quadrocopter. The lightly shaded, non-convex, doughnut defines the true thrust limits of the vehicle, while the darker rectangular area defines the decoupled per-axis acceleration limits used here. Note that the circle of radius $f_{\max}$ is truncated in the graphic.

The resulting trajectories are guaranteed to be feasible with respect to the thrust limit if

$$\ddot{x}_{\max\{1\}}^2 + \ddot{x}_{\max\{2\}}^2 + \left(\ddot{x}_{\max\{3\}} + g\right)^2 \leq f_{\max}^2 \tag{2.15}$$

as visualised for two axes in Fig. 2.2.

By taking the (induced) norm of (2.8), an upper bound for the body rates can be found as a function of the jerk, as

$$\|\boldsymbol{\omega}\| \leq \frac{1}{f} \|\boldsymbol{j}\| \leq \frac{1}{f_{\min}} \|\boldsymbol{j}\| . \tag{2.16}$$

Applying the limit (2.10) to the above, and rearranging yields and upper bound on the allowable jerk per axis

$$j_{\max} = \frac{1}{\sqrt{3}} f_{\min}\, \omega_{\max} \tag{2.17}$$

under the worst case that all three axes produce the maximum allowable jerk $j_{\max}$ at the same time as the minimum thrust $f_{\min}$ is achieved.

3. Trajectory generation

Considering the system input to be the three-dimensional jerk, the quadrocopter dynamics become a set of three triple integrators, one in each axis, with states position, velocity and acceleration. The trajectory generation is rewritten as an optimal control problem,

with boundary conditions defined by the quadrocopter's initial and (desired) final states. The cost function to minimize is chosen as

$$J_{\text{coupled}} = \int_0^T \left(j_1(t)^2 + j_2(t)^2 + j_3(t)^2 \right) dt. \tag{2.18}$$

Note that, by rearranging (2.16), this cost function can be interpreted as an upper bound for a product of the inputs, since

$$f^2 \left\| \boldsymbol{\omega} \right\|^2 \leq j_1^2 + j_2^2 + j_3^2. \tag{2.19}$$

This implies that the cost function (2.18) can be split, such that each axis is minimized independently, while remaining meaningful in the context of the coupled three-dimensional trajectory.

3.1 Discrete time formulation

The trajectory generation problem for each decoupled axis is rendered finite dimensional by discretizing the time with uniform steps of size Δt. Each axis is then a discrete time linear, time invariant system in the state z, consisting of position, velocity and acceleration, with scalar jerk input $j = \dddot{x}$, where the axis subscripts have been neglected for convenience.

$$j[k] = \dddot{x}(k\Delta t) \tag{2.20}$$

$$z[k] = \begin{bmatrix} x(k\Delta t) & \dot{x}(k\Delta t) & \ddot{x}(k\Delta t) \end{bmatrix}^T \tag{2.21}$$

$$z[k+1] = A_d z[k] + B_d j[k] \tag{2.22}$$

$$A_d = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{2.23}$$

$$B_d = \begin{bmatrix} \frac{1}{6}\Delta t^3 & \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix}^T \tag{2.24}$$

The optimal control problem is to minimize the cost function

$$J = \sum_{k=0}^{N} j[k]^2 \tag{2.25}$$

subject to the above dynamics, satisfying the boundary conditions defined by the initial and final positions (x_0 and x_f, respectively), velocities ($\dot{x}_0$ and $\dot{x}_f$) and accelerations ($\ddot{x}_0$

and $\ddot{x}_f$):

$$z[0] = \begin{bmatrix} x_0 & \dot{x}_0 & \ddot{x}_0 \end{bmatrix}^T \tag{2.26}$$

$$z[N] = \begin{bmatrix} x_f & \dot{x}_f & \ddot{x}_f \end{bmatrix}^T \tag{2.27}$$

with the end stage calculated from the end time T as $N = \text{Round}(T/\Delta t)$.

The constraints on acceleration of (2.12) - (2.14) and the jerk limit (2.17) are affine functions of the state $z[k]$ and input $j[k]$ (where n is the axis under consideration):

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix} z[k] \leq \begin{bmatrix} \ddot{x}_{\min\{n\}} \\ \ddot{x}_{\max\{n\}} \end{bmatrix} \tag{2.28}$$

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} j[k] \leq \begin{bmatrix} j_{\max} \\ -j_{\max} \end{bmatrix}. \tag{2.29}$$

The quadratic cost function (2.25) with the linear equality constraints (2.22), (2.26) and (2.27), and the affine inequality constraints of (2.28) and (2.29), together define a convex optimisation problem. This is a special case of model predictive control [2.19] with a fixed end constraint, and with a diminishing rather than receding horizon (i.e. the trajectory is only planned to intercept).

There exist efficient methods for solving problems of this sort, with CVXGen [2.20], FORCES [2.21] and FiOrdOs [2.22] presenting techniques for creating C-code based solvers for specific instances of convex optimization problems.

Here, solvers are generated using the FORCES software of [2.21], which was able to generate solvers for large problems (here trajectories up to $N = 200$ are considered). FORCES uses efficient interior point methods tailored to convex multistage problems, as are typical in model predictive control applications, and allows for high-speed implementation with good numerical stability properties [2.21].

The generated solvers either return a solution that solves the problem to within some acceptable residuals, or returns that no solution is found. In reality, failure to find a solution can mean that:

- a solution exists, but the solver failed to find it due to reaching an internal limit;
- no solution exists to the conservatively constrained decoupled problem;
- no solution exists to the fully coupled nonlinearly constrained problem.

An example trajectory is shown in Fig. 2.3, where a one-dimensional trajectory is generated over a translation of 1.25 m from rest at the origin to rest in 1 s (or 50 steps at 50 Hz). The acceleration limits are set to $\ddot{x}_{\max} = -\ddot{x}_{\min} = 7\,\mathrm{m\,s^{-2}}$ and the jerk limit to $j_{\max} = 70\,\mathrm{m\,s^{-3}}$. The minimum and maximum acceleration limits, and the maximum jerk limit, are active for portions of this trajectory.

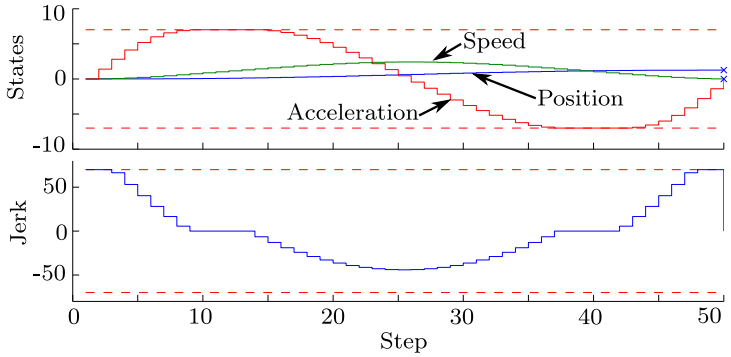

Figure 2.3. Example one-dimensional constrained trajectory solution, starting at rest at the origin and ending at $x_f = 1.25\,\mathrm{m}$, $\dot{x}_f = \ddot{x}_f = 0$ in $T = 1\,\mathrm{s}$, with acceleration limits $\ddot{x}_{\max} = -\ddot{x}_{\min} = 7\,\mathrm{ms}^{-2}$ and $j_{\max} = 70\,\mathrm{ms}^{-3}$. Each step represents 0.02 s.

3.2 Solution time

The statistical performance of the trajectory generator is investigated by solving trajectories from rest, for interception times varying from 0.2 to 4 s, in 20 ms intervals (i.e. 10 to 200 steps). For each trajectory length, 1000 end states are chosen uniformly at random in the range $x_f \in [-3, 3]\,\mathrm{m}$, $\dot{x}_f \in [-2, 2]\,\mathrm{m\,s}^{-1}$ and $\ddot{x}_f \in [-5, 5]\,\mathrm{m\,s}^{-2}$, and applying the constraints $\ddot{x}_{\max} = -\ddot{x}_{\min} = 7\,\mathrm{m\,s}^{-2}$ and $j_{\max} = 70\,\mathrm{m\,s}^{-3}$. The mean solution time over all 191000 trajectories was 1.5 ms, with a maximum of 13.7 ms.

These were calculated on a PC running Windows 7, with an Intel Core i7-2620M CPU at 2.70 GHz, with 4 GB RAM. The solver was compiled into a dynamically linked library using the Intel C++ Composer XE Windows: 2011.8.278, which was subsequently linked to an executable using Visual Studio 2008. In each case the optimisations were set to maximise speed.

3.3 Completeness

Because the generated solver does not check for feasibility, it does not distinguish between infeasible trajectories and those that hit iteration limits of the solver. The frequency of occurrence of these false negatives is investigated by generating one-dimensional trajectories from rest to end states of positions in the range $[0, 3.5]\mathrm{m}$, speeds in the range $[0, 5]\mathrm{m\,s}^{-1}$, and zero acceleration; with an end time of 1 s (50 steps) and limits as in the example trajectory above. The problem was also modelled in Yalmip [2.23], and solved as a quadratic program, allowing the detection of infeasible problems. Note that completeness here refers only to the decoupled triple integrator, not the fully coupled quadrocopter model; a discussion of the latter can be found in [2.24].

The results are visualised in Fig. 2.5: of 10'000 end states evaluated, 5278 were found to be feasible; of these feasible end states, the proposed solver found 85.0%.

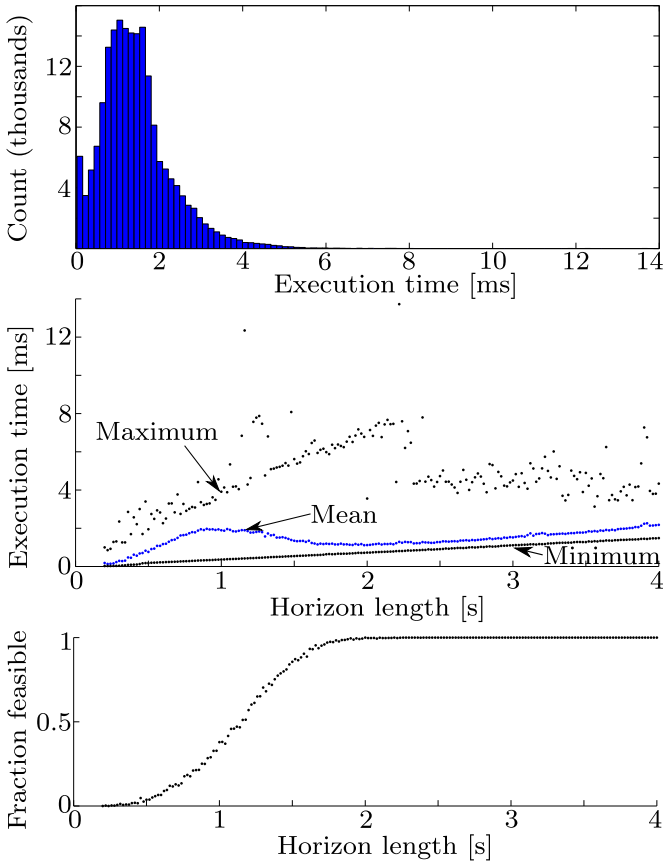

Figure 2.4. Performance of the constrained triple integrator trajectory solver, with trajectories generated from rest to a uniformly distributed random final state. The top plot shows a histogram of the solution times, using the data for all trajectory lengths, while the middle plot shows the execution time as a function of horizon length, and the bottom plot shows the fraction of randomly generated end states for which a trajectory could be found as a function of horizon length. The reduction in mean solution time at $T \approx 1\,\mathrm{s}$ can be explained by noting that, for shorter end times, almost all trajectories failed to find a solution.

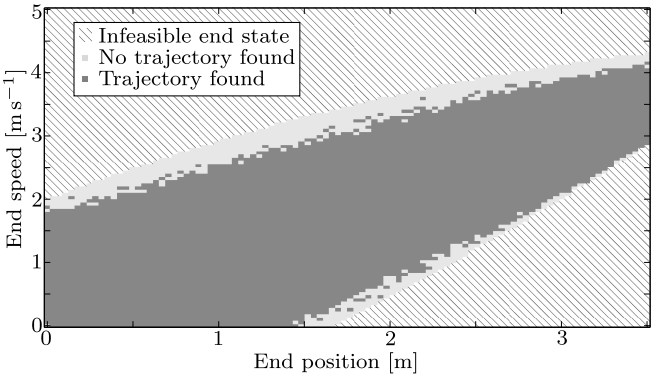

Figure 2.5. Completeness of the trajectory solver, showing whether a one-dimensional trajectory was found from rest to a specified end state within 1 s for 10'000 different end states. The middle grey area represents feasible end states for which the solver hit an internal limit, and failed to find a trajectory.

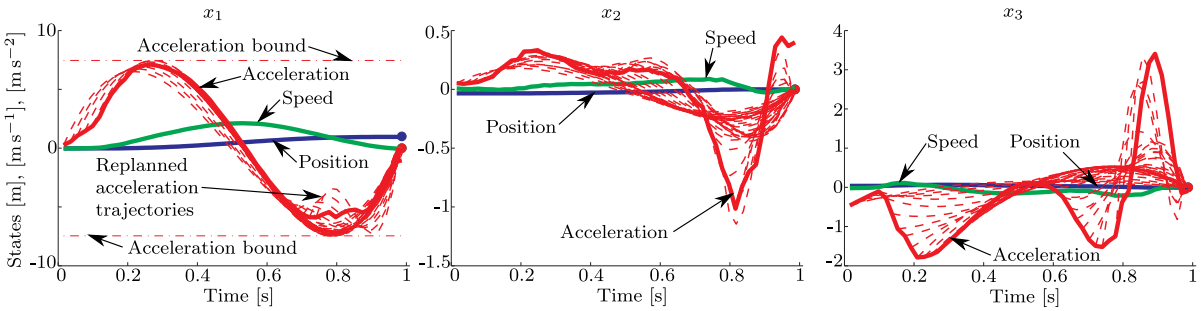

Figure 2.6. Resulting flight for the "easy" trajectory; refer to Section 4.1, where the goal is a rest-to-rest translation of 1m in x_1. The plot shows the trajectory for each axis (from left to right: x_1, x_2 and x_3) separately. The solid line shows the actual flown trajectory, and the dashed lines indicate the planned acceleration trajectories, as replanned at each stage. Note that, from approximately 0.8s onwards, feasible trajectories can no longer be found and the vehicle flies in feedback on the last valid trajectory.

4. Validation

The Flying Machine Arena (FMA) at the ETH Zurich is a platform for design and validation of autonomous aerial systems, and consists of a large motion capture volume and a fleet of quadrocopters. Commands for the three body rates and the collective thrust are sent at 50Hz over a wireless channel, and an onboard controller uses rate gyro measurements to generate motor thrust commands. The motion capture system is used to measure the position and attitude of the quadrocopter, fused into a full state estimate to be used for control.

The trajectory generator is implemented as a model predictive controller, solving a trajectory to intercept at each time step from the vehicle's current state, and applying the first input to the system. If the trajectory generator fails, the trajectory from the previous time step is followed using a feedback controller. Since the axes are independent, the constrained solution for each axis can be solved for in parallel, implying that a worst-case run-time of 13.7 ms (from the above statistical analysis) would still fit in the 20 ms control period.

The thrust and body rates are limited as below, where these values have been observed to match the FMA vehicles well.

$$f_{\min} = 5\,\mathrm{m\,s^{-2}} \tag{2.30}$$

$$f_{\max} = 20\,\mathrm{m\,s^{-2}} \tag{2.31}$$

$$\boldsymbol{\omega}_{\max} = 25\,\mathrm{rad\,s^{-1}} \tag{2.32}$$

4.1 Easy trajectory

An "easy" trajectory is examined first, where the goal is to translate 1 m in the x_1 direction, starting at rest and ending at rest. The upper thrust limit is divided amongst all axes equally, and this trajectory is considered "easy" because the planned trajectories

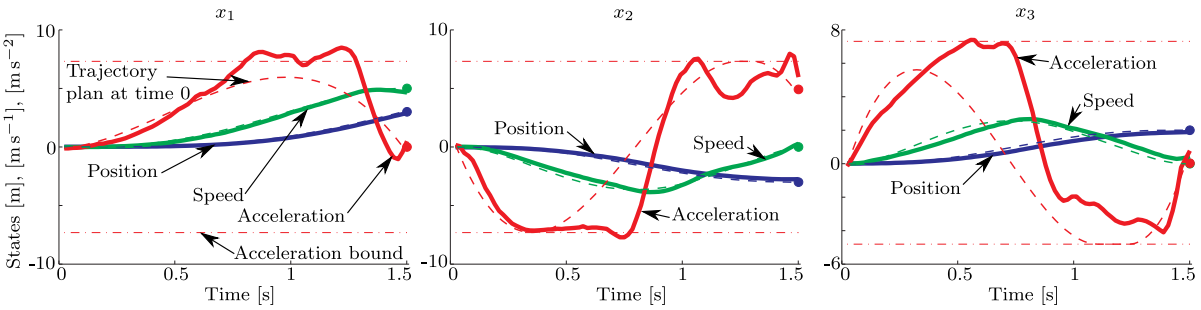

Figure 2.7. Resulting trajectories for the "hard" trajectory of Section 4.2: a quadrocopter starting at rest at the origin, and flying to an end state of $x_f = (3, -3, 2)\,\mathrm{m}$, $\dot{x}_f = (5, 0, 0)\,\mathrm{ms}^{-1}$ and $\ddot{x}_f = (0, 4.9, 0)\,\mathrm{ms}^{-2}$ in $T = 1.5\,\mathrm{s}$. Each plot shows the trajectory along a different axis in inertial space. The thick lines represent the actual trajectory as followed by the quadrocopter, while the thin broken line is the solution to the trajectory generation problem as obtained at the first time instant. Up to approximately 0.6 s, a new trajectory is generated at each time step (not shown), but from then onwards feasible trajectories can no longer be found and feedback is done using the last valid trajectory as reference. Refer to Fig. 2.8 for the corresponding inputs.

at time 0 does not have any active input constraints.

$$\ddot{x}_{\max\{1\}} = \ddot{x}_{\max\{2\}} = \ddot{x}_{\max\{3\}} \approx 7.31\,\mathrm{m\,s^{-2}} \tag{2.33}$$

The resulting flight is shown in Fig. 2.6, specifically, the system reached the end state to within state errors of 49 mm in position, $0.10\,\mathrm{m\,s^{-1}}$ in velocity and $1.1\,\mathrm{m\,s^{-2}}$ in acceleration, while remaining inside the feasibility constraints. The figure also shows that, although the initially planned trajectory does not hit any input constraints, later replanning results in trajectories that do have active constraints, due to disturbances.

4.2 Hard trajectory

Next, a "hard" trajectory is considered, one which involves active input constraints in each axis. As with the easy trajectory, the upper thrust limit was split equally amongst all axes. The quadrocopter starts at rest at the origin, with a target end state characterised by

$$x_f = (3, -3, 2)\,\mathrm{m}, \tag{2.34}$$
$$\dot{x}_f = (5, 0, 0)\,\mathrm{m\,s^{-1}}, \tag{2.35}$$
$$\ddot{x}_f = (0, 4.9, 0)\,\mathrm{m\,s^{-2}}; \tag{2.36}$$

in a time of $T = 1.5\,\mathrm{s}$. These end constraints can be interpreted as trying to pass through a window approximately 4.69 m away, with a speed of $5\,\mathrm{m\,s^{-1}}$ at an attitude at 30° from the horizontal with no vertical acceleration at the end.

A resulting trajectory is shown in Fig. 2.7, with the achieved end state

$$x_f = (2.89, -2.75, 1.89)\,\mathrm{m}, \tag{2.37}$$
$$\dot{x}_f = (4.65, 0.28, 0.44)\,\mathrm{m\,s^{-1}}, \tag{2.38}$$
$$\ddot{x}_f = (-0.49, 5.96, 0.78)\,\mathrm{m\,s^{-2}}; \tag{2.39}$$

or a position error of 0.29 m, velocity error of $0.63\,\mathrm{m\,s^{-1}}$. The direction of thrust at the end differs from the desired by an angle of 3.6°. The position error equates to approximately 6% of the total translation, and the velocity error to 13% of the total velocity change. From approximately 0.6 s onwards, no feasible trajectories can be found from the vehicle's current position, and the last valid trajectory is flown in feedback. This is possible because the constraints used for trajectory generation are conservative, especially as the true feasible regions are non-convex.

The inputs applied during this trajectory are shown in Fig. 2.8, where it can be seen that the inputs stay within the thrust and body rate constraints. The individual axes do briefly violate the acceleration constraints during the period where no feasible trajectory could be found from the current position (therefore the vehicle was flying on feedback with the last valid trajectory as reference). The conservative nature of the body rate limits of (2.17) can be also seen: the commanded body rates are generally very low, rising only under trajectory following feedback control near intercept.

4.3 Box constraint selection

If additional information on the trajectory is available in advance, or sufficient planning time is available, the convex bounds on the accelerations and jerk can be chosen more efficiently (refer to Fig. 2.2). Similarly, if the trajectory is solved for off line, the bounds can be iterated on to make use of them more efficiently.

As an example, consider a rest-to-rest motion, translating 4 m along the x_1 axis. Using the general limits as above, the fastest end time for which a feasible trajectory is found is 1.60 s. However, because the motion is only along one axis, the box constraints on the acceleration of the other two axes can be tightened. Specifically, limiting the accelerations in the x_2 and x_3 axes to $\pm 1\,\mathrm{m\,s^{-2}}$ (to maintain the ability to compensate for disturbances), the limits for x_1 can be raised to $\pm 16.80\,\mathrm{m\,s^{-2}}$. Furthermore, the minimum thrust value can be raised to $8.81\,\mathrm{m\,s^{-2}}$, allowing the jerk limit to be raised to $127.16\,\mathrm{m\,s^{-3}}$ by (2.17). Using these limits, the trajectory generator finds a solution for an end time of 1.24 s, an improvement of 22.5%. Note that these bounds are still conservative, due to the non-convex nature of the true feasible region.

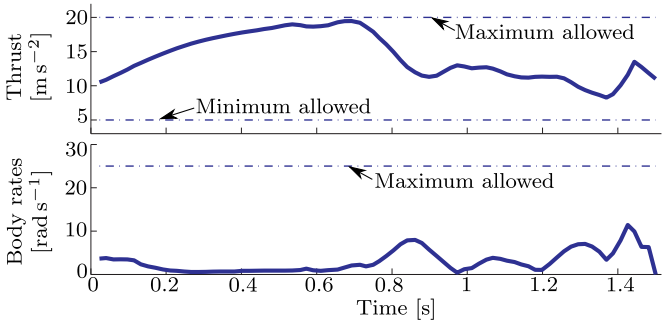

Figure 2.8. Inputs for the hard trajectory of Section 4.2. Note that the thrust limits are hit, but the magnitude of the body rate command stays well below the maximum allowed. This due to the conservative nature of the limit (2.17).

5. Outlook

This paper presents a method for calculating state interception trajectories for quadrocopters, from arbitrary initial conditions to a desired end state characterised by a position, velocity and acceleration, resulting in commands of thrust and two components of the body rates. The end acceleration constraint can be viewed as defining the attitude of the vehicle to within a rotation about the thrust axis. A possible extension to this research would be calculating an input trajectory for the third component of the body rates, such that the final degree of freedom of the attitude can also be specified.

The constrained trajectory generation problem is posed as a convex optimisation problem for each decoupled axis, which can be solved in real time. Therefore this technique is suitable for use in feedback as a model predictive controller; it also naturally handles cases where the desired end state evolves over time, such as when trying to hit an uncertain target where the target prediction evolves in real time.

The convex optimisation problem could be easily modified to achieve different goals, e.g. by removing the equality constraint on end velocity and acceleration, and adding the magnitude of the end velocity to the cost function, one solves a problem somewhat similar to that of [2.11].

Given some initial and final states, it would also be very useful to have an analytical method of calculating what the minimum required time horizon is. One possible solution would be to calculate the bang-bang trajectories as in [2.10].

The conservative nature of the jerk bounds means that only a fraction of the allowable body rates is typically used. Further work could be done to improve these bounds, e.g. using the unconstrained solution to find a better bound for the lowest thrust value produced along the trajectory. Alternatively, the optimisation problems could be solved in series and using the acceleration values from one solution to provide tighter (time varying) bounds for the next axis.

By combining all three axes into one optimisation problem that would now have 12 primary optimisation variables per stage, the constraints could be made even less

conservative. The maximum thrust constraint (the right-hand side of (2.11)) can be encoded directly, noting that this is a convex quadratic constraint, which can also be solved efficiently. This would also allow rewriting the box constraint on jerk, (2.17), as a norm constraint, making it somewhat less conservative. Furthermore, by combining all axes in the optimisation problem, it becomes straight forward to define e.g. zones where the quadrocopter is allowed to fly (using the decoupled axes presented here, only zones aligned with the axes could be used). This has the major drawback of having to solve one large optimisation problem, instead of three small problems.

The problem can be made much less restrictive by allowing the end state to lie in a region "near" the desired end state, rather than equalling exactly the desired end state. The choice of the size of this region now becomes a design variable, and will depend on the application.

Acknowledgement

The Flying Machine Arena is the result of contributions of many people, a full list of which can be found at `http://flyingmachinearena.org`.

Specifically, we would like to thank Markus Hehn for the many discussions on optimal control and quadrocopter dynamics. We would also like to thank Alex Domahidi and Stephan Richter for their discussions on convex optimisation.

This research was supported by the Swiss National Science Foundation through grant agreement number 138112.

References

[2.1] S. Weiss, M. Achtelik, M. Chli, and R. Siegwart, "Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 31–38.

[2.2] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "A prototype of an autonomous controller for a quadrotor UAV", in *Proceedings of the European Control Conference*, Kos, Greece, 2007, pp. 1–8.

[2.3] S. Lupashin, A. Schoellig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 1642–1648.

[2.4] K. Alexis, G. Nikolakopoulos, A. Tzes, and L. Dritsas, "Coordination of helicopter UAVs for aerial forest-fire surveillance", in *Applications of intelligent control to engineering systems*, Springer, 2009, pp. 169–193.

[2.5] N. E. Serrano, "Autonomous quadrotor unmanned aerial vehicle for culvert inspection", PhD thesis, Massachusetts institute of technology, 2011.

[2.6] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment", in *Proc. of the AIAA Guidance, Navigation, and Control Conference*, 2007, pp. 1–20.

[2.7] P. Martin and E. Salaün, "The true role of accelerometer feedback in quadrotor control", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 1623–1629.

[2.8] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors", in *Int. Symposium on Experimental Robotics*, 2010.

[2.9] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors", in *International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 2520–2525.

[2.10] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control", in *IFAC World Congress*, vol. 18, 2011, pp. 1485–1491.

[2.11] M. Hehn and R. DÁndrea, "Real-time trajectory generation for interception maneuvers with quadrocopters", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[2.12] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[2.13] P. Bouffard, A. Aswani, and C. Tomlin, "Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 279–284.

[2.14] K. Alexis, C. Papachristos, G. Nikolakopoulos, and A. Tzes, "Model predictive quadrotor indoor position control", in *Control & Automation (MED), 2011 19th Mediterranean Conference on*, 2011, pp. 1247–1252.

[2.15] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "MPC with nonlinear Hinf control for path tracking of a quad-rotor helicopter", in *Proc. of the IFAC World Congress*, 2008, pp. 8564–8569.

[2.16] M. W. Mueller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 5113–5120.

[2.17] S. Bouabdallah, P. Murrieri, and R. Siegwart, "Design and control of an indoor micro quadrotor", in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 5, 2004, pp. 4393–4398.

[2.18] P. H. Zipfel, *Modeling and Simulation of Aerospace Vehicle Dynamics Second Edition*. American Institute of Aeronautics and Astronautics, 2007.

[2.19] M. Morari and J. H. Lee, "Model predictive control: Past, present and future", *Computers and chemical engineering*, vol. 23, no. 4-5, pp. 667–682, 1999.

[2.20] J. Mattingley and S. Boyd, "CVXGen: A code generator for embedded convex optimization", *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[2.21] A. Domahidi, A. Zgraggen, M. Zeilinger, M. Morari, and C. Jones, "Efficient interior point methods for multistage horizons arising in receding horizon control", in *IEEE Conference on Decision and Control (CDC)*, 2012.

[2.22] F. Ullman, "A Matlab toolbox for C-code generation for first order methods", Master's thesis, ETH Zurich, Zurich, Switzerland, 2011.

[2.23] J. Lofberg, "YALMIP : A toolbox for modeling and optimization in MATLAB", in *IEEE International Symposium on Computer Aided Control Systems Design*, 2004, pp. 284–289.

[2.24] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control", *Autonomous Robots*, vol. 33, pp. 69–88, 1 2012.

Part B

Failsafe strategies and novel vehicles

Paper P3

Relaxed hover solutions for multicopters: application to algorithmic redundancy and novel vehicles

Mark W. Mueller and Raffaello D'Andrea

Abstract

This paper presents a relaxed definition of hover for multicopters with propellers pointing in a common direction. These solutions are found by requiring that the multicopter remain substantially in one position, and that the solutions be constant when expressed in a coordinate system attached to the vehicle. The vehicle's angular velocity is then shown to be either zero or parallel to gravity. The controllability of a vehicle's attitude about these solutions is then investigated. These relaxed hover solutions may be applied as an algorithmic failsafe, allowing for example a quadrocopter to fly despite the complete loss of one, two, or three of its propellers. Experimental results validate the quadrocopter failsafe for two types of failure (a single propeller, and two opposing propellers failing), and a nonlinear simulation validates the remaining two types of failure (two adjacent, and three propellers failing). The relaxed hover solutions are also shown to allow a multicopter to maintain flight in spite of extreme centre of mass offsets. Finally, the design and experimental validation of three novel vehicles is presented.

Accepted for publication in the *International Journal of Robotics Research.*

1. Introduction

Multicopters have found broad use as research platforms, used e.g. for vision based pose estimation with quadrocopters [3.1] and hexacopters [3.2], and also as platforms allowing for new capabilities. For example, the use of both quadrocopters and hexacopters for whale monitoring is investigated by [3.3], and hexacopters are used by [3.4] for weed research; a team of quadrocopters is used to carry a slung load by [3.5] and an octocopter is used to calibrate radio telescope antennae by [3.6]. Multicopters also hold promise as goods delivery vehicles [3.7].

These vehicles typically consist of an even number of propellers all pointing in a common direction, with half of the propellers having the opposite handedness from the remainder, and thus also rotating in the opposite direction. The propellers are then mechanically arranged such that the torques they produce can be made to sum to zero while the propeller thrusts support the vehicle's weight. Differences between propeller thrusts allow the vehicle's attitude to be changed, and the sum of the thrusts is used to accelerate the vehicle. This paper will only consider multicopters where all propellers have parallel axes of rotation (note however, that alternative designs exist where this is not the case, e.g. [3.8] or [3.9]).

Amongst others, a motivation for using a multicopter with six or more propellers, instead of a four propeller quadrocopter, is that the vehicle is able to maintain normal flight if one of the propellers fails (see e.g. [3.10] for a hexacopter design and [3.11] for an octocopter rotor failure strategy). The need for having more than four propellers follows from the requirement that the vehicle be able to hover with zero angular velocity even after the failure of one of the actuators.

In this paper, this requirement of having zero angular velocity is relaxed, and instead hover solutions are searched for during which the position remains approximately constant, and where the solutions may be described with constant parameters: these relaxed hover solutions form a superset of those typically used for multicopters. This means that the vehicle may rotate at a constant velocity in hover. They allow for the design of novel hover-capable vehicles, and may also be employed to offer redundancy for multicopters experiencing an actuator failure. Since the solutions are constant in the body frame the powerful techniques of linear time invariant systems theory may be applied for analysis and control design.

An approach is presented to formulate and solve for such relaxed hover solutions, and for designing linear, time invariant controllers to control a hovering vehicle. This approach is used to compute solutions for a quadrocopter experiencing any combination of up to three complete propeller failures. For two of the cases, experimental validation is provided, while the approach for the remaining two types of failure are validated in a nonlinear simulation. The case of losing two opposing propellers is investigated in detail, with a specific focus on vehicle mass and aerodynamic properties that would render the vehicle uncontrollable.

The approach may also be used to control a multicopter where the centre of mass

is located far from the propellers' geometric centre. A specific example is given for a quadrocopter with an eccentric centre of mass, where the conventional hover solution of zero angular velocity requires infeasible motor forces. However, by allowing the vehicle to rotate, a feasible solution is found, even if the centre of mass lies outside of the propellers' convex hull.

The methods presented are also applied to design novel, rotating body vehicles. A family of such vehicles, called "spinners", is presented, where each vehicle comprises a number of propellers arranged in a rotationally symmetric pattern about the vehicle's centre of mass, and all propellers have the same handedness and rotate in the same direction. At hover these vehicles rotate at a high angular velocity, in the opposite direction of their propellers. A two, three, and four propeller spinner is presented. In addition to their dynamic properties, when equipped with a camera, such a vehicle could be used as a low-cost omnidirectional flying camera, similar to e.g. [3.12] or [3.13] – note however that the spinners tend to rotate at higher velocity than the vehicles in these references, such that motion blur might present a problem.

1.1 Related work

A survey on fault detection and diagnosis and fault-tolerant control strategies for unmanned rotary wing vehicles is given by [3.14], and examples of commercial solutions are the emergency parachutes of [3.15] or those described by [3.16]. Partial failure of a quadrocopter actuator is investigated for example in [3.17], [3.18] and [3.19]. Complete propeller failure of a single propeller of a quadrocopter is investigated in [3.20], [3.21], and [3.22], where the strategy is to give up controlling the vehicle's yaw angle, and use the remaining propellers to achieve a horizontal spin. A flight strategy to cope with an actuator failure on an octocopter is presented in [3.11], and a hexacopter design with a focus on actuator redundancy is presented in [3.23].

The failsafe method presented in this paper extends those above by finding solutions when using fewer than three propellers. Furthermore, this paper presents a family of solutions when using only three propellers (such as a quadrocopter having lost one propeller) – the user may thus select the solution most appropriate to the vehicle and the situation.

The controllability of a flying, rotating vehicle with three mutually orthogonal propellers is analysed in [3.24], and a circular flight path for this vehicle is designed for it in [3.25] – the vehicle concept uses forward tilted propellers to induce rotation of the vehicle, so that the vehicle then also spins about an axis similarly to the spinners proposed in this work.

This paper will focus on vehicles that do not rely on aerodynamic effects (apart from drag and the propellers) for their stability – this is in contrast to maple-seed-like vehicles such as the Samara [3.26], or vehicles like the Spincopter [3.27].

This work follows on a conference paper published previously [3.28], and extends the results by

- presenting a more detailed derivation of the salient effects acting on a multicopter,

- deriving general conditions of the hover solutions without assumptions on the number of propellers, the locations of the propellers, or the mass distribution of the vehicle,
- likewise deriving a general framework for establishing attitude controllability,
- considering additionally the failure case of a quadrocopter losing two adjacent propellers,
- showing how the relaxed hover solutions may be applied to multicopters with large centre of mass offsets, and
- presenting the design of three novel multicopters.

1.2 Notation

Boldface symbols like $\boldsymbol{g}$ are used throughout this paper to denote vectors in three-dimensional space, while non-boldface symbols like m will generally be used for scalars, with exceptions made explicit. The coordinate system in which a vector is expressed will be denoted by a superscript, for example $\boldsymbol{g}^E$ expresses $\boldsymbol{g}$ in coordinate system E. Where possible, vectors will be left without expressing them in any coordinate system. A subscript will be used to express a relationship or attribute, for example $\boldsymbol{\omega}_{BE}$ represents the angular velocity of B with respect to E. The short-hand notation (p, q, r) will be used to compactly denote a column vector.

1.3 Organisation

This paper is organised as follows: in Section 2 the dynamic model of a generic multicopter is presented. Hover solutions are derived in Section 3 and the controllability of the vehicle about these solutions is investigated in Section 4. The approach is applied as an actuator failsafe for quadrocopters in Section 5, and applied to a quadrocopter with a large centre of mass offset in Section 6. A novel family of multicopters is presented in Section 7, and the paper concludes in Section 8.

2. Multicopter modelling

This section derives the equations of motion for a multicopter, with a special focus on the attitude dynamics. A model for the thrust force, reaction torque, and mechanical power consumption of a propeller is also presented. The equations are used in later sections to solve for hover solutions, and the power consumption of the propellers is used to compare different solutions.

Fig. 3.1 shows a multicopter with $N_p = 3$ propellers. The multicopter has a total mass m (including propellers), and its position relative to some point fixed in an inertial frame is expressed as $\boldsymbol{d}$. The mass moment of inertia of the multicopter excluding the

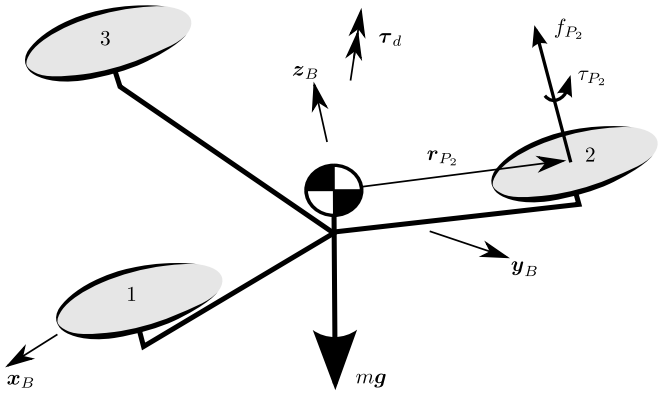

Figure 3.1. A multicopter with three propellers, showing the definition of the symbols used to derive its dynamics equations. The axes of rotation of the propellers are parallel, and are fixed with respect to the body.

propellers is $\boldsymbol{I}_B$, and the multicopter is rotating at an angular velocity $\boldsymbol{\omega}_{BE}$ with respect to the inertial frame.

The scalar thrust force f_{P_i} of each propeller i points in the body-fixed direction $\boldsymbol{z}_B$, and a body-fixed coordinate system B is defined so that $\boldsymbol{z}_B^B = (0, 0, 1)$. A perpendicular vector $\boldsymbol{x}_B^B = (1, 0, 0)$ is also defined, pointing towards propeller 1. Each thrust force i acts at a displacement $\boldsymbol{r}_{P_i}$ from the multicopter's centre of mass. The angular velocity of propeller i with respect to the inertial frame is written as $\boldsymbol{\omega}_{P_iE}$.

In addition to thrust, each propeller produces a scalar reaction torque τ_{P_i} opposing the propeller's angular velocity and parallel to $\boldsymbol{z}_B$, as a result of aerodynamic drag acting on the propeller blade. The mass moment of inertia of propeller i is $\boldsymbol{I}_{P_i}$. As a simplification, because the propellers rotate much faster than the vehicle's body, the propellers will be treated as symmetric about their axes of rotation; their moment of inertia is then constant when expressed in a body-fixed frame, independent of the propeller's instantaneous rotation angle.

It will be assumed that the vehicle travels at low translational velocities, such that translational drag forces may be neglected. For a discussion of such effects see e.g. [3.29]. As a result of the multicopter's angular velocity, a drag torque $\boldsymbol{\tau}_d$ acts on the multicopter. Gravity acts on the vehicle as a force $m\boldsymbol{g}$.

The translational dynamics of the vehicle, expressed in the inertial coordinate system E, may now be composed as [3.30]

$$m\ddot{\boldsymbol{d}}^E = \boldsymbol{z}_B^E \sum_{i=1}^{N_p} f_{P_i} + m\boldsymbol{g}^E. \tag{3.1}$$

The orientation of the body with respect to the inertial reference frame is captured by the coordinate transformation matrix $\boldsymbol{C}^{EB}$, such that a vector transforms as $\boldsymbol{v}^E =$

$\boldsymbol{C}^{EB}\boldsymbol{v}^B$. The differential equation for this matrix is [3.30]

$$\dot{\boldsymbol{C}}^{EB} = \boldsymbol{C}^{EB}[\![\boldsymbol{\omega}_{BE}^B]\!] \tag{3.2}$$

where $[\![\boldsymbol{a}]\!]$ represents the skew-symmetric matrix form of the cross product, so that $[\![\boldsymbol{a}]\!]\boldsymbol{b} = \boldsymbol{a}\times\boldsymbol{b}$ for any 3D vectors $\boldsymbol{a}$ and $\boldsymbol{b}$. Recall that $\boldsymbol{\omega}_{BE}^B$ represents the vehicle's angular velocity with respect to the inertial frame $\boldsymbol{\omega}_{BE}$, as expressed in the body-fixed frame B. The angular dynamics expressed in the body-fixed coordinate system are [3.30]

$$\begin{aligned}\boldsymbol{I}_B^B\dot{\boldsymbol{\omega}}_{BE}^B + \sum_{i=1}^{N_p}\boldsymbol{I}_{P_i}^B\dot{\boldsymbol{\omega}}_{P_iE}^B + [\![\boldsymbol{\omega}_{BE}^B]\!]\left(\boldsymbol{I}_B^B\boldsymbol{\omega}_{BE}^B + \sum_{i=1}^{N_p}\boldsymbol{I}_{P_i}^B\boldsymbol{\omega}_{P_iE}^B\right)\\ = \sum_{i=1}^{N_p}\left([\![\boldsymbol{r}_{P_i}^B]\!]\boldsymbol{z}_B^B f_{P_i} + \boldsymbol{z}_B^B\tau_{P_i}\right) + \boldsymbol{\tau}_d^B.\end{aligned} \tag{3.3}$$

The first two terms are the time derivative of respectively the vehicle's and the propellers' angular velocities with respect to the inertial frame, as expressed in the body-fixed frame. The third term expresses the cross-coupling of the angular momentum in the system, due to taking the derivative in a non-inertial frame. The right-hand side of the equation represents all the moments acting upon the body, consisting of the propeller forces acting at a distance from the centre of mass, the propeller reaction torques, and the vehicle drag torque.

2.1 Propeller model

It is assumed that the pitch of each of the propellers is fixed and that a given propeller's thrust and torque are functions only of the propeller's angular velocity with respect to the air. Specifically, these functions are taken to be proportional to the square of the angular velocity [3.31], and the air is assumed to be stationary with respect to the inertial frame. The thrust and torque are then characterised by the propeller coefficients κ_{f_i} and κ_{τ_i} as follows:

$$f_{P_i} = \kappa_{f_i}\left(\boldsymbol{\omega}_{P_iE}\cdot\boldsymbol{z}_B\right)|\boldsymbol{\omega}_{P_iE}\cdot\boldsymbol{z}_B| \tag{3.4}$$

$$\tau_{P_i} = -\kappa_{\tau_i}\left(\boldsymbol{\omega}_{P_iE}\cdot\boldsymbol{z}_B\right)|\boldsymbol{\omega}_{P_iE}\cdot\boldsymbol{z}_B| \tag{3.5}$$

with $\boldsymbol{\omega}_{P_iE} = \boldsymbol{\omega}_{P_iB} + \boldsymbol{\omega}_{BE}$ the rotational velocity of the propeller with respect to the air, and $\cdot$ denoting the vector inner product. The torque coefficient κ_{τ_i} is positive, and the sign of κ_{f_i} is a function of the propeller's handedness.

The propeller's scalar speed Ω_{P_i} with respect to the body is typically controlled by

an electronic speed controller, so that

$$\boldsymbol{\omega}_{P_iB} = \Omega_{P_i}\boldsymbol{z}_B. \tag{3.6}$$

Note that this is a very simple propeller model, neglecting for example the translation of the propeller's centre. An example of a more complicated model is given by [3.29], where a relationship between a quadrocopter's translational motion and a horizontal propeller force is given. Such effects are here assumed to be small disturbances, which will be compensated by feedback control.

1) Power consumption The mechanical power consumed by a propeller may be computed as the product of the torque it produces and the angular velocity of the propeller with respect to the body, as given below.

$$P_{P_i} = -\tau_{P_i}\Omega_{P_i}. \tag{3.7}$$

The mechanical power is used in later sections to compare the efficiency of different hover solutions. This model may be extended, for example, with information about the electrical efficiency of the vehicle's powertrain to compute the power required for the vehicle to hover.

2.2 Rotational drag model

The aerodynamic drag torque $\boldsymbol{\tau}_d$ will in general be very hard to model, and will be a function of the geometry of the vehicle, the angular velocity of the vehicle, and the angular velocities of the propellers. The model used here is based on the form drag of a translating object, where the magnitude of the drag force is quadratic in the magnitude of the translational velocity [3.32].

Here it is assumed that the magnitude of the torque is quadratic in the vehicle's angular velocity:

$$\boldsymbol{\tau}_d = -\left\|\boldsymbol{\omega}_{BE}\right\| \boldsymbol{K}_d\boldsymbol{\omega}_{BE} \tag{3.8}$$

where $\boldsymbol{K}_d$ is expressed as a constant 3×3 matrix in a body-fixed coordinate system, and the Euclidean norm of a vector is written as $\|\cdot\|$.

3. Hover solutions

The conventional hover condition for a multicopter is at zero acceleration and zero angular velocity, with the thrust vector pointing opposite to gravity. For a symmetric quadrocopter in hover, for example, this implies that each motor produces a force equal to one

quarter of the vehicle's weight. The propellers' angular momenta then sum to zero, so that the vehicle also has zero total angular momentum in hover. In this situation, the cross coupling in (3.3) disappears and the vehicle's dynamics in the three translational directions decouple to first order at the equilibrium. An example of a control design for a quadrocopter can be found in [3.33].

The definition of hover is relaxed to include flight conditions where the vehicle remains substantially at one point in space, where the vehicle may have non-zero angular velocity. The translational acceleration may likewise be non-zero, but must average to zero for the vehicle to remain substantially in the same position. Only solutions which are constant when described in a body-fixed frame will be considered, with specifically the propellers' angular velocity and the vehicle's angular velocity being constant. These solutions are then time-invariant, allowing for the design of time invariant controllers. The rich literature and powerful tools available for the design and analysis of linear time invariant systems may then be applied. Such a hover solution can be fully described by $N_p + 6$ variables: the N_p propeller speeds, three components of the vehicle's angular velocity, and three components of the vehicle's orientation.

3.1 Non-zero angular velocity during hover

If a multicopter's angular velocity with respect to the inertial frame $\boldsymbol{\omega}_{BE}$ is zero then $\boldsymbol{z}_B^E$ is constant, and by (3.1) the thrust direction $\boldsymbol{z}_B$ must point opposite to gravity. The total thrust force must then equal the weight of the vehicle, so that the vehicle has zero acceleration in hover. Furthermore, by (3.3) the moments produced by the propellers must also sum to zero – this is the conventional hover solution.

For a constant non-zero angular velocity the orientation of the body with respect to an inertial frame at time t may be solved for in closed form from (3.2) as

$$\boldsymbol{C}^{EB}(t) = \boldsymbol{C}^{EB}(0) \exp\left(\left[\!\left[\bar{\boldsymbol{\omega}}_{BE}^{B} t\right]\!\right]\right) \tag{3.9}$$

where $\exp(\cdot)$ is the matrix exponential. An overbar will be used to denote variables that are constant at the hover solution, such as $\bar{\boldsymbol{\omega}}_{BE}^{B}$.

Using Rodrigues' formula [3.34] and interpreting $\bar{\boldsymbol{\omega}}_{BE}^{B} t$ as a rotation vector of constant direction and increasing magnitude, (3.9) can be solved for explicitly as

$$\begin{aligned} \exp\left(\left[\!\left[\bar{\boldsymbol{\omega}}_{BE}^{B} t\right]\!\right]\right) = \mathbf{I} &+ \frac{1}{\left\|\bar{\boldsymbol{\omega}}_{BE}^{B}\right\|} \sin\left(\left\|\bar{\boldsymbol{\omega}}_{BE}^{B}\right\| t\right) \left[\!\left[\bar{\boldsymbol{\omega}}_{BE}^{B}\right]\!\right] \\ &+ \frac{1}{\left\|\bar{\boldsymbol{\omega}}_{BE}^{B}\right\|^2} \left(1 - \cos\left(\left\|\bar{\boldsymbol{\omega}}_{BE}^{B}\right\| t\right)\right) \left[\!\left[\bar{\boldsymbol{\omega}}_{BE}^{B}\right]\!\right]^2 \end{aligned} \tag{3.10}$$

with $\mathbf{I}$ the identity matrix. This is periodic, with period $T_{\text{hvr}} = 2\pi / \left\|\bar{\boldsymbol{\omega}}_{BE}\right\|$.

Since all other quantities are constant, the acceleration of the vehicle in hover must also be periodic, with period T_{hvr}. The average acceleration of the vehicle can be calculated

by averaging (3.1) over one period, where the time varying quantities have been made explicit below:

$$\frac{1}{T_{\text{hvr}}}\int_0^{T_{\text{hvr}}} m\ddot{\boldsymbol{d}}^E(t)\,dt = \frac{1}{T_{\text{hvr}}}\int_0^{T_{\text{hvr}}}\left(\boldsymbol{z}_B^E(t)\sum_{i=1}^{N_p}\bar{f}_{P_i} + m\boldsymbol{g}^E\right)dt \tag{3.11}$$

Transforming $\boldsymbol{z}_B^E(t)$ into the body-fixed frame, setting the above equal to zero (so that the average acceleration is zero) and simplifying yields

$$-m\boldsymbol{g}^E = \frac{1}{T_{\text{hvr}}}\int_0^{T_{\text{hvr}}}\boldsymbol{C}^{EB}(t)\,dt\;\boldsymbol{z}_B^B\sum_{i=1}^{N_p}\bar{f}_{P_i}. \tag{3.12}$$

The integral on the right hand side may be thought of as the average coordinate transformation of the vehicle, and can be solved for by substituting (3.9)-(3.10):

$$\frac{1}{T_{\text{hvr}}}\int_0^{T_{\text{hvr}}}\boldsymbol{C}^{EB}(t)\,dt = \boldsymbol{C}^{EB}(0)\left(\mathbf{I} + \frac{1}{\left\|\bar{\boldsymbol{\omega}}_{BE}^B\right\|^2}[\![\bar{\boldsymbol{\omega}}_{BE}^B]\!]^2\right). \tag{3.13}$$

Substituting and simplifying yields

$$-m\boldsymbol{g}^E = \left(\frac{\boldsymbol{z}_B\cdot\bar{\boldsymbol{\omega}}_{BE}}{\left\|\bar{\boldsymbol{\omega}}_{BE}\right\|^2}\sum_{i=1}^{N_p}\bar{f}_{P_i}\right)\boldsymbol{C}^{EB}(0)\bar{\boldsymbol{\omega}}_{BE}^B. \tag{3.14}$$

A requirement on the total thrust produced can be derived from the above by applying the Euclidean norm:

$$\sum_{i=1}^{N_p}\bar{f}_{P_i} = \frac{m\left\|\boldsymbol{g}\right\|\left\|\bar{\boldsymbol{\omega}}_{BE}\right\|}{\left|\boldsymbol{z}_B\cdot\bar{\boldsymbol{\omega}}_{BE}\right|}. \tag{3.15}$$

Note that this assumes that the sum of the propeller forces is positive.

Substituting (3.15) into (3.14) yields the condition that the angular velocity must be parallel to gravity in hover at time 0:

$$\frac{\operatorname{sgn}\left(\boldsymbol{z}_B\cdot\bar{\boldsymbol{\omega}}_{BE}\right)}{\left\|\bar{\boldsymbol{\omega}}_{BE}\right\|}\boldsymbol{C}^{EB}(0)\bar{\boldsymbol{\omega}}_{BE}^B = -\frac{1}{\left\|\boldsymbol{g}\right\|}\boldsymbol{g}^E \tag{3.16}$$

where $\operatorname{sgn}(\cdot)$ is the signum function returning the sign of its argument. As it is constant, for any hover solution with constant angular velocity and constant motor forces, the

angular velocity always remains parallel to gravity. This also implies that the vehicle's vertical acceleration is always zero in hover.

Given some angular velocity $\bar{\boldsymbol{\omega}}_{BE}^{B}$, the total thrust force produced is constrained by (3.15), and (3.16) constrains the vehicle's attitude so that the angular velocity is parallel to gravity. Additionally, the angular acceleration is required to be zero at hover, so that three additional constraints appear from (3.3):

$$[\![\bar{\boldsymbol{\omega}}_{BE}^{B}]\!]\left(\boldsymbol{I}_{B}^{B}\bar{\boldsymbol{\omega}}_{BE}^{B}+\sum_{i=1}^{N_p}\boldsymbol{I}_{P_i}^{B}\bar{\boldsymbol{\omega}}_{P_iE}^{B}\right)=\sum_{i=1}^{N_p}\left([\![\boldsymbol{r}_{P_i}^{B}]\!]\boldsymbol{z}_{B}^{B}\bar{f}_{P_i}+\boldsymbol{z}_{B}^{B}\bar{\tau}_{P_i}\right)+\boldsymbol{\tau}_{d}^{B} \tag{3.17}$$

with $\bar{f}_{P_i}$ and $\bar{\tau}_{P_i}$ solved from Section 2.1, and $\boldsymbol{\tau}_{d}^{B}$ from Section 2.2.

There are then seven constraints for the N_p+6 unknowns, leaving N_p-1 degrees of freedom for the hover solution. These degrees of freedom may be determined, for example, by finding the hover solution that requires a minimum of input power, by symmetry considerations, or by considerations related to the controllability of the system.

3.2 Position trajectory

The vehicle's acceleration in hover will either be constant, or be periodic with period T_{hvr}. Specifically, if the vehicle's thrust axis $\boldsymbol{z}_B$ is not parallel to the hover angular velocity $\bar{\boldsymbol{\omega}}_{BE}$, a component of the total thrust must point perpendicular to gravity. Because the vehicle has zero vertical acceleration, the component of the thrust in the direction of the weight must equal the vehicle's weight. The horizontal acceleration component $\bar{a}_h$ may then be calculated as:

$$(m\bar{a}_h)^2+(m\left\|\boldsymbol{g}\right\|)^2=\left(\sum_{i=1}^{N_p}\bar{f}_{P_i}\right)^2. \tag{3.18}$$

This can be rewritten as a centripetal acceleration of radius $\bar{r}_{\text{hvr}}$ at vehicle's angular velocity. Substituting the hover forces from (3.15), and simplifying yields:

$$\bar{r}_{\text{hvr}}=\frac{\left\|\boldsymbol{g}\right\|}{\left\|\bar{\boldsymbol{\omega}}_{BE}\right\|^2}\sqrt{\left(\frac{\left\|\bar{\boldsymbol{\omega}}_{BE}\right\|}{\boldsymbol{z}_B\cdot\bar{\boldsymbol{\omega}}_{BE}}\right)^2-1}. \tag{3.19}$$

The vehicle will thus move along a horizontal circle of this radius at the hover solution.

4. Position and attitude control

A given hover solution is only useful if the vehicle can enter the solution, and maintain it in the face of disturbances. In this section an approach is given to investigate the

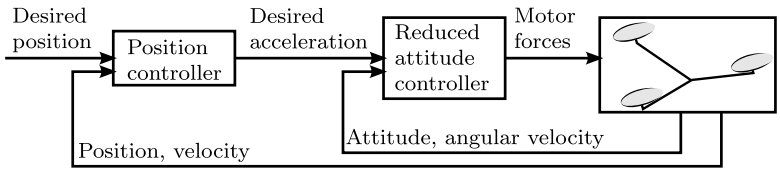

Figure 3.2. An example cascaded control strategy, where an outer position controller defines a desired acceleration, and an inner controller controls the vehicle's attitude so that the acceleration is achieved.

controllability of the vehicle about a hover solution. The problem is broken into two sections for ease of analysis: first it is argued that attitude control under constant total thrust is sufficient for position control. An approach is then given for determining the controllability of the attitude system, where the problem is reduced to investigating the controllability of a linear time invariant system with 5 states and $N_P - 1$ inputs. An example cascaded controller is shown in Fig. 3.2.

4.1 Position control

If both the angular velocity and the total thrust force are constant over one rotation, the average acceleration $\tilde{\boldsymbol{a}}(t)$ of the vehicle may be defined, similar to what was done in Section 3.1, by averaging $\ddot{\boldsymbol{d}}$ over one rotation. The goal is then to control this average acceleration, and thereby the position of the vehicle.

Rather than directly finding an equation for $\tilde{\boldsymbol{a}}(t)$, the exposition can be simplified by introducing two unit vectors. A constant body-fixed vector $\boldsymbol{n}$ is introduced, which lies parallel to the angular velocity of the vehicle in hover, and is oriented so that it has a positive component in the direction $\boldsymbol{z}_B$. This vector may be thought of as the vehicle's thrust direction, averaged over one period of rotation.

The second unit vector $\boldsymbol{n}_{\text{des}}$ is defined through the vehicle's desired acceleration. Introducing the short-hand $f_\Sigma(t)$ for the total thrust force, the vehicle's acceleration averaged over one rotation $\tilde{\boldsymbol{a}}_{\text{des}}(t)$ may be written as below, from which $\boldsymbol{n}_{\text{des}}(t)$ and $f_\Sigma(t)$ may be solved:

$$\tilde{\boldsymbol{a}}_{\text{des}}(t) = \left(\frac{\boldsymbol{z}_B \cdot \boldsymbol{n}}{m} f_\Sigma(t)\right) \boldsymbol{n}_{\text{des}}(t) + \boldsymbol{g}. \tag{3.20}$$

Note that the factor $(\boldsymbol{z}_B \cdot \boldsymbol{n}/m)$ is constant. These vectors are visualised in Fig. 3.3.

Therefore, if the body-fixed vector $\boldsymbol{n}$ can be controlled to point along some given $\boldsymbol{n}_{\text{des}}$ while the vehicle produces a total thrust force f_Σ, the vehicle's average acceleration will track the desired, and the vehicle's position can be controlled.

4.2 Reduced attitude control

Only the two attitude degrees of freedom relevant to the vehicle's translational motion are controlled, i.e. the unit vector $\boldsymbol{n}$. For convenience a control coordinate frame C is

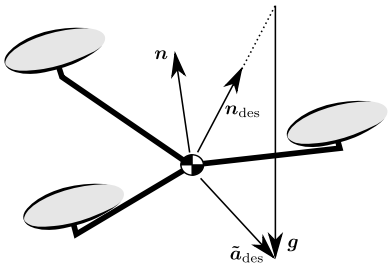

Figure 3.3. The relationships of the vectors introduced for the position control, where the desired acceleration vector $\tilde{\boldsymbol{a}}_{\mathrm{des}}$ is computed by the position controller, and from this follows the desired normal $\boldsymbol{n}_{\mathrm{des}}$ through (3.20). The goal of the attitude controller is then to rotate the body-fixed vector $\boldsymbol{n}$ into $\boldsymbol{n}_{\mathrm{des}}$.

introduced which is fixed with respect to the body-fixed frame B and where

$$\boldsymbol{n}^C = \boldsymbol{C}^{CB}\boldsymbol{n}^B = (0,0,1)\,. \tag{3.21}$$

The remaining degree of freedom of $\boldsymbol{C}^{CB}$ (that is, a rotation about $\boldsymbol{n}$) may be chosen arbitrarily. The components α_i of the angular velocity are introduced, so that

$$\boldsymbol{\omega}_{BE}^C = \boldsymbol{C}^{CB}\boldsymbol{\omega}_{BE}^B = (\alpha_1,\alpha_2,\alpha_3)\,. \tag{3.22}$$

The time derivative of $\boldsymbol{n}_{\mathrm{des}}^C$ follows from the product rule and (3.2):

$$\dot{\boldsymbol{n}}_{\mathrm{des}}^C = -[\![\boldsymbol{\omega}_{BE}^C]\!]\boldsymbol{n}_{\mathrm{des}}^C + \boldsymbol{C}^{CE}\dot{\boldsymbol{n}}_{\mathrm{des}}^E. \tag{3.23}$$

The change in direction of the desired acceleration vector from (3.20) is captured by $\dot{\boldsymbol{n}}_{\mathrm{des}}^E$.

The components η_i are introduced, such that $\boldsymbol{n}_{\mathrm{des}}^C = (\eta_1,\eta_2,\eta_3)$. The attitude control goal is then to drive $\boldsymbol{n}_{\mathrm{des}}^C$ to $(0,0,1)$, and drive $\dot{\boldsymbol{n}}_{\mathrm{des}}^C$ to zero. This can be formalised by introducing the state deviation

$$\xi = (\eta_1,\eta_2,\alpha_1,\alpha_2,\alpha_3) - (0,0,0,0,\boldsymbol{n}\cdot\bar{\boldsymbol{\omega}}_{BE})\,. \tag{3.24}$$

If ξ is stabilisable for some given f_Σ, the vehicle is stabilisable about the desired acceleration. [1]

1) Selecting input variables Because the total thrust is fixed at f_Σ, there are $N_p - 1$ free input variables remaining, which may be used to control the attitude. These may

[1] A stabilisable system is one which can be made asymptotically stable by appropriate state feedback [3.35], i.e. any uncontrollable modes are asymptotically stable.

be specified by introducing the input vector u, and then specifying the motor forces as below:

$$f_{P_i} = \bar{f}_{P_i} + u_i, \text{ for } i \in \{1, \ldots, N_p - 1\} \tag{3.25}$$

$$f_{P_{N_p}} = \bar{f}_{P_{N_p}} - \sum_{i=1}^{N_p - 1} u_i. \tag{3.26}$$

The propeller speeds (instead of thrusts) may alternatively be used as inputs, through the relationships of Section 2.1. The inputs may also be based on symmetry properties of a specific vehicle, so that e.g. independent torques are used as inputs, from which the required propeller speeds are recovered (as is commonly done when controlling quadrocopters, e.g. [3.33]).

2) Linearised system The time derivative $\dot{\xi}$ follows from (3.3) and (3.23). For the analysis it will be assumed that $\dot{\boldsymbol{n}}^E_{\text{des}} = 0$, i.e. that the desired average vehicle acceleration in (3.20) is constant. In practise, a cascaded control strategy as shown in Fig. 3.2 may be employed, with an outer position controller computing $\boldsymbol{n}_{\text{des}}$. If the dynamics for this outer controller are sufficiently slow, the term $\dot{\boldsymbol{n}}^E_{\text{des}}$ in (3.23) may be neglected.

The effect of the angular acceleration of the propellers with respect to the body, $\dot{\boldsymbol{\omega}}^B_{P_iB}$, may be included through a model of the motor dynamics. Alternatively, if the angular inertia of the propellers $\boldsymbol{I}^B_{P_i}$ is sufficiently small, this term may be neglected.

Linearising about the hover solution yields a system as below:

$$\dot{\xi} \approx \begin{bmatrix} 0 & \pm\|\bar{\boldsymbol{\omega}}_{BE}\| & 0 & -1 & 0 \\ \mp\|\bar{\boldsymbol{\omega}}_{BE}\| & 0 & 1 & 0 & 0 \\ 0 & 0 & a_{11} & a_{12} & a_{13} \\ 0 & 0 & a_{21} & a_{22} & a_{23} \\ 0 & 0 & a_{31} & a_{32} & a_{33} \end{bmatrix} \xi + Bu \tag{3.27}$$

where a_{ij}, B and u depend on the specific vehicle configuration, and follow in part from (3.3).

The requirement that (3.27) be stabilisable adds an additional criterion to the search for suitable hover solutions for a vehicle, in addition to the seven algebraic constraints derived in Section 3. Furthermore, (3.27) may serve as a basis for linear time invariant controller design: in Section 5 controllers are designed for a quadrocopter experiencing propeller failures, and in Section 7 controllers are designed for novel vehicles.

5. Quadrocopter actuator failsafe

Quadrocopters are popular as research testbeds, toys, and sensor platforms (e.g. for aerial photography). Compared to hexa- or octocopters, which have six or eight propellers, quadrocopters are mechanically simpler and have to carry less structural mass (which in turn may make them more efficient). However, they do not offer an obvious hardware redundancy if a propeller fails (that is, they cannot hover at zero angular velocity after losing a propeller).

In this section it is shown that quadrocopters can maintain a hover as defined in Section 3, after the complete loss of a propeller. In fact, under some restrictions on the vehicle's mass distribution and aerodynamic properties, the vehicle remains controllable after the loss of any number of propellers, as long as a single propeller remains operable.

Five different quadrocopter failure scenarios are considered:

- no failure,
- failure of a single propeller,
- failure of two opposing propellers,
- failure of two adjacent propellers, and
- failure of three propellers.

The hover solutions will first be presented for each, followed by controller design, and then each scenario is validated in a non-linear simulation, For losing a single propeller, and losing two opposing propellers, the simulation is validated by experiment. Extension 1 shows a video of a quadrocopter in flight after the loss of a single, and two opposing propellers.

This section focuses on controlled flight near hover in a controlled environment, rather than discussing practical implementation issues related to using these hover solutions on an actual quadrocopter in the field. For fault detection strategies, the reader is referred to the references in the introduction. Low-cost and robust sensing/estimation strategies for rapidly rotating vehicles and robust transitions from one flight mode to another may be promising areas of future research.

5.1 Platform

The work is validated using quadrocopters based on the Ascending Technologies Hummingbird [3.36], in the Flying Machine Arena [3.37], which will be referred to here simply as "the quadrocopter". Such a quadrocopter is shown in Fig. 3.4.

The vehicle's mass was measured to be 0.50 kg, and the distance from the centre of mass to the centre of the propellers is 0.17 m. The quadrocopter's inertia was determined from a CAD model, and validated by measuring its period of oscillation when suspended

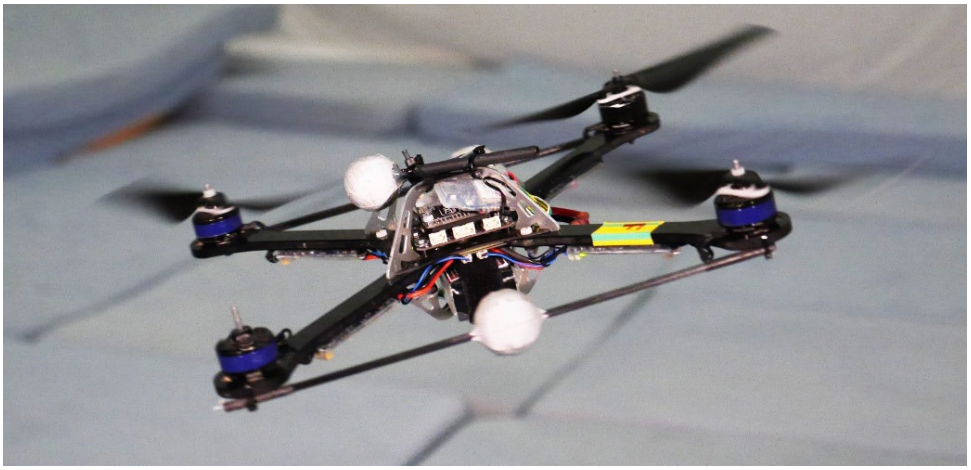

Figure 3.4. The quadrocopter used for the experiments, maintaining controlled flight despite the complete loss of one propeller.

around three different axes. The inertia matrix was estimated to be

$$\boldsymbol{I}_B^B = \mathrm{diag}\,(2.7, 2.7, 5.2) \times 10^{-3}\mathrm{kgm}^2. \tag{3.28}$$

The propeller inertia about its axis of rotation was estimated by approximating the propeller and motor rotor as disks and cylinders, respectively. The remainder of the inertia matrix is neglected, so that the propeller's inertia is constant when expressed in the body-fixed frame:

$$\boldsymbol{I}_P^B = \mathrm{diag}\,(0, 0, 1.5) \times 10^{-5}\mathrm{kgm}^2. \tag{3.29}$$

The propellers used were characterised using a force-torque sensor, and the thrust and reaction torque coefficients were estimated as $|\kappa_{f_i}| = 6.41 \times 10^{-6}\mathrm{N\,s^2\,rad^{-2}}$ and $\kappa_{\tau_i} = 1.1 \times 10^{-7}\mathrm{N\,m\,s^2\,rad^{-2}}$, respectively. On a static test bench, the propellers are able to produce thrust forces in the range of 0.2 N to 3.8 N.

The aerodynamic drag torque acting on the body was estimated to be diagonal in the body-fixed frame, with entries experimentally identified as below.

$$\boldsymbol{K}_d^B = \mathrm{diag}\,(0.7, 0.7, 1.4) \times 10^{-4}\mathrm{Nms^2/rad^2} \tag{3.30}$$

5.2 Hover solutions

Two short-hand notations will be used in this section: the total mechanical power consumed by the propellers will be expressed as $P_\Sigma = P_{P_1} + P_{P_2} + P_{P_3} + P_{P_4}$. The propeller speeds Ω_{P_i} will be collected into the vector $\Omega = \{-\Omega_{P_1}, \Omega_{P_2}, -\Omega_{P_3}, \Omega_{P_4}\}$, where the signs of the entries are changed so that all entries are positive if the propellers spin in their intended directions.

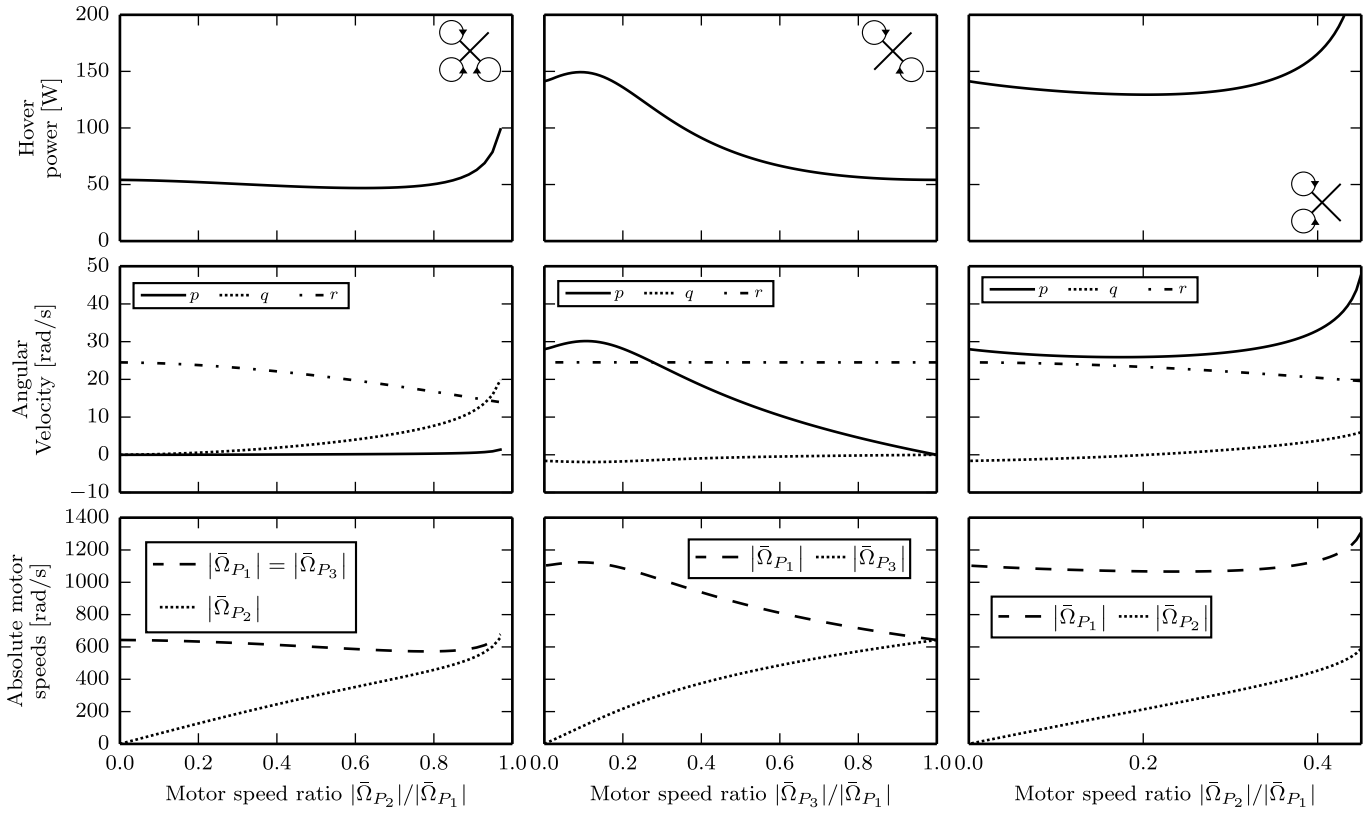

Figure 3.5. Hover solutions for different failure scenarios, from left to right: failure of a single propeller, failure of two opposing propellers, and failure of two adjacent propellers. The graphs depict how the hover solutions vary as a function of the ratio of angular velocities of the remaining motors. For the single failure, solutions may also be found where the odd propeller turns faster than the two propellers with the same handedness – these are not shown as they consume much more power than the solutions shown. For both scenarios of two propellers failing only half of the possible solutions are shown with the remainder following by symmetry – e.g. for two opposing propellers failing, hover solutions exist with $\bar{\Omega}_3/\bar{\Omega}_1 > 1$. The angular velocity is plotted as expressed in the body-fixed coordinate system, where $\bar{\boldsymbol{\omega}}_{BE}^{B} = (p, q, r)$. Note that the propeller speeds shown are absolute.

All configurations with more than one remaining functional propeller have at least one degree of freedom in selecting a hover solution. In such cases, numerical optimisation is used to find the hover solution that uses least mechanical power, as computed in Section 1. Depending on the application, minimising other variables may be more useful, e.g. minimising the vehicle's total rotation rate, or maximising the minimum distance from the equilibrium propeller speeds to the actuator limits.

1) No failure For the quadrocopter with no propellers having failed there are 3 degrees of freedom for finding a hover solution. As expected, the minimum power solution is to

have all propellers turning at angular velocities of equal magnitude, and specifically

$$\bar{\Omega} = (438, 438, 438, 438)\,\mathrm{rad\,s^{-1}} \tag{3.31}$$
$$\bar{\boldsymbol{\omega}}_{BE}^{B} = (0, 0, 0)\,\mathrm{rad\,s^{-1}} \tag{3.32}$$
$$\bar{P}_{\Sigma} = 36.9\,\mathrm{W} \tag{3.33}$$
$$\bar{r}_{\mathrm{hvr}} = 0\,\mathrm{mm}. \tag{3.34}$$

2) Failure of a single propeller Without loss of generality it will be assumed that propeller 4 has failed, that is the propeller pointing to the right if looking at the quadrocopter from the top. In this case the constraint $\bar{\Omega}_4 = 0$ is added and two degrees of freedom remain for the hover solution. Through numerical optimisation the following solution was found to consume least power in hover:

$$\bar{\Omega} = (585, 362, 585, 0)\,\mathrm{rad\,s^{-1}} \tag{3.35}$$
$$\bar{\boldsymbol{\omega}}_{BE}^{B} = (0.2, 4.3, 19.5)\,\mathrm{rad\,s^{-1}} \tag{3.36}$$
$$\bar{P}_{\Sigma} = 46.8\,\mathrm{W} \tag{3.37}$$
$$\bar{r}_{\mathrm{hvr}} = 6\,\mathrm{mm}. \tag{3.38}$$

Some intuition for this may be gained by constraining $\bar{\Omega}_1 = \bar{\Omega}_3$, and performing a line search over the ratio $\bar{\Omega}_2/\bar{\Omega}_1$. This is illustrated in Fig. 3.5.

3) Failure of two opposing propellers Here it will be assumed that the left and right hand side propellers have failed, so that $\bar{\Omega}_4 = \bar{\Omega}_2 = 0$. Now there is only one degree of freedom for the hover solution, which may be explored with a line search over the ratio of the remaining propeller speeds: $\bar{\Omega}_3/\bar{\Omega}_1$. This line search is illustrated in Fig. 3.5. The least power solution is for $\boldsymbol{z}_B$ to point opposite to gravity, with the two remaining propellers rotating at equal speeds:

$$\bar{\Omega} = (643, 0, 643, 0)\,\mathrm{rad\,s^{-1}} \tag{3.39}$$
$$\bar{\boldsymbol{\omega}}_{BE}^{B} = (0, 0, 24.5)\,\mathrm{rad\,s^{-1}} \tag{3.40}$$
$$\bar{P}_{\Sigma} = 54.1\,\mathrm{W} \tag{3.41}$$
$$\bar{r}_{\mathrm{hvr}} = 0\,\mathrm{mm}. \tag{3.42}$$

4) Failure of two adjacent propellers If both the right hand side and rear propellers have failed, the two added constraints are $\bar{\Omega}_4 = \bar{\Omega}_3 = 0$. Again, only one degree of freedom remains in the hover solution, which is also explored with a line search in Fig. 3.5. In this case, no hover solution exists for the two remaining propellers rotating at equal speed. Thus, it will be assumed that the magnitude of propeller 1's velocity is larger than that

of propeller 2. The least power solution is then:

$$\bar{\Omega} = (1067, 218, 0, 0)\,\mathrm{rad\,s^{-1}} \tag{3.43}$$
$$\bar{\boldsymbol{\omega}}_{BE}^{B} = (26.0, 0, 23.3)\,\mathrm{rad\,s^{-1}} \tag{3.44}$$
$$\bar{P}_{\Sigma} = 129\,\mathrm{W} \tag{3.45}$$
$$\bar{r}_{\mathrm{hvr}} = 9\,\mathrm{mm}. \tag{3.46}$$

Note that a symmetric solution exists for $\left|\bar{\Omega}_{P_2}\right| > \left|\bar{\Omega}_{P_1}\right|$, consuming the same power, but where the vehicle rotates in the opposite direction.

5) Failure of three propellers If all but one propeller have failed, there are no degrees of freedom remaining in the hover solution. It will be assumed that only the front propeller remains, and the hover solution is then

$$\bar{\Omega} = (1103, 0, 0, 0)\,\mathrm{rad\,s^{-1}} \tag{3.47}$$
$$\bar{\boldsymbol{\omega}}_{BE}^{B} = (28.0, -1.6, 24.5)\,\mathrm{rad\,s^{-1}} \tag{3.48}$$
$$\bar{P}_{\Sigma} = 141\,\mathrm{W} \tag{3.49}$$
$$\bar{r}_{\mathrm{hvr}} = 8\,\mathrm{mm}. \tag{3.50}$$

Note that it is unlikely that any practical quadrocopter has sufficient excess thrust that a single propeller is able to produce the force required.

5.3 Control and validation

The controllability of the vehicle about the hover solution is investigated for each of the five different failure cases presented above. If at least two propellers remain the method of Section 4 was used directly. For the case of losing three propellers (and thus only a single propeller remaining), the method was modified to use the single remaining force as input. In this case, the total force cannot be controlled independently of the attitude. The layout of the controller is shown in Fig. 3.6, showing the arrangement of the cascaded position and attitude controllers. The controllers are described in more detail below.

1) Position control A position controller was created which calculates a desired acceleration based on the vehicle's current position and velocity, such that the vehicle's position behaves as a damped second order system [3.38]. This desired acceleration can then be substituted for the desired average acceleration in (3.20), and be transformed into a desired vehicle direction $\boldsymbol{n}_{\mathrm{des}}$ and total thrust f_{Σ}. The second order system's natural frequency and damping ratio were set to $1.5\,\mathrm{rad\,s^{-1}}$ and 0.7, respectively.

2) Attitude control Given a linearised attitude system of the form (3.27), a linear quadratic regulator (LQR) controller [3.39] was designed for the attitude system. The

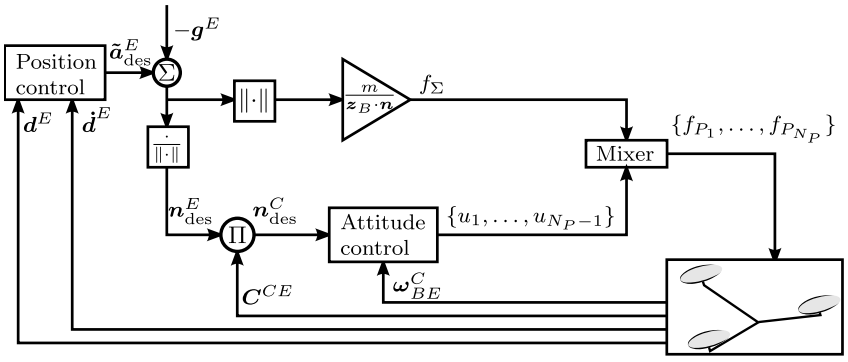

Figure 3.6. The feedback structure used for the experiments. The position controller computes a desired average acceleration $\tilde{\boldsymbol{a}}^E_{\text{des}}$, which is then converted into a desired total thrust f_Σ and direction $\boldsymbol{n}^E_{\text{des}}$ according to (3.20). The attitude controller's outputs u_i are combined with the desired total thrust force to compute the desired individual propeller forces as in (3.25)-(3.26).

same LQR weights are used for each failure scenario. The cost on deviations from the desired normal was set to 20, and the cost on the angular rates set to zero. The input cost was set to the identity matrix of the appropriate size, with units N^{-2}.

The five dimensional attitude subsystem (3.27) is stabilisable for each scenario, and is controllable in each case except for two opposing propellers failing. In this case, for the given hover solution, the vehicle's angular velocity about $\boldsymbol{z}_B$ is uncontrollable. Note that this may be avoided by selecting a different hover condition – this would however increase the hover power consumption and does not actually affect the vehicle's hover performance.

3) Simulation results The control strategy is validated in a nonlinear simulation for each of the different failure scenarios. No sensor noise or external disturbances are simulated. The actuators are simulated to have additional dynamics, so that the propeller speeds respond as a first order system with time constant 15 ms.

A comparison of the flight performance for each failure scenario in simulation is shown in Fig. 3.7. The data starts with the vehicle at the hover attitude, but with a 1 m horizontal position offset. In each scenario, the vehicle is able to stabilise its position at the origin.

5.4 Experimental validation

The simulation is validated by comparing it to data collected from flight experiments in the Flying Machine Arena. The vehicle's state is estimated using position and orientation measurements from a motion capture system, and inertial sensors on the vehicle. A video of such experiments can be found in Extension 1. These experiments are only possible for the scenarios of no failure, failure of a single propeller, and failure of two opposing propellers – for the remaining cases (two adjacent propellers failing or three propellers failing) the quadrocopter of Section 5.1 can not produce sufficient thrust to compensate for the vehicle's weight. The data for a single propeller failure are compared in Fig. 3.8, and for the failure of two opposing propellers in Fig. 3.9.

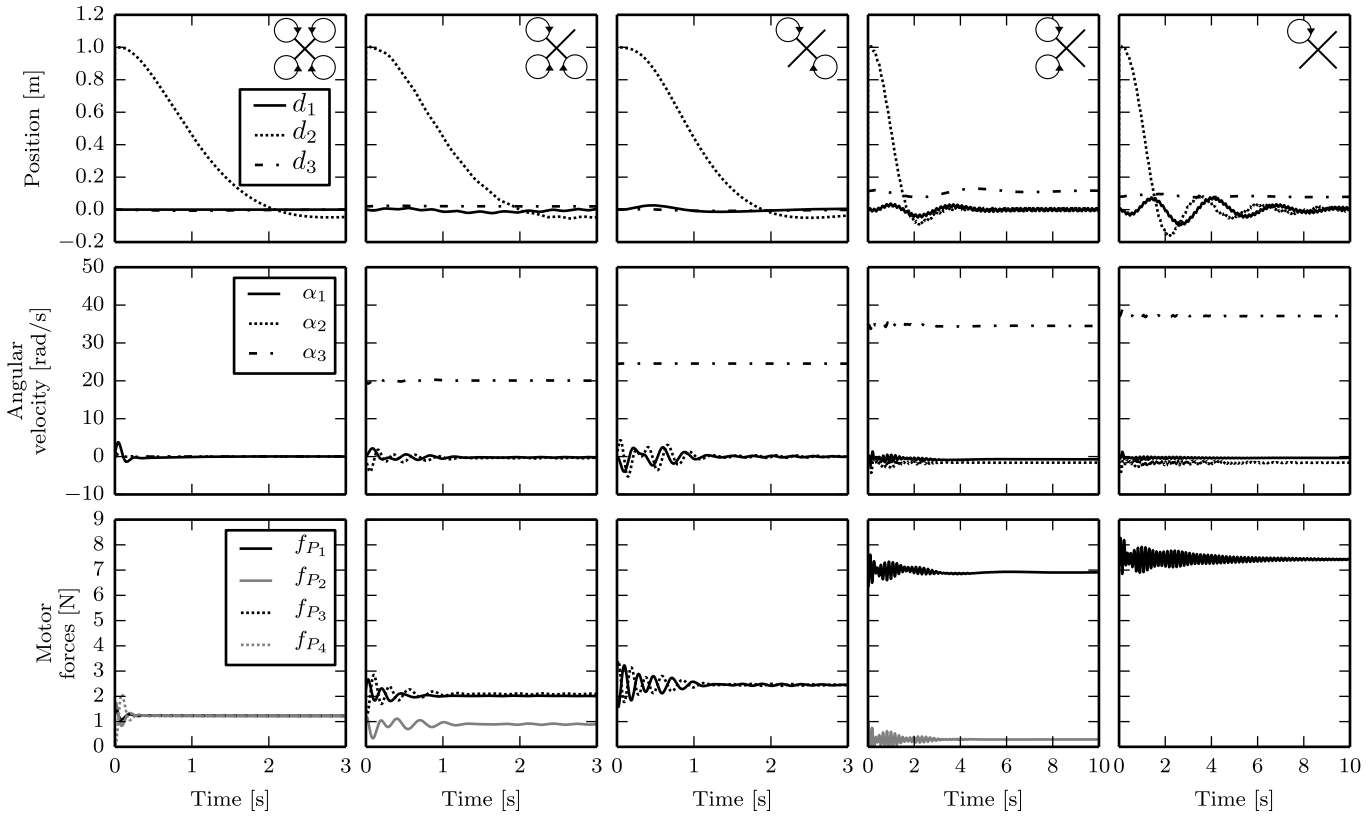

Figure 3.7. Simulation results for a quadrocopter recovering from a 1 m horizontal position error for different failure scenarios. Each column represents a different failure condition, from left to right: no failure (nominal operation), failure of a single propeller, failure of two opposing propellers, failure of two adjacent propellers, failure of three propellers. Note that the angular velocity is expressed in the control frame C, so that two components must be zero at equilibrium. The position is expressed as $\boldsymbol{d}^E = (d_1, d_2, d_3)$, with d_3 pointing opposite to gravity. The simulation is validated by experiment in Section 5.4. Each row shares its legend. The steady-state vertical offset for the failure of two adjacent propellers and the failure of three propellers is due to a steady state attitude error, as can also be seen in the corresponding steady-state errors of the angular velocity. This error could be explained by interaction between the attitude controller and the position control, and higher-order dynamic effects not compensated for by the linear controller strategy. The steady-state position offsets may be readily compensated for, e.g. by adding an integral term to the position control.

For a single failure the experimental data differs somewhat from the simulated data, most notably for the steady-state angular velocity. In experiment, the vehicle has an angular velocity component perpendicular to the $\boldsymbol{n}$ axis of approximately $4.1\,\mathrm{rad\,s^{-1}}$. The angular velocity component about $\boldsymbol{n}$ is also lower in experiment than expected from the simulation ($15.8\,\mathrm{rad\,s^{-1}}$ compared to $19.9\,\mathrm{rad\,s^{-1}}$). This discrepancy may be due to additional torques acting on the vehicle, specifically torques generated by the translation of the centres of the propellers. Further characterisation of the aerodynamic effects acting on the vehicle at high rotational velocities represent a possibly interesting area of future research.

Furthermore, the vehicle has an approximately 0.8 m vertical offset in experiment for

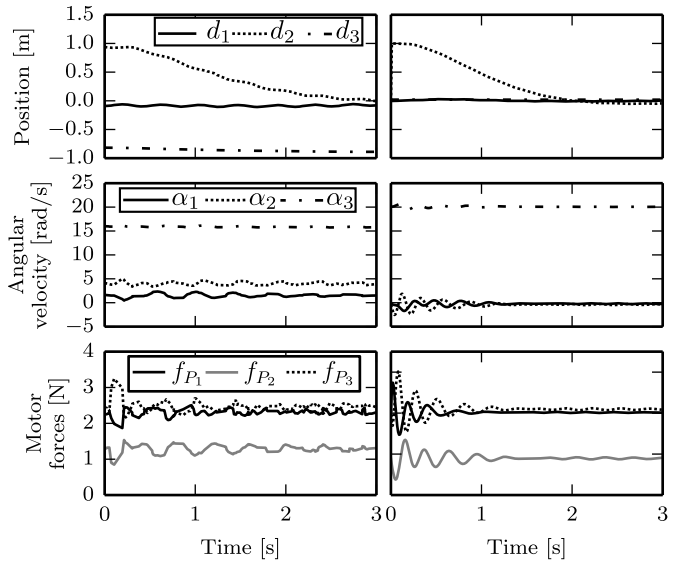

Figure 3.8. Comparison of experimental (left column) and simulated (right column) data for a quadrocopter with a single failed propeller, and an initial 1 m horizontal position error. The angular velocity is as expressed in the control frame C, and the position is expressed as $\boldsymbol{d}^E = (d_1, d_2, d_3)$. The steady-state vertical position offset of approximately 0.8 m in the experiment is discussed in the text.

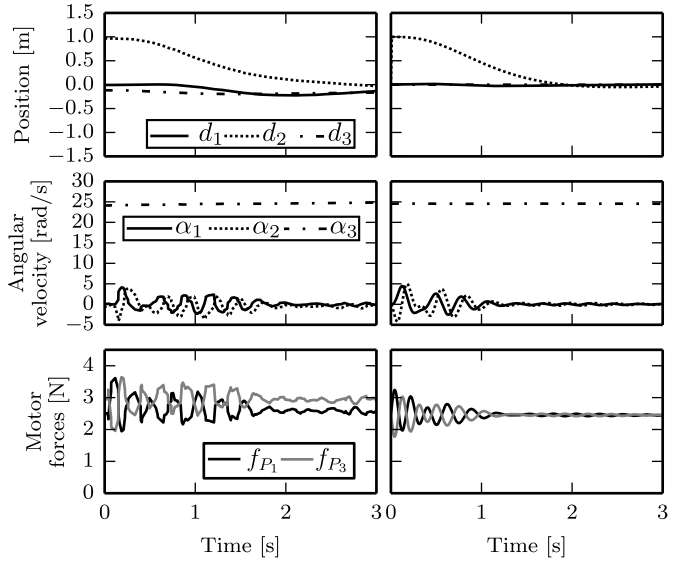

Figure 3.9. Comparison of experimental (left column) and simulated (right column) data for a quadrocopter with two opposing failed propellers, and an initial 1 m horizontal position error. The angular velocity is as expressed in the control frame C, and the position is expressed as $\boldsymbol{d}^E = (d_1, d_2, d_3)$.

a single failure. This may be partially explained by the angular velocity offset of the experiments w.r.t. the simulation – in the experiment, the quadrocopter's thrust axis is tilted farther away from gravity, and thus the feed-forward thrust to compensate for the vehicle's weight from (3.20) will be too low. This could be readily compensated by e.g.

adding an integral control term to the position controller, or by refining the model to allow for better prediction of the vehicle's angular velocity equilibrium. Nonetheless, the controller successfully stabilises the vehicle about the hover solution, and the vehicle's horizontal error is controlled to zero as expected.

For the case of two failed opposing propellers, the simulation matches the experimental data well. It is notable that the two motors do not produce equal force in experiment after the transients have died away – this may be explained by a centre of mass offset of the experimental vehicle, or an imperfection in one of the two actuators.

5.5 Special cases for controllability

Some intuition for when it is possible to control a quadrocopter having lost one or more propellers may be gained by analysing the hover conditions and controllability requirements of Sections 3 and 4 under some simplifying assumptions, especially for the case of losing two opposing propellers.

Specifically, it will be assumed that the vehicle's mass moment of inertia expressed in the body frame is diagonal, with only two unique entries. The inertia of the propeller is assumed to be zero except about its axis of rotation:

$$\boldsymbol{I}_B^B = \operatorname{diag}\left(I_{B,xx}, I_{B,xx}, I_{B,zz}\right) \tag{3.51}$$

$$\boldsymbol{I}_P^B = \operatorname{diag}\left(0, 0, I_{P,zz}\right) \tag{3.52}$$

where furthermore $I_{P,zz} \ll I_{B,zz}$. The propellers are assumed identical except for their handedness, so that

$$\kappa_{f_i} = (-1)^i \kappa_f \tag{3.53}$$

$$\kappa_{\tau_i} = \kappa_\tau. \tag{3.54}$$

It is assumed that the vehicle's angular velocity with respect to the inertial frame is much smaller than the propellers' angular velocity with respect to the body, i.e $\|\boldsymbol{\omega}_{BE}\| \ll \|\boldsymbol{\omega}_{P_iB}\|$ for each remaining propeller i. The propellers are mounted at a distance l from the vehicle's centre of mass.

The vehicle's drag torque coefficient is given below, again with only two unique entries:

$$\boldsymbol{K}_d^B = \operatorname{diag}\left(K_{d,xx}, K_{d,xx}, K_{d,zz}\right). \tag{3.55}$$

Defining the components of the vehicle angular velocity as $\boldsymbol{\omega}_{BE}^B = (p, q, r)$, and applying

the simplifications, it follows that:

$$\begin{aligned}\dot{p}I_{B,xx} = &\left(I_{B,xx} - I_{B,zz}\right) qr - K_{d,xx} p \left\|\boldsymbol{\omega}_{BE}\right\| + \\ &l\kappa_f \left(\Omega_2^2 - \Omega_4^2\right) - qI_{P,zz} \left(\Omega_1 + \Omega_2 + \Omega_3 + \Omega_4\right)\end{aligned} \tag{3.56}$$

$$\begin{aligned}\dot{q}I_{B,xx} = &\left(-I_{B,xx} + I_{B,zz}\right) pr - K_{d,xx} q \left\|\boldsymbol{\omega}_{BE}\right\| + \\ &l\kappa_f \left(\Omega_1^2 - \Omega_3^2\right) + pI_{P,zz} \left(\Omega_1 + \Omega_2 + \Omega_3 + \Omega_4\right)\end{aligned} \tag{3.57}$$

$$\begin{aligned}\dot{r}I_{B,zz} = &-K_{d,zz} r \left\|\boldsymbol{\omega}_{BE}\right\| - \\ &\kappa_\tau \left(\left|\Omega_1\right| \Omega_1 + \left|\Omega_2\right| \Omega_2 + \left|\Omega_3\right| \Omega_3 + \left|\Omega_4\right| \Omega_4\right).\end{aligned} \tag{3.58}$$

Considering specifically the loss of two opposing motors, such that $\Omega_2 = \Omega_4 = 0$, and constraining the hover solution so that $\bar{\Omega}_1 = \bar{\Omega}_3$, the hover solution follows as

$$\bar{p} = \bar{q} = 0 \tag{3.59}$$

$$\bar{r} = \sqrt{\frac{\kappa_\tau m \left\|\boldsymbol{g}\right\|}{\kappa_f K_{d,zz}}} \tag{3.60}$$

$$\bar{\Omega}_1 = \bar{\Omega}_3 = -\sqrt{\frac{m \left\|\boldsymbol{g}\right\|}{2\kappa_f}}. \tag{3.61}$$

The attitude controllability can now be analysed as described in Section 4, by introducing the scalar attitude input u and requiring that the total force remain constant; it follows that $u = \left(f_{P_1} - f_{P_3}\right)/2$. The control frame C is selected to coincide with the body frame B. The entries a_{ij} of the linearised dynamics matrix (3.27) and the input matrix B are then as follows:

$$a_{11} = a_{22} = -\frac{K_{d,xx}\bar{r}}{I_{B,xx}} \tag{3.62}$$

$$a_{33} = -\frac{2K_{d,zz}\bar{r}}{I_{B,zz}} \tag{3.63}$$

$$a_{12} = -a_{21} = \frac{\sqrt{2m \left\|\boldsymbol{g}\right\|} I_{P,zz}}{\sqrt{\kappa_f} I_{B,xx}} + \frac{\left(I_{B,xx} - I_{B,zz}\right)\bar{r}}{I_{B,xx}} \tag{3.64}$$

$$a_{13} = a_{23} = a_{31} = a_{32} = 0 \tag{3.65}$$

$$B = \frac{2l}{I_{B,xx}} \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T. \tag{3.66}$$

It is clear that r is an exponentially stable, uncontrollable mode of the linearised attitude system, and it will be neglected for the remainder of the analysis. A reduced, four state system remains with a single input. The controllability of this system may be determined by computing the determinant of the 4×4 controllability matrix of the

system and ensuring that this is non-zero [3.35].

After some tedious algebra it emerges that there are two plausible conditions where this system is uncontrollable. Firstly, if (3.67) holds, the cross-coupling in the attitude dynamics (3.3) disappears and the vehicle's roll rate p is uncontrollable:

$$I_{P,zz}\sqrt{2K_{d,zz}} = (I_{B,zz} - I_{B,xx})\sqrt{\kappa_\tau}. \tag{3.67}$$

If, instead, the following two conditions hold, the system is also uncontrollable:

$$K_{d,xx} = 0 \quad \text{and} \tag{3.68}$$

$$I_{P,zz}\sqrt{2K_{d,zz}} = (I_{B,zz} - 2I_{B,xx})\sqrt{\kappa_\tau}. \tag{3.69}$$

In this case there are two uncontrollable modes, which correspond to $p + a_{12}\eta_1$ and η_2.

These two conditions may be used to determine whether a quadrocopter's attitude will be close to uncontrollable with two opposing propellers, given its physical parameters. Unfortunately, the aerodynamic drag parameters $K_{d,xx}$ and $K_{d,zz}$ may be hard to accurately predict.

If only a single propeller has failed, the same hover solution may be selected as when two opposing propellers have failed with $\bar{p} = \bar{q} = 0$. Then the system has an additional input which directly actuates the roll rate p, and it will be controllable for all mass distributions.

1) Further simplifications The preceding results may be given a more intuitive geometric interpretation by further assuming that the propellers have zero inertia, i.e. $I_{P,zz} = 0$. The first case for being uncontrollable then simplifies from (3.67) to the requirement that $I_{B,xx} = I_{B,zz}$ (recall that by assumption also $I_{B,xx} = I_{B,yy}$). This happens if the vehicle's mass distribution is symmetric, i.e. similar to that of a sphere. The second case for being uncontrollable simplifies from (3.69) to $I_{B,zz} = 2I_{B,xx} = 2I_{B,yy}$ – this corresponds to the mass distribution of a two-dimensional object, e.g. a flat plate with the propellers' thrust axes pointing along the plate's normal.

6. Quadrocopter centre of mass offsets

If a quadrocopter's centre of mass is not located at its geometric centre the propeller forces will no longer all be equal when the vehicle hovers at zero angular velocity. Specifically, for the input torques of (3.3) to balance, propellers closer to the centre of mass must produce larger forces.

For large centre of mass offsets the required propeller forces may be close to (or even exceed) the propeller limits, leaving little or no control authority for feedback control. In

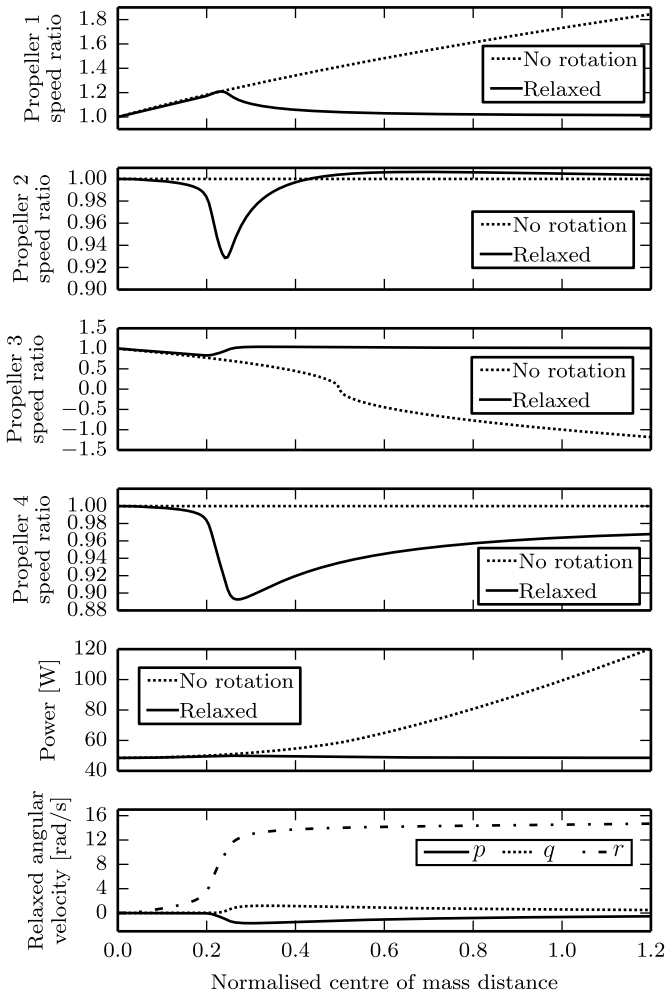

Figure 3.10. Hover solutions for a quadrocopter with an additional mass mounted at varying distances from the vehicle's geometric centre, showing the solution at which the vehicle has zero angular velocity, and a relaxed hover solution of Section 3 that minimizes power consumption. Note that the abscissa is the distance of the centre of mass to the geometric centre, normalised by the quadrocopter's arm length. Because of the mass ratio, the additional mass is placed six times farther from the propellers' geometric centre than the resulting centre of mass. The top four graphs plot for each propeller the ratio of each required propeller speed to that speed required if the centre of mass coincides with the geometric centre. The bottom graph shows the three components of the vehicle's angular velocity in the relaxed hover solution expressed in the body-fixed frame, where $\boldsymbol{\omega}_{BE}^{B} = (p, q, r)$.

such a situation, the approach of Sections 3-4 may be used to find a hover solution where the nominal motor forces are further away from the actuator limits.

As an extreme example: for a quadrocopter with propellers distributed on the corners of a square, if the centre of mass is located halfway between the geometric centre of the propellers and one propeller, the propeller farthest from the centre of mass must produce zero force for the input torques to balance. If the centre of mass moves further away from the geometric centre, the far propeller must produce a negative thrust. In this situation, the vehicle will consume significantly more power than if the centre of mass coincides

with the geometric centre. Furthermore, specific hardware may not be able to produce negative thrust.

If however the vehicle is allowed to rotate during hover, one may search e.g. for a hover solution that minimizes the required mechanical power, or maximizes the distance from the propellers' thrust in hover to the actuator saturation limits.

As an example, consider the quadrocopter of Section 5 carrying an additional load of 0.1 kg at some distance ρ from the geometric centre of the propellers. As the mass is shifted farther from the vehicle's geometric centre, the vehicle's centre of mass also moves farther from the geometric centre. The ratio of vehicle mass to the additional mass implies that the centre of mass is located at a distance $\rho/6$ from the geometric centre.

The mass moment of inertia of the vehicle with the added mass may be computed using Huygen's theorem [3.30]. For the simplicity of analysis, it is assumed that the vehicle's drag matrix $\boldsymbol{K}_d$ remains unchanged with the added mass and the resulting shifted centre of mass.

Fig. 3.10 compares the motor speeds and power requirements of the quadrocopter as the mass is moved farther away from the geometric centre, for both the conventional hover solution (with zero angular velocity) and the relaxed hover solution of Section 3. The mass is moved from the geometric center along the axis pointing towards propeller 1. For the conventional hover solution, the power required increases significantly as the mass is moved outwards, and at distances of $\rho > 0.51\,\text{m}$ the propeller farthest from the added mass must rotate in the direction opposite to the usual.

The figure also shows the results if the vehicle is allowed to rotate, and a search is done to find motor speeds that minimise the mechanical power in hover. The vehicle's mechanical power remains within 5% of the nominal power, even with displacements up to $\rho = 1.2\,\text{m}$. The motor speeds also remain close to the nominal speeds, even at large displacements. Note that for $\rho > 1.02\,\text{m}$ the centre of mass lies outside of the convex hull of the four propellers.

Thus, if the propeller's direction of rotation is fixed, a quadrocopter may be operated under a much wider range of centre of mass positions with the relaxed hover conditions compared to the restriction that the vehicle's angular velocity is zero at hover.

7. Novel vehicles

A novel class of hover-capable vehicles may also be designed using the approach of Section 3. Here, three examples are given, with novel vehicles having two, three, or four propellers arranged symmetrically about the vehicle's centre of mass. The propellers are mounted such that their thrust axes are parallel, and all propellers have the same handedness.

This family of vehicles is dubbed "spinners", allowing a prefix for the number of propellers. A quadspinner, trispinner, and bispinner are shown in Fig. 3.11, and Extension 2

Figure 3.11. A novel class of flying vehicles, from left to right: bispinner, trispinner, and quadspinner. For each vehicle, all propellers rotate in the same direction, causing the vehicle to rotate in the opposite direction in hover. Extension 2 contains a video showing each of these vehicles flying.

contains a video showing each vehicle in flight. Because of their symmetries, the hover equilibria are chosen such that all propellers have the same rotational speed.

Each vehicle is controlled similarly to the quadrocopter in Fig. 3.6, where the position and attitude control are separated. A desired attitude is generated by the position controller, which makes the position behave like a second order damped system. An LQR controller is created for the attitude, based on the first order model of Section 4.

The notation of Section 5.2 will be used to describe the hover solutions, and again so that all entries in $\bar{\Omega}$ are nominally positive: $\bar{\Omega} = \left\{-\Omega_{P_1}, \ldots, -\Omega_{P_{N_p}}\right\}$. The motors and propellers used are the same as those of Section 5.1.

7.1 Quadspinner

A normal quadrocopter may easily be converted into a quadspinner, by reversing the direction of rotation of two motors (and accordingly replacing the propellers). The quadspinner shown in Fig. 3.11 has the same physical properties as the quadrocopter presented in Section 5.1, except for the propellers' handedness.

It will then hover at the following condition:

$$\bar{\Omega} = (462, 462, 462, 462)\,\mathrm{rad\,s^{-1}} \tag{3.70}$$

$$\bar{\boldsymbol{\omega}}_{BE}^{B} = (0, 0, 24.2)\,\mathrm{rad\,s^{-1}} \tag{3.71}$$

$$\bar{P}_{\Sigma} = 38.9\,\mathrm{W}. \tag{3.72}$$

7.2 Trispinner

A three propeller vehicle was also created, with the propellers mounted at intervals of 120°. Conceptually, this trispinner is very similar to the quadspinner, as the propellers can produce a torque in any direction in the plane perpendicular to the thrust vector, for some given total thrust value.

Visually similar tri-rotor vehicles exist (see for example [3.40], [3.41]) that use a servo motor in addition to the three propellers, where the servo motor can rotate one of the vehicle's arms and thus vector the thrust of one propeller. These vehicles can then fully control their attitudes in addition to their total thrust. The trispinner, in comparison, does not attempt to control its orientation about the thrust axis, and has the advantage of being mechanically simpler.

The vehicle depicted in Fig. 3.11 has a mass of 0.44 kg, with the propellers mounted 0.17 m from the vehicle's centre of mass. The vehicle's inertia matrix and drag characteristics are then given below:

$$\boldsymbol{I}_B^B = \mathrm{diag}\,(2.4, 2.3, 4.4) \times 10^{-3}\mathrm{kg\,m^2} \tag{3.73}$$

$$\boldsymbol{K}_d^B = \mathrm{diag}\,(0.55, 0.55, 1.1) \times 10^{-4}\mathrm{Nms^2/rad^2} \tag{3.74}$$

and the hover condition is then

$$\bar{\Omega} = (498, 498, 498)\,\mathrm{rad\,s^{-1}} \tag{3.75}$$

$$\bar{\boldsymbol{\omega}}_{BE}^B = (0, 0, 26.1)\,\mathrm{rad\,s^{-1}} \tag{3.76}$$

$$\bar{P}_\Sigma = 36.6\,\mathrm{W}. \tag{3.77}$$

7.3 Bispinner

Finally, a bispinner was created, by removing two complete arms from a quadrocopter. The vehicle has a mass of 0.38 kg, and the inertia matrix and drag characteristics are as below:

$$\boldsymbol{I}_B^B = \begin{bmatrix} 270 & 65 & 0 \\ 65 & 3300 & 0 \\ 0 & 0 & 3400 \end{bmatrix} \times 10^{-6}\mathrm{kg\,m^2} \tag{3.78}$$

$$\boldsymbol{K}_d^B = \mathrm{diag}\,(0, 3.6, 7.2) \times 10^{-5}\mathrm{Nms^2/rad^2}. \tag{3.79}$$

Note that the vehicle's inertia about the axis connecting the motors is an order of magnitude smaller than the other terms on the diagonal. The off-diagonal terms follow as a result of the battery being positioned at an angle.

The hover condition for this vehicle is

$$\bar{\Omega} = (567, 567)\,\mathrm{rad\,s^{-1}} \tag{3.80}$$

$$\bar{\boldsymbol{\omega}}_{BE}^B = (0, -0.1, 29.7)\,\mathrm{rad\,s^{-1}} \tag{3.81}$$

$$\bar{P}_\Sigma = 36.1\,\mathrm{W}. \tag{3.82}$$

Because of its mass distribution, this vehicle has some unique dynamic properties

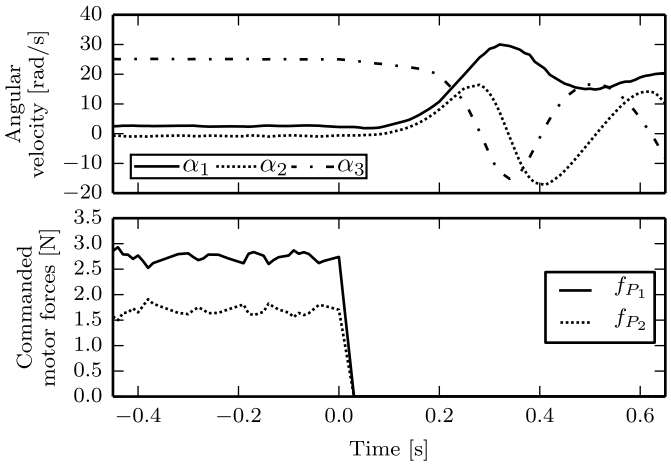

Figure 3.12. Experimental data for a bispinner at hover, where the motors (and thus attitude control) are switched off at time 0. The open loop exponential instability of the angular velocity is clearly visible, and angular accelerations up to $450\,\mathrm{rad\,s^{-2}}$ are measured by the vehicle's rate gyroscopes.

compared to the tri- and quadspinner. The matrices of the linearised attitude system, computed with the method of Section 4, are as below:

$$A = \begin{bmatrix} 0 & 29.7 & 0 & -1 & 0 \\ -29.7 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1.6 & 54.1 & -0.2 \\ 0 & 0 & 22.8 & -2.0 & 0 \\ 0 & 0 & 0 & 0 & -1.3 \end{bmatrix} \tag{3.83}$$

$$B = \begin{bmatrix} 0 & 0 & 24.9 & -103.5 & 0.3 \end{bmatrix}^T. \tag{3.84}$$

This linearised system has a single exponentially unstable mode, with eigenvalue at 35.2. This open loop instability can be clearly seen in Fig. 3.12.

8. Conclusion and outlook

This paper presents a broadened definition of hover for multicopters, where specifically the vehicle is not constrained to have zero angular velocity when hovering. The class of vehicles that can achieve these hover conditions are a superset of conventional multicopters.

By allowing a conventional multicopter to hover at non-zero angular velocity, it is shown that the vehicle can maintain controlled flight despite the complete loss of all but one propeller. This seems particularly relevant in the case of quadrocopters, as this negates one of the main arguments for using hexacopters or octocopters instead of quadrocopters. As such, the use of the proposed failsafe solutions may allow for a more widespread use

(and greater public acceptance of) quadrocopters. The extended hover solutions are also shown to allow a quadrocopter to maintain flight under large centre of mass disturbances.

Novel vehicles may also be conceived, such as the spinners presented herein. These vehicles could be used as novel hobbyist platforms or toys, or could be used as novel sensing platforms. For example, by mounting a line scanner to such a vehicle, and using the vehicle's angular velocity to sweep the sensor over the environment, a mechanically simple omnidirectional sensor can be created from a line scanner.

Acknowledgements

The Flying Machine Arena is the result of contributions of many people, a full list of which can be found at `www.flyingmachinearena.org`. The authors would like to thank Alex Wilkinson for his help designing the trispinner.

Funding

This research was supported by the Swiss National Science Foundation (SNSF), under grant application 138112.

A. Index to multimedia extensions

The multimedia extensions to this article are at: www.ijrr.org.

Ext.	Media type	Description	URL
1	Video	Quadrocopter failsafe flight experiments	`www.mwm.im/l/Relaxed1`
2	Video	Novel vehicles flight experiments	`www.mwm.im/l/Relaxed2`

References

[3.1] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor MAV", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2012, pp. 4557–4564.

[3.2] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: A compendium", *Journal of Field Robotics*, vol. 30, no. 5, pp. 803–831, 2013.

[3.3] W. Selby, P. Corke, and D. Rus, "Autonomous aerial navigation and tracking of marine animals", in *Proceedings of the 2011 Australasian Conference on Robotics and Automation*, Australian Robotics & Automation Association, 2011, pp. 1–7.

[3.4] J. Rasmussen, J. Nielsen, F. Garcia-Ruiz, S. Christensen, and J. C. Streibig, "Potential uses of small unmanned aircraft systems (UAS) in weed research", *Weed Research*, vol. 53, no. 4, pp. 242–248, 2013.

[3.5] J. Fink, N. Michael, S. Kim, and V. Kumar, "Planning and control for cooperative manipulation and transportation with aerial robots", *International Journal of Robotics Research*, vol. 30, no. 3, pp. 324–334, 2011.

[3.6] J. R. Hörandel, S. Buitink, A. Corstanje, J. E. Enriquez, and H. Falcke, "The LOFAR radio telescope as a cosmic ray detector", in *International Cosmic Ray Conference*, 2013.

[3.7] R. D'Andrea, "Can drones deliver?", *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 3, pp. 647–648, 2014.

[3.8] R. Voyles and G. Jiang, "Hexrotor UAV platform enabling dextrous interaction with structures - preliminary work", in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, IEEE, 2012, pp. 1–7.

[3.9] H. Efraim, A. Shapiro, and G. Weiss, "Quadrotor with a dihedral angle: On the effects of tilting the rotors inwards", *Journal of Intelligent & Robotic Systems*, pp. 1–12, 2015.

[3.10] D. Scaramuzza, M. Achtelik, L. Doitsidis, F. Friedrich, E. Kosmatopoulos, A. Martinelli, M. Achtelik, M. Chli, S. Chatzichristofis, L. Kneip, D. Gurdan, L. Heng, G. H. Lee, S. Lynen, M. Pollefeys, A. Renzaglia, R. Siegwart, J. Stumpf, P. Tanskanen, C. Troiani, S. Weiss, and L. Meier, "Vision-controlled micro flying robots: From system design to autonomous navigation and mapping in gps-denied environments", *IEEE Robotics Automation Magazine*, vol. 21, no. 3, pp. 26–40, Sep. 2014.

[3.11] A. Marks, J. F. Whidborne, and I. Yamamoto, "Control allocation for fault tolerant control of a VTOL octorotor", in *UKACC International Conference on Control*, IEEE, 2012, pp. 357–362.

[3.12] T. Haus, M. Orsag, and S. Bogdan, "Omnidirectional vision based surveillance with spincopter", in *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, 2013, pp. 326–332.

[3.13] H. Youngren, S. Jameson, and B. Satterfield, "Design of the SAMARAI monowing rotorcraft nano air vehicle", in *Proceedings of the American Helicopter Society AHS 65th Annual Forum and Technology Display*, 2009.

[3.14] Y. Zhang, A. Chamseddine, C. Rabbath, B. Gordon, C.-Y. Su, S. Rakheja, C. Fulford, J. Apkarian, and P. Gosselin, "Development of advanced FDD and FTC, techniques with application to an unmanned quadrotor helicopter testbed", *Journal of the Franklin Institute*, vol. 350, no. 9, pp. 2396–2422, 2013.

[3.15] Fruity chutes, "Multicopter, quadcopter, drone, RC aircraft recovery and rescue chutes", *(Accessed 29 Nov. 2014)* `http://www.fruitychutes.com/uav_rpv_drone_recovery_parachutes/multicopter_quadcopter-rc_aircraft_recovery_and_rescue_chutes.htm`, 2014.

[3.16] Dronologista, "DJI dropsafe system", *(Accessed 29 Nov. 2014)* `http://robohub.org/dji-dropsafe-system/`, Oct. 2014.

[3.17] M. Ranjbaran and K. Khorasani, "Fault recovery of an under-actuated quadrotor aerial vehicle", in *IEEE Annual Conference on Decision and Control (CDC)*, IEEE, 2010, pp. 4385–4392.

[3.18] H. A. Izadi, Y. Zhang, and B. W. Gordon, "Fault tolerant model predictive control of quad-rotor helicopters with actuator fault estimation", in *Proceedings of the 18th IFAC World Congress*, 2011, pp. 6343–6348.

[3.19] A. Chamseddine, Y. Zhang, C. A. Rabbath, C. Join, and D. Theilliol, "Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 2832–2848, 2012.

[3.20] A. Freddi, A. Lanzon, and S. Longhi, "A feedback linearization approach to fault tolerance in quadrotor vehicles", in *Proceedings of the 18th IFAC World Congress*, 2011, pp. 5413–5418.

[3.21] A. Akhtar, S. Waslander, and C. Nielsen, "Fault tolerant path following for a quadrotor", in *IEEE Annual Conference on Decision and Control (CDC)*, Dec. 2013, pp. 847–852.

[3.22] A. Lanzon, A. Freddi, and S. Longhi, "Flight control of a quadrotor vehicle subsequent to a rotor failure", *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 2, pp. 580–591, 2014.

[3.23] M. C. Achtelik, K.-M. Doth, D. Gurdan, and J. Stumpf, "Design of a multi rotor MAV with regard to efficiency, dynamics and redundancy", in *AIAA Guidance, Navigation, and Control Conference*, 2012, pp. 1–17.

[3.24] Y. Kataoka, K. Sekiguchi, and M. Sampei, "Nonlinear control and model analysis of trirotor uav model", in *Proceedings of the 18th IFAC World Congress*, IFAC, vol. 18.1, 2011, pp. 10 391–10 396.

[3.25] Y. Kataoka, K. Sekiguchi, and M. Sampei, "Circle motion control of trirotor uav via discrete output zeroing", in *IEEE Annual Conference on Decision and Control (CDC)*, IEEE, 2013, pp. 226–231.

[3.26] E. R. Ulrich, J. S. Humbert, and D. J. Pines, "Pitch and heave control of robotic samara micro air vehicles", *Journal of Aircraft*, vol. 47, no. 4, pp. 1290–1299, 2010.

[3.27] M. Orsag, J. Cesic, T. Haus, and S. Bogdan, "Spincopter wing design and flight control", English, *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1-4, pp. 165–179, 2013.

[3.28] M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[3.29] P. Martin and E. Salaün, "The true role of accelerometer feedback in quadrotor control", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 1623–1629.

[3.30] P. H. Zipfel, *Modeling and Simulation of Aerospace Vehicle Dynamics Second Edition*. American Institute of Aeronautics and Astronautics, 2007.

[3.31] P. Pounds, R. Mahony, P. Hynes, and J. Roberts, "Design of a four-rotor aerial robot", in *Australasian Conference on Robotics and Automation*, vol. 27, 2002, p. 29.

[3.32] B. W. McCormick, *Aerodynamics Aeronautics and Flight Mechanics*. John Wiley & Sons, Inc, 1995.

[3.33] R. Mahony, V. Kumar, and P. Corke, "Aerial vehicles: Modeling, estimation, and control of quadrotor", *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[3.34] M. D. Shuster, "A survey of attitude representations", *The Journal of the Astronautical Sciencies*, vol. 41, no. 4, pp. 439–517, 1993.

[3.35] F. Callier and C. Desoer, *Linear System Theory*, ser. Springer Texts in Electrical Engineering. Springer, 1994.

[3.36] D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz", in *IEEE International Conference on Robotics and Automation (ICRA)*, Apr. 2007, pp. 361–366.

[3.37] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena", *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[3.38] R. Dorf and R. Bishop, *Modern Control Systems, Eleventh Edition*. Prentice-Hall, 2008.

[3.39] B. D. Anderson and J. B. Moore, *Optimal control: Linear quadratic methods*. Courier Dover Publications, 2007.

[3.40] S. Salazar-Cruz, F. Kendoul, R. Lozano, and I. Fantoni, "Real-time control of a small-scale helicopter having three rotors", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2006, pp. 2924–2929.

[3.41] J.-T. Zou, K.-L. Su, and H. Tso, "The modeling and implementation of tri-rotor flying robot", *Artificial Life and Robotics*, vol. 17, no. 1, pp. 86–91, 2012.

Paper P4

Critical subsystem failure mitigation in an indoor UAV testbed

Mark W. Mueller and Raffaello D'Andrea

Abstract

An autonomous safety mechanism is presented, as implemented in an indoor flying vehicle research testbed. The safety mechanism relies on integration of onboard gyroscope measurements and thrust commands to estimate the vehicle state for short lengths of time. It is used in the case of loss of external control signal or loss of external measurement data, to reduce the likelihood of a vehicle crash, or at least reduce the severity of an unavoidable crash. As UAVs move into ever more mainstream applications with increased public interaction, such safety systems become more critical.

Published in *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).*

1. Introduction

The Flying Machine Arena (FMA) is a testbed for aerial vehicle research, with a unique mix of fundamental research and public engagement. Research and demonstrations are done using a fleet of flying vehicles, and the correct functioning of the system relies on a high-precision motion capture system, and a low-latency command radio link. Failure of either of these subsystems renders the vehicles effectively blind, except for their inertial sensors – this paper describes a strategy to reduce the impact of such a failure.

The FMA is a $10 \times 10 \times 10$m space, covered by netting on the sides and padding on the floor. The vehicles used in research are quadrocopters, control of which is split into two layers (onboard the vehicle and offboard on a ground station), which are connected by wireless links. Testbeds similar to the FMA are GRASP at the University of Pennsylvania [4.1]; Stanford/Berkley Starmac [4.2] and MIT Raven [4.3].

The FMA has become well known for regular interactive demonstrations to the public, including dancing to music [4.4], balancing an inverted pendulum [4.5] and juggling balls [4.6]. During 2011, 44 events were held, during which 430 visitors attended demonstrations. Examples of demonstrations are shown in Fig. 4.1, where the mobile version of the FMA infrastructure is also shown. This mobile infrastructure allows the system to be deployed at temporary sites – the first external demonstration of the FMA was for the Flight Assembled Architecture project at an exhibition in Orléans, France. The fully autonomous construction of a 6 m structure consisting of 1500 foam bricks was done over four days, in front of (and directly above) a large crowd. This mix of research and public engagement places a high demand on the safety features of the system, where it is needed that the system be safe to operate at demonstrations with lay-people, without restricting the performance of the system, and still allowing the system to be flexible enough to allow for the rapid development of new projects.

Traditional safety mechanisms, e.g. safety pilot are not applicable here: the relatively small working space implies little time to react, while flying with a large number of vehicles would necessitate a large number of safety pilots. For example, in [4.7] the risk reduction strategy requires two safety operators: one to monitor a ground station, and an emergency pilot to take over manual control in the case of an emergency. In this paper, algorithms with low computational cost are developed allowing the vehicle greater autonomy from the ground station, without requiring additional bulky hardware (such as laser scanners, or cameras). It is important to note that the scope of this not as broad as e.g. completely autonomous flight using vision [4.8] or laser scanners [4.9].

The paper is organised as follows: Section 2 describes the system and its components, in 3 the failure modes are discussed, while the mechanism is discussed in Section 4, with results given in 5. A conclusion is given in Section 6.

2. System overview

2.1 Vehicle dynamics

The vehicles of choice in the FMA are modified Ascending Technologies Hummingbird quadrocopters, as shown in Fig. 4.1 and 4.2, each with four alternately rotating propellers. By mixing the thrusts f_i from each propeller i, the vehicle can generate moments about all the body axes, and a total thrust force f_T. The position, velocity and acceleration of the vehicle in the inertial frame are written as $\boldsymbol{x}^I$, $\dot{\boldsymbol{x}}^I$ and $\ddot{\boldsymbol{x}}^I$, respectively. The rotation of the vehicle body frame w.r.t. the inertial frame is $\boldsymbol{R}^{IB}$, and the body rates (in the body frame) are $\boldsymbol{\omega}^B = (p, q, r)$. The inertia of the vehicle, expressed about its axes, is $\boldsymbol{I}_B^B$ and the mass of the vehicle is m_B.

The rigid body dynamics of the vehicle are given by

$$\ddot{\boldsymbol{x}}^I = \boldsymbol{R}^{IB}\boldsymbol{c}^B + \boldsymbol{g}^I \tag{4.1}$$

$$\dot{\boldsymbol{R}}^{IB} = \boldsymbol{R}^{IB}\left[\boldsymbol{\omega}^B\times\right] \tag{4.2}$$

$$\boldsymbol{I}_B^B\dot{\boldsymbol{\omega}}^B = \boldsymbol{I}_B^B\boldsymbol{l}^B - \left[\boldsymbol{\omega}^B\times\right]\boldsymbol{I}_B^B\boldsymbol{\omega}^B, \tag{4.3}$$

where $\left[\boldsymbol{\omega}^B\times\right]$ is the skew-symmetric 3×3 matrix representation of the cross product of $\boldsymbol{\omega}^B = (p, q, r)$[4.10]. The mass-normalised thrust force $\boldsymbol{c}^B$ and moment $\boldsymbol{l}^B$ produced by

Figure 4.1. Close interaction with the public places high demands on the safety systems of the Flying Machine Arena. At the top, a primary school class attending a demonstration, and at the bottom picture a quadrocopter placing a brick as part of the Flight Assembled Architecture project.

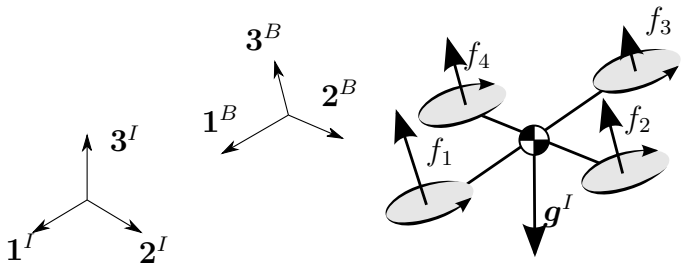

Figure 4.2. Dynamics of a quadrocopter, with the triple $(\mathbf{1}^I, \mathbf{2}^I, \mathbf{3}^I)$ defining the ground-fixed inertial frame, and $(\mathbf{1}^B, \mathbf{2}^B, \mathbf{3}^B)$ the body-fixed frame, related by the rotation $\boldsymbol{R}^{IB}$ and translation $\boldsymbol{x}^I$ – see also (4.1).

the propellers are expressed in the body frame as [4.1]

$$\boldsymbol{l}^B = \begin{bmatrix} l(f_2 - f_4) \\ l(f_3 - f_1) \\ \kappa(f_1 - f_2 + f_3 - f_4) \end{bmatrix} \tag{4.4}$$

$$\boldsymbol{c}^B = (0, 0, c) \tag{4.5}$$

$$c = (f_1 + f_2 + f_3 + f_4)/m_B \tag{4.6}$$

$$\boldsymbol{g}^I = (0, 0, -g)\,. \tag{4.7}$$

The moment is a function of the thrusts, propeller distance from the body centre l and an experimentally determined constant κ.

The rotation matrix $\boldsymbol{R}^{IB}$ can be be characterised by three Euler angles: here the yaw, pitch, roll sequence is used; rotating consecutively about the inertial z axis by the yaw angle ψ, then about the (new) y axis by the roll angle θ and finally about the (resulting) x axis by the roll angle ϕ, to get the rotated body frame. The rotation matrix is then characterised as $\boldsymbol{R}^{IB}(\psi, \theta, \phi) = \boldsymbol{R}_3(\psi)\boldsymbol{R}_2(\theta)\boldsymbol{R}_1(\phi)$ [4.10], where

$$\boldsymbol{R}_1(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{4.8}$$

$$\boldsymbol{R}_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{4.9}$$

$$\boldsymbol{R}_3(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.10}$$

2.2 Communication

There are two communication channels linking the vehicle to the ground station, which will be called the command and the data channel. The command channel is a low-latency, simplex channel, while the data channel is a high-bandwidth, variable latency duplex channel. The output of the high-level controller is transmitted to the vehicle over the command channel, while the data channel is used for non-critical data such as sending offboard state estimates (as described in this paper), and also vehicle feedback and parameter read/write commands from the ground station.

2.3 Control

Control of the quadrocopter is split into two parts, see Fig. 4.3: firstly a high-level controller generating desired normalised thrust and desired body rates on the ground station, in order to achieve some high-level goal (e.g. tracking a trajectory to balance a pendulum [4.5]). This loop runs at approximately 60 Hz, and the commands are sent to the vehicle over the command channel. The high-level controller performs feedback on a state estimate of the vehicle, using pose measurements from the motion capture system.

The second part is the low-level controller, running onboard the vehicle, which controls the motor speeds to achieve the desired body rates and total thrust, using rate gyro measurements in feedback. The onboard controller is run at 800 Hz. For the purposes of this paper, the control of the motor speeds is ignored, and assume that individual motor thrusts are directly controlled.

Each vehicle is marked with a unique configuration of three markers, which are visible to a motion capture system. This system measures the position and attitude of each vehicle, with a precision of approximately 0.1 mm and 0.1°, respectively, at a rate of 200 Hz. These measurements are then used to estimate the state of the vehicle.

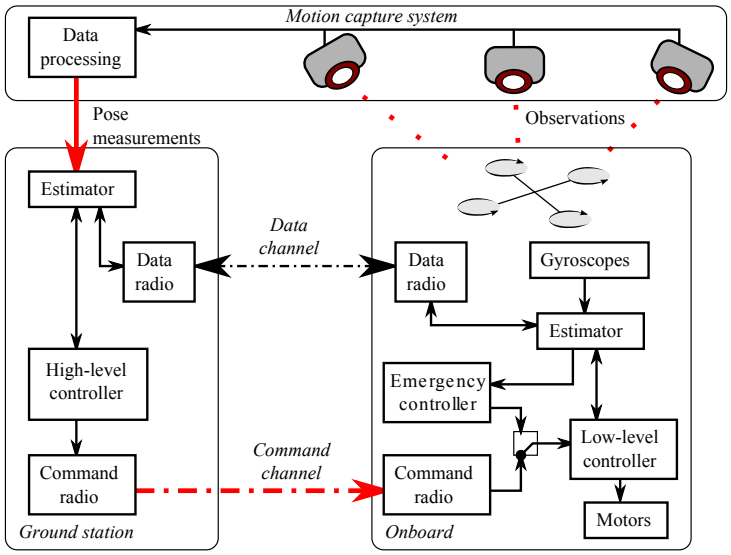

Figure 4.3. System layout of the Flying Machine Arena. Critical paths are marked in red, radio connections are marked with dash-dotted lines.

Further details of the FMA can be found in [4.11].

3. Failure modes

We wish to guard against failures of the offboard system, which would result in its inability to control the vehicle normally. Specifically, we are concerned with the red lines of Fig. 4.3; loss of the motion capture data and a loss of the command channel.

Loss of the motion capture data can be due to loss of data for a specific vehicle (e.g. occlusion), or complete loss of motion capture data (e.g. software crash). Upon losing the motion capture data, the offboard system is rendered blind and can no longer estimate the dynamic state of the vehicle. The second mode of failure is loss of the command channel: in this case the offboard commands can no longer reach the vehicles.

Loss of only the data channel would mean that the failsafe mechanism cannot take over if a critical system were to also fail (as the offboard updates are needed so that the vehicle has a reliable onboard estimate). If a failure of the data channel is detected, the vehicle is landed (using the regular offboard systems) until the data channel is restored.

4. Fail-safe mechanism

Here a mechanism is developed to delay a vehicle crash in the event of a critical subsystem failure so that the system may recover; or if a crash is inevitable, to mitigate its severity and allow sufficient warning for any people inside the arena. An emergency controller to achieve hover for short periods of time is developed, controlling on a state estimated using inertial sensors and occasional offboard updates.

It is assumed that the vehicle is in near-hover such that the angles ϕ and θ are small and that the yaw angle remains constant, so that dynamics decouple along the axes defined by a yawed reference frame:

$$\boldsymbol{x}^Y = \boldsymbol{R}_3(\psi)^T \boldsymbol{x}^I \tag{4.11}$$

$$\dot{\boldsymbol{x}}^Y = \boldsymbol{R}_3(\psi)^T \dot{\boldsymbol{x}}^I = (v_x, v_y, v_z) \tag{4.12}$$

$$\ddot{\boldsymbol{x}}^Y = \boldsymbol{R}_3(\psi)^T \ddot{\boldsymbol{x}}^I = (\dot{v}_x, \dot{v}_y, \dot{v}_z) \tag{4.13}$$

$$\left(\dot{\phi}, \dot{\theta}, \dot{\psi}\right) \approx (p, q, 0)\,. \tag{4.14}$$

Note that the yaw angle is typically controlled explicitly to some constant angle by the offboard high level controller.

Because $\boldsymbol{g}^I$ lies along the axis of rotation of $\boldsymbol{R}_3$, and is thus expressed the same in

both frames, rewriting (4.1) yields

$$\ddot{\boldsymbol{x}}^Y \approx \boldsymbol{R}_2(\theta)\boldsymbol{R}_1(\phi)\boldsymbol{c}^B + \boldsymbol{R}_3(\psi)^T\boldsymbol{g}^I \tag{4.15}$$

$$\begin{bmatrix}\dot{v}_x\\ \dot{v}_y\\ \dot{v}_z\end{bmatrix} \approx \begin{bmatrix}\sin\theta\cos\phi\\ -\sin\phi\\ \cos\theta\cos\phi\end{bmatrix} c + \begin{bmatrix}0\\0\\-g\end{bmatrix}. \tag{4.16}$$

Because the axes are assumed to decouple, only the strategy for estimating and controlling v_x, θ and q are described; the analogue for v_y, ϕ and p being straightforward to derive.

4.1 Onboard lateral state estimate

Via the data channel, the current off-board estimates of v_x and θ are periodically sent, denoted with $\bar{v}_x(t_n)$ and $\bar{\theta}(t_n)$, respectively, valid at time t_n.

The procedure to estimate the speed v_x of the quadrocopter is then as follows, where $\hat{v}$ denotes the onboard estimate of a variable v. The body rate is estimated directly from the gyroscopes, using a simple predict-correct estimator. Let Δt represent the period of the onboard loop (in reality, $\Delta t = 1.25\,\text{ms}$), $\hat{q}_0$ the estimate of the gyro bias, $\check{q}[k]$ the gyro output at time step k, and I_{xx} is the moment of inertia of the body about the body x axis:

$$\hat{q}^-[k+1] = \hat{q}[k] + \Delta t\, l\, (f_2 - f_4)\, / I_{xx} \tag{4.17}$$

$$\hat{q}[k+1] = \lambda\, \hat{q}^-[k+1] + (1-\lambda)\,(\check{q}[k] - \hat{q}_0)\,, \tag{4.18}$$

with λ a filter parameter. The rate estimate is then integrated for the angle estimate, using simple Euler integration. The acceleration is calculated from (4.16), and integrated for the velocity estimate, where $\hat{\theta}[k|n]$ is the estimate at step k, using an external update number n,

$$\hat{\theta}[k+1|n] = \hat{\theta}[k|n] + \hat{q}[k]\Delta t \tag{4.19}$$

$$\hat{v}_x[k+1|n] = \hat{v}_x[k|n] + c[k]\,\sin\hat{\theta}[k|n]\Delta t. \tag{4.20}$$

If a new external estimate is received at step k, the estimate is updated as follows:

$$\theta[k|n+1] = \bar{\theta}[n+1] + \sum_{l=k-N}^{k} \hat{q}[k]\Delta t \tag{4.21}$$

$$\hat{v}_x[k|n+1] = \bar{v}_x[n+1] + \sum_{l=k-N}^{k} c[k]\sin\hat{\theta}[k]\Delta t, \tag{4.22}$$

where N is age of the offboard estimate in time steps, which is the delay between the data being captured from which the offboard estimate is formed, and its arrival onboard. This means the last N steps of rate and angle estimates, as well as the collective acceleration command, need to be stored.

Upon loss of the motion capture data, or the data channel, the estimates would be propagated using only the gyroscopes, i.e. (4.19)-(4.20). The estimation error is discussed more fully below.

4.2 Emergency control strategy

Here the control strategy to bring the vehicle to hover is developed, i.e. lateral speeds to zero. Initially, he distinction between true state and the estimate thereof is disregarded, and we neglect the effects of discretization and design the controller in continuous time. The controller is designed by linearising (4.16) about hover, and it is desired that the vertical acceleration be zero, so that differentiating yields

$$c \approx g,\ \dot{c} \approx 0,\ \ddot{v}_x \approx \dot{\theta} g,\ \sin\theta \approx \theta, \tag{4.23}$$

where $\dot{\theta} = q^c$ is taken as system input. Choosing a feedback law

$$q^c = -2\zeta\omega_n\theta - \frac{\omega_n^2}{g} v_x \tag{4.24}$$

results in the differential equation for velocity

$$\ddot{v}_x + 2\zeta\omega_n\dot{v}_x + \omega_n^2 v_x = 0, \tag{4.25}$$

by which the speed should be driven to zero like a second order system with damping ratio ζ and natural frequency ω_n.

By (4.3) it is clear that $q^c = q$ is not directly controlled, but only $\dot{q}$, which is controlled through the low-level controller with the proportional feedback law

$$\dot{q} = k_q \left(q^c - q\right), \tag{4.26}$$

which combined with the above feedback law of (4.24) introduces a third pole to the system, giving

$$\frac{1}{k_q}\dddot{v}_x + \ddot{v}_x + 2\zeta\omega_n\dot{v}_x + \omega_n^2 v_x = 0. \tag{4.27}$$

If $k_q \geq 10|\zeta\omega_n|$, the response of the system can be approximated by the dominant roots of the second order system [4.12]. The values used in the FMA are $k_q = 100\,/\mathrm{s}$, $\zeta = 0.7$ and $\omega_n = 2\,\mathrm{rad/s}$, justifying the neglect of the effects of rotational inertia.

In reality we do not have access to the true variables, but only their estimates $\hat{q}$, $\hat{\theta}$ and

$\hat{v}_x$. These will have some initial error and be further corrupted by measurement errors, which can will be modelled as a constant bias error $e_q = q[k] - \hat{q}[k]$, and some zero-mean noise whose effect is neglected. We define the error variable $e_v[k] = v_x[k] - \hat{v}_x[k]$, $e_\theta[k] = \theta[k] - \hat{\theta}[k]$, so that

$$e_\theta[k+1] = e_\theta[k] + e_q\, t = e_\theta[0] + e_q \Delta t k \tag{4.28}$$

$$\begin{aligned} e_v[k+1] &= e_v[k] + g\, e_\theta[k] \Delta t \\ &= e_v[0] + g\, e_\theta[0] \Delta t k + \frac{1}{2} g\, e_q \Delta t^2 k(k+1). \end{aligned} \tag{4.29}$$

This implies that, although the feedback law acts to drive $\hat{v}_x$ to zero (and therefore $\hat{\theta}$ too), an initial error will cause the true angle θ to settle at a non-zero value, while v_x will grow linearly. A gyro bias will cause θ to linearly diverge, and the speed to grow with k^2. By implication, therefore, the position will grow with k^3. This clearly shows the importance of well calibrated gyros, and unbounded errors mean that this strategy cannot hold the vehicle in a hover indefinitely.

To maintain zero inertial z acceleration,

$$c[k] = \frac{g}{\cos\hat{\theta}[k] \cos\hat{\phi}[k]} \tag{4.30}$$

is applied, from (4.16), instead of (4.23). Here it should be noted $c[k] \geq g$, i.e. the effects of noise on the angle estimate will be visible in the vehicle accelerating upwards. In the implementation, (4.24) is limited such that $|\theta| < 60°$.

4.3 Switching logic

The emergency controller is triggered when the vehicle stops receiving external commands via the command channel, or the commands indicate (by an explicit flag) that the external controller has stopped receiving measurements from the motion capture system. When the emergency controller is run for the first time, the vehicle evaluates its state estimate: firstly, the last update needs to be less than 2 s old, and the estimated angles need to be less than 60°, and the speed less than 10 m/s. These values were chosen based on experiments, as limits of a "good" estimate. If the onboard estimate does not satisfy all of these criteria, the vehicle immediately switches off all propellers, and will fall ballistically. The idea here is that it is safer to just fall than to control the vehicle on a bad estimate.

The vehicle will then fly using the onboard emergency controller for a period of 2 s, after which it will enter a "crash landing" mode, where the total thrust is reduced to $c = g - 1\,\mathrm{m/s^2}$ such that the vehicle will accelerate downwards at $1\,\mathrm{m/s^2}$. The duration of the crash landing manoeuvre is calculated as the time required to descend at $1\,\mathrm{m/s^2}$ from the last measured vehicle height. When the crash landing period is completed, the vehicle switches off its motors and enters an idle mode. At any point, if the cause of the emergency procedure is resolved (i.e. the command channel comes back online, or

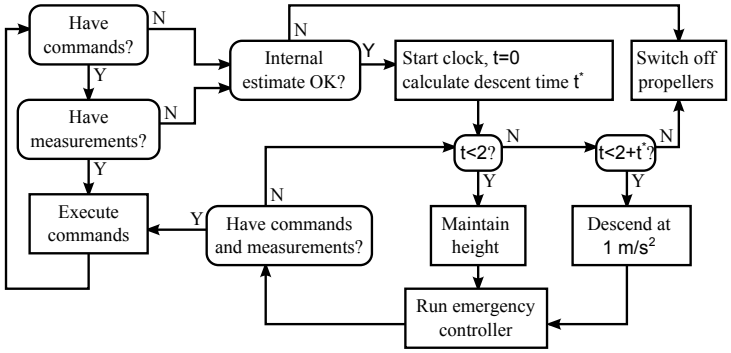

Figure 4.4. Emergency controller switching logic.

the motion capture data returns), control is immediately handed back to the offboard controller. This is illustrated in Fig. 4.4.

5. Experimental results

5.1 Experimental setup

To test the emergency algorithm, a complete loss of data from the motion capture system is triggered. The vehicle is flown horizontally along the inertial x-axis at constant speed, using feedback on the motion capture system and external commands sent through the command channel, while external state updates are sent to the vehicle at 10 Hz through the data channel. When the vehicle passes through $x = 0$, the motion capture data is cut off, and the vehicle is commanded to fly using only onboard estimates. The external state updates also stop. For two seconds, the vehicle uses the controller of Section 4.2 to attempt a level hover, after which it will descend at $1\,\mathrm{m/s^2}$ for the time necessary to descend from its last observed height.

The damping ratio for the emergency controller is set to $\zeta = 0.7$, and the natural frequency to $\omega_n = 2\,\mathrm{rad/s}$. The age of the offboard estimate is set to $N = 21$ internal cycles (or about 27 ms) – this is an estimate of the mean total time elapsed between the motion capture frame grab and corresponding update arriving at the vehicle, and is the sum of the motion capture image processing, offboard estimation processing and data channel communication times.

5.2 Calibration

It is clear from Section 4.2 that the emergency control strategy relies on a well-calibrated vehicle, i.e. with correctly identified gyroscope biases and the ability to accurately follow motor thrust commands. To this end a static calibration routine has been developed in the FMA, which identifies the gyroscope biases and a thrust mapping factor for each rotor.

The initial guess for the gyroscope biases $\hat{p}_0$ is formed by integrating the gyro output for the first two seconds after the vehicle has been switched on, and taking the mean value. A further calibration is performed when the vehicle is in hover (as determined by the motion capture system), and the gyroscope outputs are again averaged over a similar time period. During the hover periods, the system also calibrates for a static measurement frame misalignment, noting that the thrust vector during hover must point exactly opposite gravity and thus any pitch or roll angles measured are the frame misalignments.

Since direct control of the total thrust is assumed in (4.23), the propellers must be characterised: due to e.g. motor and propeller wear, the individual motors will not produce exactly the expected forces. To compensate for this, a static thrust correction factor γ_i for each propeller i is calibrated for, which relates the thrust output f_i to the desired motor thrust f_i^c by $f_i = \gamma_i f_i^c$. Combining this with (4.4) and (4.6) and setting the moments to zero and total thrust to gravity yields

$$\begin{bmatrix} 0 & lf_2^c & 0 & -lf_4^c \\ -lf_1^c & 0 & lf_3^c & 0 \\ \kappa f_1^c & -\kappa f_2^c & \kappa f_3^c & -\kappa f_4^c \\ \frac{f_1^c}{m} & \frac{f_2^c}{m} & \frac{f_3^c}{m} & \frac{f_4^c}{m} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \gamma_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ g \end{bmatrix}. \tag{4.31}$$

Calibration is automatically performed whenever the vehicle is in stationary hover throughout a flight.

5.3 Results

The trajectory results for a series of experiments are shown in Fig. 4.5. Over ten runs the average velocity magnitude at the end of 2 s of emergency onboard control is 0.3 m/s when starting from hover, 0.6 m/s when starting with a lateral speed of 1 m/s, 1.1 m/s when starting with at 2 m/s and 2.0 m/s when starting with at 4 m/s. When starting at 8 m/s, the vehicle will hit the floor before 2 s, but the velocity magnitude at that time is on average 3.5 m/s, which represents an 80% decrease in vehicle kinetic energy – i.e. even though a crash is not prevented, its severity is hugely decreased. Furthermore, the crash is delayed by some time, allowing warning of any people in the flying space; and crucially if the failure is quickly resolved, the external control can resume control of the vehicle in the air, thereby avoiding a crash altogether.

Notable also is that the vehicle consistently overshoots its velocity target, i.e. the final velocity achieved points in the direction opposite the initial velocity. There are two likely explanations for this error: firstly, the propellers are more efficient during the initial phases of the braking manoeuvre, due to an increased angle of attack [4.13], meaning that the produced thrust during braking exceeds the desired thrust. Secondly, the vehicle experiences aerodynamic drag for which the emergency controller does not compensate. The effect of the drag can be seen in the history plot when starting at 4 m/s in Fig. 4.5, where the vehicle has an initial large pitch angle (ca. 10°), but is not accelerating, i.e.

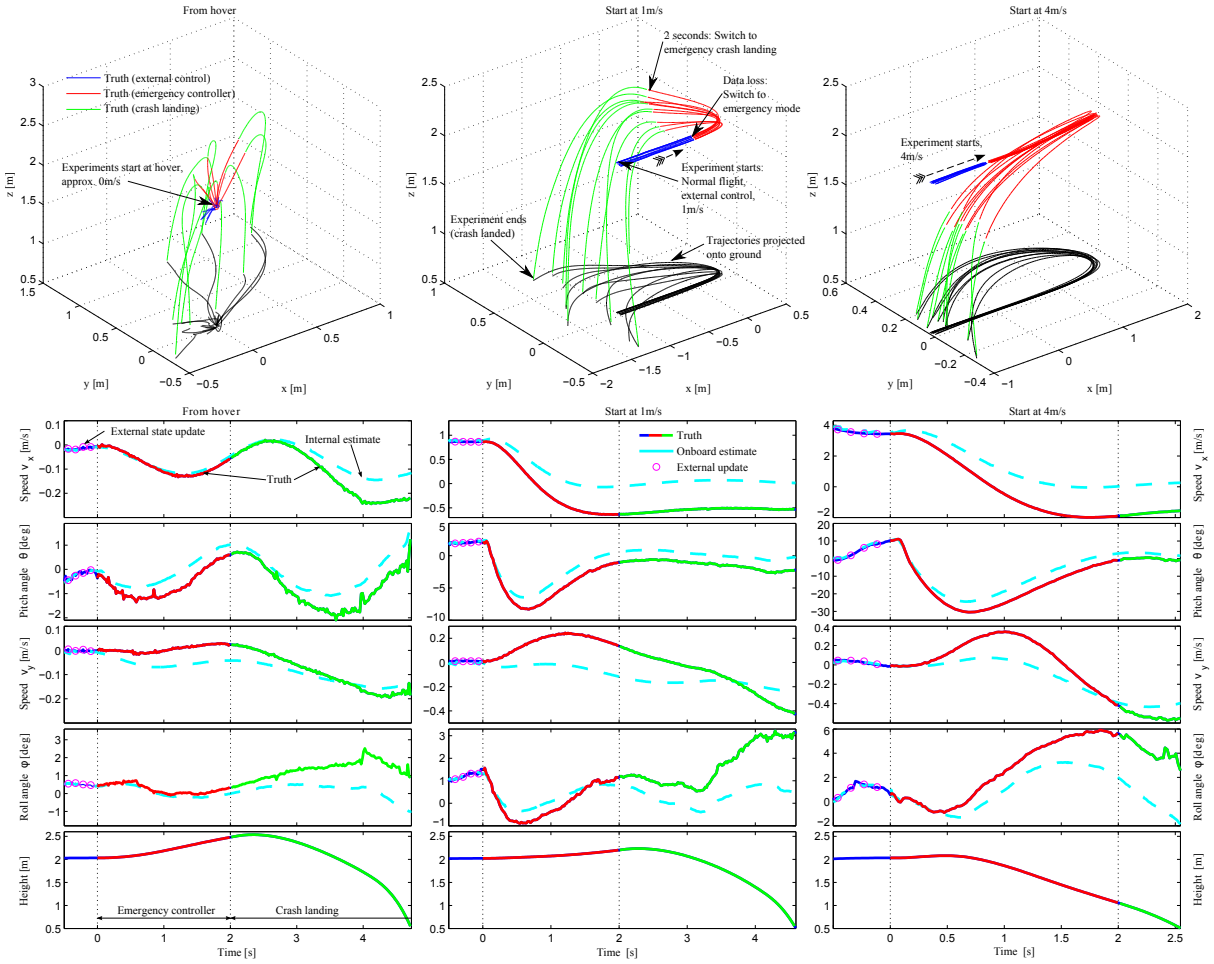

Figure 4.5. The top row of plots show vehicle trajectories under emergency onboard control, for three different initial conditions: starting at hover (left), with 1 m/s initial lateral velocity, and 4 m/s (right). Each plot shows ten trajectories, each divided into three stages: external control is marked blue, emergency hover red and the emergency descent crash landing in green. The experiment terminates at a height of 0.5 m. For ease of interpretation, the paths are also projected onto the $x-y$ plane in black. The lower plots show time histories of a typical trajectory for each starting speed, also showing the onboard estimate. Note that the colours of the "truth" line correspond to those of the top plots, and the differing scales for the lower plots. Data from the motion capture system is used as "truth".

the lateral components of the thrust are needed to balance out the drag force. The final velocity then represents the total impulse imparted by the drag force.

In Fig. 4.5 we see that the vehicle consistently accelerates slightly upwards when entering the emergency controller from a hover. This is most likely due to the effects of zero-mean noise on the angle estimate, as noted in Section 4.2, (4.30). The systematic tendency to turn to the left from speed is most likely due to vehicle mis-calibration, e.g. the left propeller producing less thrust than the opposing propeller.

6. Conclusion and Outlook

The strategy described in this paper allows for much greater confidence when using the FMA. They allow the system to recover from short-term failures which would otherwise render the vehicles uncontrollable, and reduce the severity of long-term failures. The strategy is implemented in such a way to be transparent to the user, allowing individual users to rapidly implement new controllers and run experiments, without in each case needing to be concerned with additional safety aspects.

This system is currently deployed, and runs in the background whenever a demonstration or experiment is in progress. One notable example of its use was during the architecture project where the mobile infrastructure of the FMA was deployed in Orléans, France (see Fig. 4.1). During operations, the commercial motion capture software crashed while a vehicle was flying – the emergency controllers kicked in and an alarm sounded. The emergency controller allowed sufficient time for an operator to grab a vehicle, protecting it from what would otherwise have been a fall of approximately 2 m onto a bare concrete floor.

Further improvements are possible to the emergency controller, especially if additional onboard sensors can be incorporated. Use of accelerometers, particularly, could be useful: both in feedback on lateral accelerations, taking advantage of aerodynamic effects to stabilise lateral speeds [4.14], and in feedback on the body z acceleration, reducing the effects of relative airflow on total produced thrust. Finally, the systematic nature of the errors seen in Fig. 4.5 suggest that a learning strategy might greatly improve the hovering performance.

7. Acknowledgements

This work builds on contributions to the FMA by many individuals, and particularly the authors would like to thank Sergei Lupahshin, Markus Hehn, Angela Schöllig and Federico Augugliaro for their discussions and feedback during the implementation of this work; Christoph Wegmüller, Guillaume Ducard and Markus Hehn for their work on vehicle calibration; and Michael Sherback and Thomas Kägi.

References

[4.1] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed", *Robotics Automation Magazine, IEEE*, vol. 17, no. 3, pp. 56–65, Sep. 2010.

[4.2] G. Hoffmann, D. Rajnarayan, S. Waslander, D. Dostal, J. Jang, and C. Tomlin, "The stanford testbed of autonomous rotorcraft for multi agent control (STARMAC)", in *Digital Avionics Systems Conference*, 2004.

[4.3] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment", *Control Systems Magazine, IEEE*, vol. 28, no. 2, pp. 51–64, Apr. 2008.

[4.4] A. Schoellig, F. Augugliaro, S. Lupashin, and R. D'Andrea, "Synchronizing the motion of a quadrocopter to music", in *Robotics and Automation, 2010 IEEE International Conference on*, May 2010, pp. 3355–3360.

[4.5] M. Hehn and R. D'Andrea, "A flying inverted pendulum", in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011.

[4.6] M. W. Mueller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling", in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[4.7] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Autonomous navigation and exploration of a quadrotor helicopter in GPS-denied indoor environments", in *Robotics: Science and Systems Conference*, Jun. 2008.

[4.8] S. Weiss, M. Achtelik, M. Chli, and R. Siegwart, "Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV", in *IEEE International Conference on Robotics and Automation*, 2012.

[4.9] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying", in *IEEE International Conference on Robotics and Automation*, 2009.

[4.10] P. H. Zipfel, *Modeling and Simulation of Aerospace Vehicle Dynamics Second Edition*. AIAA, 2007.

[4.11] S. Lupashin, A. Schoellig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips", in *IEEE International Conference on Robotics and Automation*, 2010, pp. 1642–1648.

[4.12] R. Dorf and R. Bishop, *Modern Control Systems, Eleventh Edition*. Prentice-Hall, 2008.

[4.13] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Precision flight control for a multi-vehicle quadrotor helicopter testbed", *Control engineering practice*, vol. 19, pp. 1023–1036, June 2011.

[4.14] P. Martin and E. Salaün, "The true role of accelerometer feedback in quadrotor control", in *IEEE International Conference on Robotics and Automation*, 2010.

Part C
State estimation

Paper P5

Kalman filtering with an attitude

Mark W. Mueller, Markus Hehn, and Raffaello D'Andrea

Abstract

This paper presents two generic algorithms for estimating a dynamic state that includes an attitude, one based on the extended Kalman filter, and the other on the unscented Kalman filter. The algorithms follow an approach popular in the current state of the art by implementing a minimal three dimensional attitude error in the filter state, and an additional reference attitude, where only the attitude error is assigned a covariance. The mean of the attitude error is kept small, while the reference attitude can be of any magnitude. This formulation avoids having redundant states in the filter, and the need to apply constraints to the state estimate. During a reset step, performed after each Kalman filtering step, the reference attitude is adjusted so that the attitude error is zero in expectation and thus far away from singularities. Two approaches for the reset step are proposed, one based on a first order analysis and the other on the unscented transform. In contrast to the current state of the art, the reset steps transform the estimate covariance. The reset steps are validated using Monte Carlo sampling. The algorithms may be applied to any system whose state includes an attitude or rotation, and where the dynamics and measurement equations are sufficiently smooth for the extended or unscented Kalman filters to be applied.

1. Introduction

For many systems, especially in robotics, the state to be estimated includes the attitude, or orientation, of one reference frame with respect to another. The estimation for such systems is often done with a Kalman filter, and, if the underlying system is sufficiently nonlinear, with an extended Kalman filter or an unscented Kalman filter. The extended Kalman filter applies the Kalman filter to a first-order approximation of the underlying nonlinear system, with the approximation evaluated at the current state estimate. The unscented Kalman filter instead uses a set of deterministically chosen points which approximate the underlying distribution, and these points are then transformed through the full nonlinear equations [5.1].

System dynamics that include attitudes are typically nonlinear. The task of state estimation with attitudes is further complicated by the inherent difficulties in parametrising attitudes: a rigid body has three degrees of freedom in rotation, however no three dimensional parametrisation of attitude exists that is both global and without singular points [5.2].

One approach when implementing a Kalman filter whose state includes an attitude representation is to choose a minimal three parameter representation, which has the disadvantage of singular points. An alternative approach is to use a higher dimensional representation in the filter state, thus avoiding singularities at the cost of coupling the states through constraints.

A third approach uses a redundant attitude representation, where both a reference attitude and an attitude error are used and their composition represents the attitude estimate: the attitude error is encoded with a minimal representation in the stochastic state (i.e. it has an associated covariance), and the reference attitude is updated to keep the attitude error small (but has no associated covariance). This avoids issues with the first two approaches: if the attitude were encoded using only three elements in the estimator state (i.e. without the reference attitude) the filter state would have discontinuous jumps at the singular points of the chosen representation; if instead a higher-dimensional representation is used in the stochastic state additional constraints need to be enforced.

A recent practical example of the minimal three parameter approach is given by [5.3], where three Euler angles are used for visual tracking of rigid objects. An example of the second approach is given in e.g. [5.4], where the goal is to track the 6DOF motion of a camera: a 13 state extended Kalman filter is used, with the attitude encoded in the state through the four-dimensional Euler symmetric parameters[2]. The symmetric parameters are constrained to have unit norm, and the constraint is enforced at each step of the filter.

The use of nonlinear constraints in Kalman filtering is discussed in [5.6], where the authors present methods for applying constraints, including the use of pseudo-measurements (where a zero-noise measurement of the constraint violation is used to enforce the con-

[2]The Euler symmetric parameters are often referred to as a "quaternion of rotation". This is unfortunate, as their invention predates the invention of quaternion algebra [5.5], and invoking four-dimensional hypercomplex numbers seems both unnecessary and distracting.

straint) and projection methods, and discuss the difficulties that arise specifically for nonlinear constraints. The example given is of estimating a dynamic state including an attitude, where the attitude is represented using the four dimensional symmetric parameters (and thus subject to a unit norm constraint). Norm constraints in Kalman filtering are specifically discussed in [5.7].

In addition to the difficulties of enforcing nonlinear constraints in a Kalman filtering setting, higher dimensional representations have the disadvantage of increasing the filter's computational complexity. The computational cost of a Kalman filter is approximately proportional to n^3, where n is the greater of the number of states and the number of measurements [5.8, p. 210].

An early example of the third approach, using a minimal attitude representation in the filter state to represent an attitude error in addition to having a reference attitude, is given in [5.9], where the goal is to estimate the attitude and angular velocity of a spacecraft using an extended Kalman filter, where the filter state includes three Euler angles which are reset to zero after each measurement update and thus kept far from their singularities. Another influential example is [5.10], where the resulting algorithm is called the Multiplicative Extended Kalman Filter (MEKF). The MEKF is also the inspiration for this work (note that the MEKF traces its origins to [5.11]). In [5.10] the problem of estimating the attitude and rate gyroscope bias of a spacecraft is addressed, and the presented algorithm uses a minimal three dimensional attitude error in a Kalman filter, in addition to a reference attitude. Information from the attitude error is moved to the reference attitude after each measurement update, in a so-called reset step. The authors claim that the reset step does not affect the filter covariance, specifically [5.10]: "The reset does not modify the covariance because it neither increases nor decreases the total information content of the estimate; it merely moves this information from one part of the attitude representation to another." Similar algorithms (with similar reset steps) are presented in [5.12]–[5.16], where similar claims are made that the reset does not modify the estimate covariance.

Unfortunately, this statement that the covariance is unchanged during the reset is incorrect, even to first order in the attitude error. One of the main goals of this work is to derive corrections for the attitude error mean and covariance during the reset. Two such corrections are derived: the first is computationally inexpensive, and based on a first-order approximation of the attitude error during the reset. This first order approximation is in line with the general philosophy behind the extended Kalman filter, of using first-order approximations of the system equations to approximate the conditional probability distributions of the state estimate, and the resulting covariance correction is of a similar form to the familiar Kalman filter prediction step. The second correction uses sigma-points computed with an unscented transform, and may be easily combined with an unscented Kalman filter. For each of the two corrections, a full Kalman filter algorithm is presented for estimating the state of a dynamic system including an attitude. These two algorithms are called the Extended Kalman Filter with an Attitude (EKFA) and the Unscented Kalman Filter with an Attitude (UKFA).

That something is amiss with the MEKF of [5.10] has been noted previously in the literature, specifically in [5.7], [5.17], and [5.18]. In [5.17] a filter is presented for estimating an attitude based on bias-free rate gyroscope measurements and unit vector measurements; and specifically a covariance correction is suggested similar to the one presented herein (see (28) and (29) of [5.17] – note however, that the assumptions made for the derivation are somewhat different). A second-order covariance correction is proposed in [5.7], but it is however still stated that no correction to the covariance is necessary to first order. Furthermore, in [5.15, p. 243] the reset of [5.10] is stated as being correct, with the caveat that "Not everyone agrees with this statement".

The approach of [5.10] has inspired many other works, such as [5.19]–[5.21], which also fail to do the reset correctly. The method of [5.10] with the incorrect reset has also been used on physical systems, such as the Radio Aurora Explorer satellite [5.22] or a balloon-borne telescope [5.23].

Further examples where a minimal three element representation is used in a Kalman filter with an additional reference attitude are [5.24] and [5.25], however no specific mention is made as to how information is moved from the filter state to the reference attitude.

There is also a rich literature on attitude estimation using frameworks other than Kalman filtering: an overview is given in [5.26], [5.27].

1.1 Contribution

The contribution of this paper is as follows:

1. To propose two algorithms which allow to estimate the dynamic state of a system (where the state includes an attitude), by making use of a reference attitude and an attitude error. One algorithm is an extension to the extended Kalman filter, the other an extension to the unscented Kalman filter, called respectively the Extended Kalman filter with an Attitude (EKFA), and the Unscented Kalman Filter with an Attitude (UKFA). Both algorithms may be directly applied to systems with arbitrarily complicated dynamics, including, for example, angular velocity dynamics dependent on other states.

2. To show that the widely used method presented in e.g. [5.10], [5.12]–[5.15] does not keep track of the estimate statistics correctly, even to first order.

3. To demystify the attitude reset step, used to move information from the small error rotation to the reference attitude, and to propose two alternative corrections for adapting the attitude error statistics during the reset.

1.2 Organisation

This paper is organised as follows: the next section presents the attitude representations and the related definitions and manipulations that are required throughout this work. In Section 3 the estimation problem is stated, and an outline of the estimation approach is presented, as well as a formal statement of the attitude reset problem. A reset step, where

information is moved from the minimal attitude representation in the filter state to the reference attitude, based on a first-order analysis is presented in Section 4. An alternative reset step, using the unscented transform, is presented in Section 5. Sections 4 and 5 both include Monte Carlo analyses to validate the approximations. Section 6 presents the resulting algorithms EKFA and UKFA, and the paper concludes in Section 7. In Appendix A a more detailed comparison is given of the EKFA and the MEKF of [5.10].

2. Attitude representations

For all derivations in this work the fundamental representation of attitude is the 3×3 rotation matrix, so that the transformation of a vector by an attitude from one reference frame to another is a matrix multiplication. Furthermore, the composition of consecutive attitude transformations is a matrix product.

This is best illustrated through an example: let u be a fixed vector in space, and let A, B, and C represent three reference frames in which u may be expressed; let $u^{(A)}$ be the expression of u in frame A, and similarly $u^{(B)}$ and $u^{(C)}$ respectively for frames B and C. Let $R^{(AB)}$ be the matrix representation of the attitude of frame A with respect to B, and define similarly $R^{(AC)}$ and $R^{(BC)}$. The expressions of u in the reference frames is then related by

$$u^{(C)} = R^{(CB)}\, u^{(B)} \tag{5.1a}$$
$$= R^{(CB)}\, R^{(BA)}\, u^{(A)} \tag{5.1b}$$
$$= R^{(CA)}\, u^{(A)}. \tag{5.1c}$$

This makes it clear that the composition of the rotations $R^{(CB)}$ and $R^{(BA)}$ is simply the matrix product $R^{(CA)} = R^{(CB)} R^{(BA)}$.

Different representations may however be used in the implementation of the resulting algorithm, which may offer practical benefits such as improved numerical stability or computational speed – for example the Euler symmetric parameters. Then the matrix multiplications, which represent composition of attitudes for the rotation matrix, are replaced by the composition rule for the chosen representation.

In the following, operators are defined that are used throughout this paper, followed by the definitions of two minimal attitude representations which may be used to encode an attitude as a three element vector (the rotation vector and the Rodrigues parameters). Finally, the relevant equations governing kinematics of attitude are presented. These representations (and their kinematics) are used in deriving and analysing the reset step.

2.1 Operator definitions

The skew-symmetric matrix form of the cross product $\mathrm{S}(v)$ of a vector $v = (v_1, v_2, v_3)$ is

defined as[3]

$$\mathrm{S}(v) := \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}. \tag{5.2}$$

The notation (v_1, v_2, v_3) is used to compactly denote the elements of a column vector.

Proposition 1. *The spectrum of the skew-symmetric matrix is* $\mathrm{spec}\,(\mathrm{S}(v)) = \{\pm i\,|v|\,, 0\}$*; where* $i^2 = -1$ *and* $|\cdot|$ *represents the Euclidean norm.*

Proof. By direct computation. □

Proposition 2. *Let* R *be a rotation matrix. Then for a vector* v

$$\mathrm{S}(Rv) = R\,\mathrm{S}(v)\,R^{-1}. \tag{5.3}$$

Proof. Follows from Fact 3.10.1 xxxvii) of [5.28], and noting that the determinant of a rotation matrix is unity. □

The operator $\mathrm{exps}(\cdot)$ and the Cayley transform $\mathrm{rot}(\cdot)$ are defined as

$$\mathrm{exps}(v) := \exp(\mathrm{S}(v)) \tag{5.4}$$

$$\mathrm{rot}(v) := (I + \mathrm{S}(v))\,(I - \mathrm{S}(v))^{-1} \tag{5.5}$$

where I is the identity matrix of appropriate dimension and $\exp(M)$ is the matrix exponential of a square matrix M, defined as

$$\exp(M) := \sum_{k=0}^{\infty} \frac{1}{k!} M^k. \tag{5.6}$$

It will be shown that both $\mathrm{exps}(v)$ and $\mathrm{rot}(v)$ are rotation matrices.

Proposition 3. *The Cayley transform* (5.5) *may be rewritten as an infinite sum if* $|v| < 1$*:*

$$\mathrm{rot}(v) = I + 2\sum_{k=1}^{\infty} \mathrm{S}(v)^{\,k}. \tag{5.7}$$

[3]Note the sign difference compared to the operator $[[\cdot]]$ defined in (44) of [5.5]. Note also the use "passive" rotations here, rather than the "active" rotations of [5.5]. The two forms are related through an inverse.

Proof. If $|v| < 1$, by Proposition 1 the spectral radius of $\mathrm{S}(v)$ is less than one and (by Fact 9.4.13 of [5.28])

$$(I - \mathrm{S}(v))^{-1} = \sum_{k=0}^{\infty} \mathrm{S}(v)^k. \tag{5.8}$$

Substituting now (5.8) into (5.5), (5.7) follows. □

Fact 1 (see (204) in [5.5])**.** *The inverse operation* $\mathrm{rot}^{-1}(\cdot)$ *of the Cayley transform* (5.5) *from a rotation matrix* R *to a vector is given by:*

$$\mathrm{rot}^{-1}(R) = \frac{1}{1 + \mathrm{tr}\,(R)} \begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix} \tag{5.9}$$

where $\mathrm{tr}\,(\cdot)$ *is the matrix trace, and* R_{jk} *the* (j,k)*th element of* R*.*

2.2 Rotation vector

An attitude may be expressed using a rotation vector $v \in \mathbb{R}^3$, where the unit vector in the direction of v represents the axis of rotation, and the magnitude of v represents the angle of rotation.

Fact 2 (see (112) in [5.5])**.** *The rotation matrix corresponding to the rotation vector may be computed as* $\exp(v)$*. Note that this matrix exponential can be readily computed in closed form using Euler's formula, see [5.5, eq. (96)-(99)].*

Proposition 4. *The following relationships hold for the matrix inverse of* $\exp(v)$

$$\exp(v)^{-1} = \exp(v)^{T} = \exp(-v)\,. \tag{5.10}$$

Proof. Follows directly from (5.4), and applying the fact that $\mathrm{S}(v)^{T} = -\mathrm{S}(v)$ for any vector v. □

Proposition 5. *Any vector* v *is unchanged by the rotation* $\exp(v)$*, i.e.*

$$\exp(v)\, v = v \tag{5.11}$$

Proof. By direct computation $\mathrm{S}(v)\, v = 0$ for all vectors v, from (5.2). The proposition follows by applying this to the infinite expansion of $\exp(v)$. □

2.3 Rodrigues parameters

The Rodrigues parameters (also known as the Gibbs parameters) $\delta \in \mathbb{R}^3$ are another way

to parametrise attitude, related to a rotation vector v by

$$\delta := \begin{cases} \tan\left(\frac{|v|}{2}\right) \frac{1}{|v|} v & \text{if } v \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5.12}$$

Fact 3 (see (203) in [5.5]). *The rotation matrix corresponding to the Rodrigues parameters δ is given by* $\mathrm{rot}(\delta)$.

Proposition 6. *The first order relationship between the rotation vector and the Rodrigues parameters may be expressed as*

$$\mathrm{exps}(v) = \mathrm{rot}\left(\tfrac{1}{2}v\right) + o(|v|) \tag{5.13}$$

where the Landau symbol $o(x)$ is used to represent higher-order terms, i.e. a quantity for which the following holds:

$$\lim_{x \to 0} \frac{o(x)}{x} = 0. \tag{5.14}$$

Proof. Follows by substituting the definition of the Rodrigues parameters (5.12) for the infinite expansion (5.7) and comparing to the infinite expansion of the matrix exponential (5.6) and (5.4). □

2.4 Kinematics of rotation

Fact 4 (see (261) of [5.5]). *If two reference frames are moving with respect to one another, with the time varying rotation matrix $R(t)$ representing their relative orientation and the vector $\omega(t)$ representing their relative angular velocity (with $\omega(t)$ expressed in the reference frame associated with the attitude $R(t)$), the differential equation governing $R(t)$ is as below:*

$$\frac{d}{dt} R(t) = R(t)\, \mathrm{S}(\omega(t))\,. \tag{5.15}$$

Note that if the angular velocity ω is constant over some time period $t \in [t_0, t_1]$, the above is a linear, time-invariant differential equation whose solution is

$$R(t_1) = R(t_0)\, \mathrm{exps}((t_1 - t_0)\omega)\,. \tag{5.16}$$

Fact 5 (see (330) of [5.5]). *The time derivative of the Rodrigues parameters $\delta(t)$, again*

as a function of the angular velocity $\omega(t)$, is given by

$$\frac{d}{dt}\delta(t) = \frac{1}{2}\left(\omega(t) + \mathrm{S}(\delta(t))\,\omega(t) + \left(\omega(t)^T\delta(t)\right)\delta(t)\right). \tag{5.17}$$

Remark. The Rodrigues parameters have the advantage over the rotation vector that their kinematic differential equation is continuous at zero (compare the above to (276) in [5.5]). This fact will be used in deriving the attitude reset.

3. Problem statement and solution approach

The problem considered is that of estimating the dynamic state of a system using measurements, a model for the state dynamics, a model of the measurement system, and information about the dynamic states' initial distributions. The system's dynamic state includes an attitude R and other states collected into the vector $\xi \in \mathbb{R}^{n_\xi}$, where the attitude R is taken to be a rotation matrix (although alternative representations may be used). The state evolves in discrete time steps k according to the following dynamic equations, with $\eta[k]$ process noise (assumed white, independent of the initial condition, and zero-mean):

$$\xi[k] = \bar{f}_1\left(k{-}1, \xi[k{-}1], R[k{-}1], \eta[k{-}1]\right) \tag{5.18}$$

$$R[k] = \bar{f}_2\left(k{-}1, \xi[k{-}1], R[k{-}1], \eta[k{-}1]\right). \tag{5.19}$$

If the underlying system dynamics are continuous, rather than in discrete time, the above equations may be taken as the integrals of the continuous equations over one sampling interval. These may then be approximated by e.g. an Euler discretisation.

Measurements $z[k]$ are available at discrete times, as a function of the state and the measurement noise variable $\zeta[k]$ (also assumed white, zero mean, and independent of both the initial condition and η):

$$z[k] = \bar{h}\left(k, \xi[k], R[k], \zeta[k]\right). \tag{5.20}$$

The goal is to estimate the system's state $\xi[k]$ and $R[k]$ recursively from the measurement sequence z, information about their initial conditions, and the dynamic and measurement models.

This is done by introducing the stochastic state $x[k] \in \mathbb{R}^{n_\xi+3}$, partitioned such that $x[k] = (\xi[k], \delta[k])$ where $\delta[k] \in \mathbb{R}^3$ represents a (small) attitude error parametrised through Rodrigues parameters. The attitude $R[k]$ associated with the system is written

as the composition of a reference attitude $R_{\text{ref}}[k]$ and the attitude error $\delta[k]$:

$$R[k] =: R_{\text{ref}}[k]\,\text{rot}(\delta[k])\,. \tag{5.21}$$

This representation introduces a redundant attitude, and this redundancy is exploited to avoid issues relating to singularities and constraints in attitude representations. It should be noted that this redundant formulation is not novel, and can be found in e.g. [5.10], or in a somewhat different form using Euler angles in [5.9].

The system dynamics (5.18)-(5.19) are now combined and rewritten to use the stochastic state $x[k]$ and the reference attitude $R_{\text{ref}}[k]$ so that

$$x[k] = f(k{-}1, x[k{-}1], R_{\text{ref}}[k{-}1], \eta[k{-}1]) \tag{5.22}$$

where specifically the change in attitude of (5.19) now affects only the attitude error $\delta[k]$, and not the reference attitude $R_{\text{ref}}[k]$. Rewriting these equations requires compositions and kinematics of attitudes using the Rodrigues parameters, such as those in Section 2.

The measurement equation is likewise rewritten as a function of the new state variables:

$$z[k] = h(k, x[k], R_{\text{ref}}[k], \zeta[k]). \tag{5.23}$$

The proposed algorithms use these rewritten equations: the EKFA based on the extended Kalman filter (see e.g. [5.1], [5.29]), and the UKFA based on the unscented Kalman filter (see e.g. [5.30]). Both introduce two additional steps to correct the attitude error statistics. The recursive estimation strategy then consists of four steps:

1. A Kalman prediction step, that uses the process equation (5.22) to propagate the estimate through the dynamics. During this step, the reference attitude is unchanged.

2. A prediction reset step, where the reference attitude is changed such that the post-reset attitude error estimate has zero expectation, i.e. the attitude error variable is maximally far from its singularities.

3. A Kalman measurement update, that uses the measurement model (5.23) to correct the estimate with a given measurement. During this step, the reference attitude is again unchanged.

4. A measurement reset step, where again the reference attitude is adapted such that the estimate of the post-reset attitude error is reset to zero.

The two alternative methods for achieving the reset steps (and their derivation) are novel compared to the methods of [5.10], [5.12]–[5.15], as is the applicability of the method to systems of any dynamics that can be expressed in the form (5.18)-(5.20).

3.1 Attitude error reset

The reset step modifies the reference attitude R_{ref} and the attitude error estimate δ, so that the post-reset attitude error estimate has zero expectation, i.e. the attitude error variable is maximally far from its singularities. Furthermore, the actual attitude remains unchanged, so that the composition of R_{ref} and δ is unchanged by the reset.

Problem 1. Let the pre-reset reference attitude be $R_{\text{ref,pre}}$, and the pre-reset attitude error be δ_{pre} with associated mean and covariance:

$$\mathrm{E}(\delta_{\text{pre}}) =: \mu_{\text{pre}} \tag{5.24}$$

$$\mathrm{Var}(\delta_{\text{pre}}) =: \Sigma_{\text{pre}} \tag{5.25}$$

We similarly define the post-reset variables $R_{\text{ref,post}}$, and δ_{post} with mean and covariance μ_{post} and Σ_{post}. The problem is to compute $R_{\text{ref,post}}$ and Σ_{post} subject to the following two equalities:

$$R_{\text{ref,post}}\,\mathrm{rot}(\delta_{\text{post}}) = R_{\text{ref,pre}}\,\mathrm{rot}(\delta_{\text{pre}}) \tag{5.26}$$

$$\mathrm{E}(\delta_{\text{post}}) = \mu_{\text{post}} = 0. \tag{5.27}$$

Two approximate solutions are presented to Problem 1. The first is based on a first-order approximation of the reset analysed as a continuous operation, derived in the next section. The second approach is based on the unscented transform and is presented in Section 5.

4. First order attitude reset

In this section an approximate solution to Problem 1 is derived based on studying the reset operation as a continuous rotation, whose effects are analysed to first order in the attitude error.

4.1 Solution approximation

Theorem 1. *To first order in the attitude error δ_{pre}, the solution to Problem 1 is:*

$$R_{\text{ref,post}} = R_{\text{ref,pre}}\,\mathrm{rot}(\mu_{\text{pre}}) \tag{5.28}$$

$$\Sigma_{\text{post}} = \mathrm{rot}\big(-\tfrac{1}{2}\mu_{\text{pre}}\big)\,\Sigma_{\text{pre}}\,\mathrm{rot}\big(-\tfrac{1}{2}\mu_{\text{pre}}\big)^{T}. \tag{5.29}$$

Proof. The pseudo-time $t \in [0, 1]$ is introduced, and the reset is considered as a continuous operation starting at $t = 0$ and ending at $t = 1$. The time-varying reference attitude $R_{\text{ref}}(t)$ and $\delta(t)$ are introduced, which have to satisfy the boundary conditions

$\delta(0) = \delta_{\text{pre}}$, $R_{\text{ref}}(0) = R_{\text{ref,pre}}$, $\delta(1) = \delta_{\text{post}}$, and $R_{\text{ref}}(1) = R_{\text{ref,post}}$. Given these boundary conditions, a sufficient condition for satisfying (5.26) is

$$\frac{d}{dt}\left(R_{\text{ref}}(t)\,\text{rot}(\delta(t))\right) = 0. \tag{5.30}$$

Applying the derivative product rule, and substituting the kinematic equation for the rotation matrix (5.15) yields

$$R_{\text{ref}}(t)\,\text{S}(\omega_{\text{ref}}(t))\,\text{rot}(\delta(t)) + R_{\text{ref}}(t)\,\text{rot}(\delta(t))\,\text{S}(\omega_\delta(t)) = 0 \tag{5.31}$$

where $\omega_{\text{ref}}(t)$ and $\omega_\delta(t)$ are angular velocities, with specifically $\omega_{\text{ref}}(t) = \omega_{\text{ref}}$ taken as a deterministic constant, to be computed. This then yields

$$\text{S}(\omega_\delta(t)) = -\text{rot}(\delta(t))^{-1}\text{S}(\omega_{\text{ref}})\,\text{rot}(\delta(t)) \tag{5.32}$$

or, by simplifying (see Proposition 2),

$$\omega_\delta(t) = -\text{rot}(-\delta(t))\,\omega_{\text{ref}}. \tag{5.33}$$

Substituting (5.33) for the angular velocity in the kinematic equation of the Rodrigues parameters (5.17), assuming that $|\delta| < 1$ to allow substitution of the infinite series (5.7) for $\text{rot}(\delta)$, and expanding gives

$$\frac{d}{dt}\delta(t) = -\frac{1}{2}\omega_{\text{ref}} - \frac{1}{2}\text{S}(\omega_{\text{ref}})\,\delta(t) + o(|\delta(t)|)\,. \tag{5.34}$$

Neglecting the higher order terms, (5.34) represents an affine, time-invariant, differential equation in $\delta(t)$, with ω_{ref} a deterministic constant, the solution to which is

$$\delta(1) = \exps\left(-\tfrac{1}{2}\omega_{\text{ref}}\right)\,\delta(0) - \tfrac{1}{2}\omega_{\text{ref}} + o(|\delta(0)|)\,. \tag{5.35}$$

The simple closed-form solution follows from the fact that $\text{S}(\omega)\,\omega = 0$ for all ω.

Neglecting again the higher order terms, and noting that $\delta_{\text{post}} = \delta(1)$, and $\delta_{\text{pre}} = \delta(0)$, it follows that

$$\text{E}(\delta_{\text{post}}) \approx \exps\left(-\tfrac{1}{2}\omega_{\text{ref}}\right)\,\text{E}(\delta_{\text{pre}}) - \tfrac{1}{2}\omega_{\text{ref}} \tag{5.36}$$

$$\text{Var}(\delta_{\text{post}}) \approx \exps\left(-\tfrac{1}{2}\omega_{\text{ref}}\right)\,\text{Var}(\delta_{\text{pre}})\,\exps\left(-\tfrac{1}{2}\omega_{\text{ref}}\right)^{T}. \tag{5.37}$$

Rearranging (5.36), exploiting Proposition 5, and enforcing the requirement that

$\mathrm{E}(\delta_{\mathrm{post}}) = 0$ yields

$$\omega_{\mathrm{ref}} = 2\mathrm{E}(\delta_{\mathrm{pre}})\,. \tag{5.38}$$

From this follows, to first order,

$$\mu_{\mathrm{post}} = \exps(-\mu_{\mathrm{pre}})\,\mu_{\mathrm{pre}} - \mu_{\mathrm{pre}} = 0 \tag{5.39}$$

$$\mathrm{Var}(\delta_{\mathrm{post}}) = \exps(-\mu_{\mathrm{pre}})\,\Sigma_{\mathrm{pre}}\,\exps(-\mu_{\mathrm{pre}})^{\,T}. \tag{5.40}$$

From (5.38) the post-reset reference $R_{\mathrm{ref,post}}$ can be computed, by noting that during the reset

$$\frac{d}{dt}R_{\mathrm{ref}}(t) = R_{\mathrm{ref}}(t)\,\mathrm{S}(\omega_{\mathrm{ref}})\,. \tag{5.41}$$

Substituting (5.16), it follows that

$$R_{\mathrm{ref,post}} = R_{\mathrm{ref,pre}}\exps(\omega_{\mathrm{ref}}) \tag{5.42a}$$

$$= R_{\mathrm{ref,pre}}\,\exps(2\,\mu_{\mathrm{pre}})\,. \tag{5.42b}$$

Using Proposition 6 it follows that, to first order, (5.39), (5.40), and (5.42b) may be rewritten using operator $\mathrm{rot}(\cdot)$ which is naturally associated with the Rodrigues parameters δ, rather than $\exps(\cdot)$. □

Remark. Note that Theorem 1 contradicts the assertions made in [5.10], [5.12]–[5.15] that a reset step does not affect the estimate covariance. Without a transformation of the form (5.29) any Kalman filter implementation will fail to estimate the error statistics correctly, even to first order.

Remark. Although the derivation approach is different, the result of Theorem 1 is related to the result of [5.17, eq. (28)-(29)]. For Theorem 1, however, the correction to the reference attitude (5.42b) is shown to follow as a consequence of the requirement that $\mathrm{E}(\delta_{\mathrm{post}}) = 0$, rather than being taken as an assumption.

4.2 Validation

The accuracy of the reset step may be quantified by Monte Carlo sampling, where samples from an initial attitude distribution are transformed through the proposed reset step. The sample mean and covariance of the transformed samples is then compared to the mean and covariance as computed in Theorem 1. First, a single example is analysed, which is then followed by an ensemble of Monte Carlo tests to allow for statistical evaluation of performance.

1) Single example Consider the following example, where the pre-reset variables are taken as below:

$$R_{\text{ref,pre}} = I, \quad \mu_{\text{pre}} = (0.1, 0, 0) \tag{5.43}$$

$$\Sigma_{\text{pre}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 0 \end{bmatrix} \times 10^{-2}. \tag{5.44}$$

It should be noted that there are two directions with zero uncertainty: this was chosen because the effects of the reset are made more obvious.

Pre-reset particles $\delta_{\text{mc,pre}}[i]$ are sampled independently from a normal distribution $\mathcal{N}(\mu_{\text{pre}}, \Sigma_{\text{pre}})$, with mean μ_{pre} and covariance Σ_{pre}. They are transformed to post-reset particles $\delta_{\text{mc,post}}[i]$, with

$$\delta_{\text{mc,post}}[i] := \text{rot}^{-1}\big(R_{\text{ref,post}}^{-1} \, \text{rot}(\delta_{\text{mc,pre}}[i])\big) \tag{5.45}$$

where $R_{\text{ref,post}}$ is computed as in Theorem 1 using the mean μ_{pre}. This implies that the pre- and post-reset particles, with the given $R_{\text{ref,post}}$, satisfy (5.26).

A set of 10^9 pre-reset samples $\delta_{\text{mc,pre}}[i]$ were transformed, and the sample mean and covariance of $\delta_{\text{mc,post}}[\cdot]$ were computed as:

$$\text{E}(\delta_{\text{mc,post}}[\cdot]) \approx (0, -1.92, -0.19) \times 10^{-8} \tag{5.46}$$

$$\text{Var}(\delta_{\text{mc,post}}[\cdot]) \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9.803 & -0.980 \\ 0 & -0.980 & 0.098 \end{bmatrix} \times 10^{-2}. \tag{5.47}$$

For 10^9 pre-reset samples from a normal distribution with variance 0.1, the standard deviation of the sample mean equals 10^{-5}, and the standard deviation of the sample variance equals approximately 4.4×10^{-6} [5.31, Example 7.3.3], so that a similar standard deviation may be expected for the the post-reset sample covariance.

The post-reset statistics computed with Theorem 1 are:

$$\mu_{\text{post}} = (0, 0, 0) \tag{5.48}$$

$$\Sigma_{\text{post}} \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9.901 & -0.993 \\ 0 & -0.993 & 0.100 \end{bmatrix} \times 10^{-2}. \tag{5.49}$$

Two dimensionless scalar error measures are defined for comparing the post-reset sample mean and covariance to the predicted mean and covariance: ϵ_μ is a normalised indication of the error in the mean, and ϵ_Σ is a normalised indication of the error in the

covariance:

$$\epsilon_\mu(\bar{\mu}) := \frac{|\mathrm{E}(\delta_{\mathrm{mc,post}}[\cdot]) - \bar{\mu}|}{|\mu_{\mathrm{pre}}|} \tag{5.50}$$

$$\epsilon_\Sigma(\bar{\Sigma}) := \frac{\bar{\sigma}\left(\mathrm{Var}(\delta_{\mathrm{mc,post}}[\cdot]) - \bar{\Sigma}\right)}{\bar{\sigma}\left(\Sigma_{\mathrm{pre}}\right)} \tag{5.51}$$

where $\bar{\sigma}(\cdot)$ represents the maximum singular value of its matrix argument.

For the post-reset estimates as computed in (5.48)-(5.49), the mean and covariance errors are $\epsilon_\mu \approx 1.9 \times 10^{-7}$ and $\epsilon_\Sigma \approx 9.9 \times 10^{-3}$, respectively. For this example, the mean error has a lower order of magnitude than the standard deviation of the pre-reset sample mean (normalised by the initial mean). The covariance error, however, is larger than the standard deviation of the pre-reset sample covariance – this is explained by the first-order approach to the analysis of the reset approximation, where terms of order $o(|\delta_{\mathrm{pre}}|)$ were neglected, meaning also that higher-order moments of δ do not affect the estimated mean and covariance.

Remark. If the covariance is left unchanged in the reset, as is claimed correct in [5.10], [5.12]–[5.15], the covariance error is $\epsilon_\Sigma \approx 104 \times 10^{-3}$, or more than ten times larger than with the proposed method. Furthermore, the post-reset covariances in the third row and column would be estimated as zero.

2) Ensemble validation The preceding Monte Carlo analysis may be extended by computing the error statistics $\epsilon_\mu[j]$ and $\epsilon_\Sigma[j]$ over an ensemble of initial attitude distributions, where each instance j in the ensemble consists of a large number of individual Monte Carlo samples $\delta_{\mathrm{mc,pre}}[j, i]$.

Let the variable ρ represent a magnitude, which will be used to quantify the magnitude of an ensemble: for each instance j in the ensemble, a pre-reset mean $\mu_{\mathrm{pre}}[j]$ is sampled from $\mathcal{N}(0, (\rho\,\pi/180)\,I)$, and a pre-reset covariance $\Sigma_{\mathrm{pre}}[j]$ is generated as

$$\Sigma_{\mathrm{pre}}[j] = \sum_{k=1}^{3} s_k[j] s_k[j]^T \tag{5.52}$$

where the $s_k[j]$ are also independently sampled from $\mathcal{N}(0, (\rho\,\pi/180)\,I)$. The pre-reset reference attitude is taken as identity in all instances. A million samples $\delta_{\mathrm{mc,pre}}[j, i]$ are generated from $\mathcal{N}(\mu_{\mathrm{pre}}[j], \Sigma_{\mathrm{pre}}[j])$, similarly to in Section 1, and are subsequently transformed to post-reset particles $\delta_{\mathrm{mc,post}}[j, i]$ analogously to (5.45).

Ten thousand instances j are generated in each ensemble, and for each instance the sample mean and covariance of the transformed particles are compared to the values computed with Theorem 1, similar to what was done for the single example. The distribution of the resulting errors ϵ_μ and ϵ_Σ are shown in Fig. 5.1. The figure shows the results for ensemble magnitudes $\rho \in \{1, 5, 10, 20\}$. The estimation errors are shown to increase as

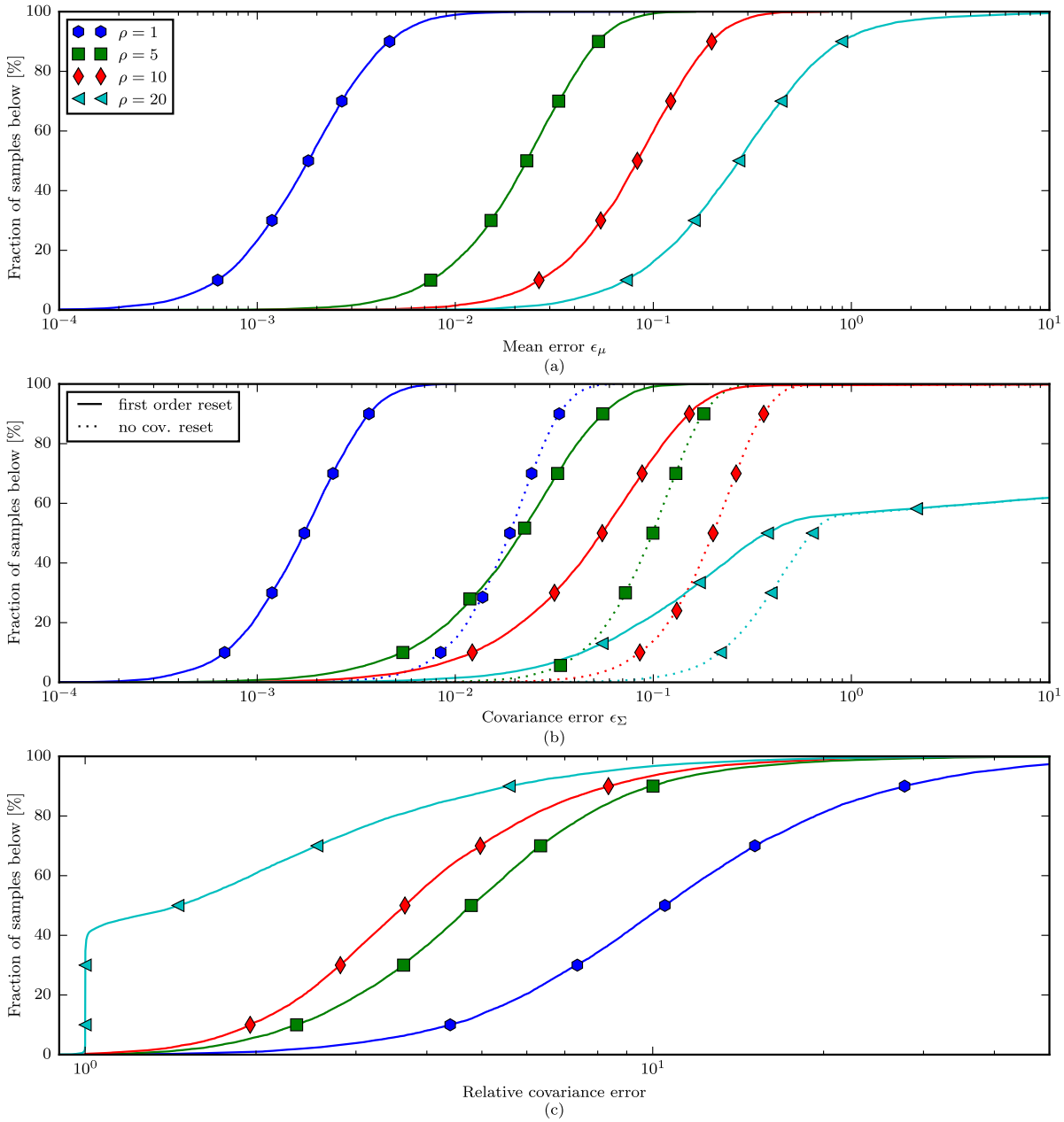

Figure 5.1. First order reset performance: The mean and covariance errors for ensembles of different magnitudes ρ, when comparing the post-reset sample mean and covariance to the predicted mean and covariance computed with the first order attitude reset. The icons are used to indicate the magnitude ρ across all sub-plots. Sub-plot (a) shows the distribution of the mean error ϵ_μ as defined in (5.50). The covariance error ϵ_Σ as defined in (5.51) is shown in sub-plot (b) for the proposed method as solid lines, and with the reset step as in [5.10] as dotted lines (where the covariance is left unchanged during the reset step). For an ensemble with $\rho = 1$, for example, for 95% of the instances the error ϵ_Σ was below 0.005 with the proposed method (as compared to 0.039 when not changing the covariance during the reset step). Sub-plot (c) shows the ratio of the covariance error ϵ_Σ for the method of [5.10] to that of the proposed method.

the ensemble magnitude ρ is increased: this is to be expected as for larger vales of ρ the higher-order terms neglected in the derivation of Theorem 1 have a larger influence.

Remark. Fig. 5.1(c) compares the performance of the reset of Theorem 1 to that proposed in [5.10], [5.12]–[5.15] where the covariance is left unchanged. The covariance error is shown to be significantly lower with the proposed method than if the covariance is unchanged during the reset – in 10% of the cases (over all values ρ) the covariance error is at least a factor 5 larger if the covariance is unchanged during the reset. For the ensemble with $\rho = 1$, the error is at least a factor 5 larger in 86% of the cases, and at least a factor 20 larger in 20% of the cases.

4.3 Extension to full state

The reset of Theorem 1 may be extended to apply to the full estimator state $x = (\xi, \delta)$ and R_{ref}. The extended reset transformation matrix $T_{\text{reset}}(\cdot)$ is introduced as

$$T_{\text{reset}}(\delta) := \text{diag}\left(I, \text{rot}\left(-\tfrac{1}{2}\delta\right)\right) \tag{5.53}$$

with $\text{diag}(\cdot)$ returning a block diagonal matrix.

The pre-reset state $x_{\text{pre}} = (\xi_{\text{pre}}, \delta_{\text{pre}})$ is introduced, which is transformed to a post-reset state x_{post} as follows:

$$x_{\text{post}} := T_{\text{reset}}(\text{E}(\delta_{\text{pre}}))x_{\text{pre}} - (0, \text{E}(\delta_{\text{pre}})) \tag{5.54}$$

from which follows

$$\text{E}(x_{\text{post}}) = (\text{E}(\xi_{\text{pre}}), 0) \tag{5.55}$$

$$\text{Var}(x_{\text{post}}) = T_{\text{reset}}(\text{E}(\delta_{\text{pre}}))\text{Var}(x_{\text{pre}})\, T_{\text{reset}}(\text{E}(\delta_{\text{pre}}))^T \tag{5.56}$$

$$R_{\text{ref,post}} = R_{\text{ref,pre}}\, \text{rot}(\text{E}(\delta_{\text{pre}}))\,. \tag{5.57}$$

All covariances related to the attitude error are affected through (5.56), i.e. the last three rows and columns of the covariance matrix. Note however that the trace of the covariance is left unchanged.

Remark. The rotation applied to the covariance through (5.56) is *half* the rotation applied to the reference attitude R_{ref} in (5.57).

Remark. The change of the covariances in (5.56) should not be thought of as a coordinate transformation due to the change of R_{ref}, as no quantities in the state ξ are expressed in the reference frame represented by R_{ref}. Instead, components of ξ may be expressed in the attitude defined by the composition of R_{ref} and δ, which remains unchanged during the reset operation to first order in the variable δ.

5. Unscented attitude reset

An alternative approximate solution to Problem 1 may be computed through the unscented transformation, which is founded on the intuition that [5.30] "it is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function or transformation". The unscented transformation proceeds by deterministically choosing a set of sigma-points, so that these points have a given mean and covariance. Each sigma point is then transformed through the given nonlinear function, and the mean and covariance of the transformed points are computed as approximations of the mean and covariance of the random variable transformed through the function.

Given an initial distribution with mean $\mu \in \mathbb{R}^n$ and covariance Σ, one set $s^{[\cdot]}$ of sigma-points $s^{[i]}$ may be selected as follows [5.1]:

$$s^{[i]} = \mu + \left(\sqrt{n\Sigma}\right)_i^T \qquad s \in \{1, \dots, n\} \tag{5.58}$$

$$s^{[n+i]} = \mu - \left(\sqrt{n\Sigma}\right)_i^T \qquad s \in \{1, \dots, n\} \tag{5.59}$$

where $\sqrt{n\Sigma}$ is a matrix square root such that $\left(\sqrt{n\Sigma}\right)^T \sqrt{n\Sigma} = n\Sigma$, and $(\cdot)_i$ is the ith row of its matrix argument. Note that alternative schemes exist for selecting sigma-points, see e.g. [5.30].

The mean and covariance, respectively, of a set of sigma-points may be computed using the functions $\mathrm{E}_\sigma(\cdot)$ and $\mathrm{Var}_\sigma(\cdot)$, defined as:

$$\mathrm{E}_\sigma\left(s^{[\cdot]}\right) := \frac{1}{2n} \sum_{i=1}^{2n} s^{[i]} \tag{5.60}$$

$$\mathrm{Var}_\sigma\left(s^{[\cdot]}\right) := \frac{1}{2n} \sum_{i=1}^{2n} \left(s^{[i]} - \mathrm{E}_\sigma\left(s^{[\cdot]}\right)\right) \left(s^{[i]} - \mathrm{E}_\sigma\left(s^{[\cdot]}\right)\right)^T. \tag{5.61}$$

These may be used as approximations for the true mean and covariance. It can be shown that this approximated mean matches the true mean of a transformed random variable correctly to second order [5.30, p. 406].

5.1 Solution approximation

Problem 1 may now be reformulated in terms of sigma-points, as follows:

Problem 2. Let the pre-reset reference attitude be $R_{\mathrm{ref,pre}}$, and the pre-reset attitude error δ_{pre} be described by a set of sigma-points $\delta_{\mathrm{pre}}^{[\cdot]}$ so that

$$\mathrm{E}_\sigma\left(\delta_{\mathrm{pre}}^{[\cdot]}\right) = \mu_{\mathrm{pre}} \tag{5.62}$$

$$\mathrm{Var}_\sigma\left(\delta_{\mathrm{pre}}^{[\cdot]}\right) = \Sigma_{\mathrm{pre}}. \tag{5.63}$$

For each pre-reset sigma point, define a post-reset point as follows, where $R_{\text{ref,post}}$ is a post-reset reference attitude. Each pair of points must satisfy (5.26), so that

$$R_{\text{ref,post}} \operatorname{rot}\left(\delta_{\text{post}}^{[i]}\right) = R_{\text{ref,pre}} \operatorname{rot}\left(\delta_{\text{pre}}^{[i]}\right) \tag{5.64}$$

or, by solving for $\delta_{\text{post}}^{[i]}$:

$$\delta_{\text{post}}^{[i]} = \operatorname{rot}^{-1}\left(R_{\text{ref,post}}^{-1} R_{\text{ref,pre}} \operatorname{rot}\left(\delta_{\text{pre}}^{[i]}\right)\right) . \tag{5.65}$$

The problem is to compute $R_{\text{ref,post}}$ subject to

$$\mathrm{E}_\sigma\left(\delta_{\text{post}}^{[\cdot]}\right) = 0. \tag{5.66}$$

A solution $R_{\text{ref,post}}$ and $\delta_{\text{post}}^{[\cdot]}$ to Problem 2 is then an approximate solution to Problem 1, with

$$\Sigma_{\text{post}} = \mathrm{Var}_\sigma\left(\delta_{\text{post}}^{[\cdot]}\right) . \tag{5.67}$$

Numerical methods may be used to compute solutions to Problem 2, by for example defining the decision variable $v \in \mathbb{R}^3$ so that $R_{\text{ref,post}} = \exp(v)$, and minimizing the error $\left|\mathrm{E}_\sigma\left(\delta_{\text{post}}^{[\cdot]}\right)\right|$.

In Procedure 1 an iterative approach to approximating a solution for Problem 2 is presented as an alternative.

1) Approximation iteration A straight-forward approach to finding an approximate solution to Problem 2 is given in Procedure 1, based on the intuition of recursively applying Theorem 1. The maximum number of iterations allowed is $N_{\max}$, $e_{\min}$ is a termination threshold, and κ is a non-increasing step size. The output is an updated reference attitude, and a set of sigma-points from which the post-reset covariance may be recovered through $\mathrm{Var}_\sigma(\cdot)$.

It should be noted that Procedure 1 has the following properties by construction:

- because the first iteration of the while-loop applies Theorem 1, the mean error $\left|\mathrm{E}_\sigma\left(\delta_{\text{best}}^{[\cdot]}\right)\right|$ of the result of Procedure 1 can be no larger than the mean error of Theorem 1,
- the procedure terminates after at most $N_{\max}$ steps.

5.2 Validation

As with the first-order reset, the accuracy of the unscented reset using Procedure 1 may be quantified by Monte Carlo sampling.

Procedure 1 Unscented reset iteration

Input a set of sigma points $\delta_{\text{pre}}^{[\cdot]}$, and $R_{\text{ref,pre}}$.
Initialise $R_{\text{ref,post}} \leftarrow R_{\text{ref,pre}}$; $\delta_{\text{best}}^{[\cdot]} \leftarrow \delta_{\text{pre}}^{[\cdot]}$; $\kappa \leftarrow 1$; $i \leftarrow 1$
repeat
  $i \leftarrow i + 1$
  $R \leftarrow R_{\text{ref,post}} \operatorname{rot}\left(\kappa \operatorname{E}_\sigma\left(\delta_{\text{best}}^{[\cdot]}\right)\right)$
  `#transformed sigma points`
  for $k \in \{1, \ldots, 2n\}$ **do**
    $\delta_{\text{post}}^{[k]} \leftarrow \operatorname{rot}^{-1}\left(R^{-1} R_{\text{ref,pre}} \operatorname{rot}\left(\delta_{\text{pre}}^{[k]}\right)\right)$
  end for
  `#compare current estimate mean`
  if $\left|\operatorname{E}_\sigma\left(\delta_{\text{post}}^{[\cdot]}\right)\right| < \left|\operatorname{E}_\sigma\left(\delta_{\text{best}}^{[\cdot]}\right)\right|$ **then**
    `#improvement`
    $\delta_{\text{best}}^{[\cdot]} \leftarrow \delta_{\text{post}}^{[\cdot]}$
    $R_{\text{ref,post}} \leftarrow R$
  else
    `#reduce step`
    $\kappa \leftarrow \frac{1}{2}\kappa$
  end if
until $\left|\operatorname{E}_\sigma\left(\delta_{\text{best}}^{[\cdot]}\right)\right| \leq e_{\min}$ or $i \geq N_{\max}$
Return $\delta_{\text{best}}^{[\cdot]}$, $R_{\text{ref,post}}$

1) Single example The example used for the first order reset in Section 1 is revisited, with the pre-reset values taken as before, repeated here from (5.43)-(5.44) for ease of comparison:

$$R_{\text{ref,pre}} = I, \quad \mu_{\text{pre}} = (0.1, 0, 0) \tag{5.68}$$

$$\Sigma_{\text{pre}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 0 \end{bmatrix} \times 10^{-2}. \tag{5.69}$$

Procedure 1 is run with the error threshold $e_{\min} = 10^{-9}$, and the procedure terminates after one step (i.e. for this problem, Procedure 1 computes the same $R_{\text{ref,post}}$ as Theorem 1). The statistics of the post-reset samples are computed as below. Note that, because for this example the $R_{\text{ref,post}}$ computed by Procedure 1 is the same as that compute by Theorem 1, the post-reset particles are the same and thus the sample statistics are also

the same as given in (5.46)-(5.47).

$$\mathrm{E}(\delta_{\mathrm{mc,post}}[\cdot]) \approx (0, -1.92, -0.19) \times 10^{-8} \tag{5.70}$$

$$\mathrm{Var}(\delta_{\mathrm{mc,post}}[\cdot]) \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9.8033 & -0.9803 \\ 0 & -0.9803 & 0.0980 \end{bmatrix} \times 10^{-2} \tag{5.71}$$

This can be compared to the values computed with Procedure 1:

$$\mu_{\mathrm{post}} = (0, 0, 0) \tag{5.72}$$

$$\Sigma_{\mathrm{post}} \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9.8030 & -0.9803 \\ 0 & -0.9803 & 0.0980 \end{bmatrix} \times 10^{-2}. \tag{5.73}$$

The mean and covariance errors for this estimate, using the errors defined in (5.50)-(5.51), are $\epsilon_\mu = 1.9 \times 10^{-7}$ and $\epsilon_\Sigma = 3.6 \times 10^{-5}$, respectively. The covariance error is approximately a factor 270 lower than the covariance error of the estimate computed with the first order correction in Section 1 in (5.46)-(5.47), and approximately a factor 2800 lower than if the covariance is left unchanged during the reset.

2) Ensemble validation As for the first order reset, an ensemble of Monte Carlo runs may be used to compute error statistics over an ensemble of initial conditions. The instances and samples are generated as described in Section 2, again as a function of the magnitude $\rho \in \{1, 5, 10, 20\}$.

The results are shown in Fig. 5.2, which compares the performance of the unscented reset computed with Procedure 1 to the first order reset of Theorem 1. Especially for larger magnitudes ρ, the unscented reset significantly outperforms the first order reset, both for the mean error ϵ_μ and the covariance error ϵ_Σ.

5.3 Extension to full state

The unscented attitude reset may be extended to apply to the full estimator state $x = (\xi, \delta)$ and R_{ref}, given a set of sigma-points $x^{[i]}$ capturing the state mean and covariance. First, Problem 2 is extended to the full state:

Problem 3. Let the pre-reset reference attitude be $R_{\mathrm{ref,pre}}$, and the pre-reset state $x_{\mathrm{pre}} = (\xi_{\mathrm{pre}}, \delta_{\mathrm{pre}})$ be described by a set of sigma-points $x_{\mathrm{pre}}^{[i]} = \left(\xi_{\mathrm{pre}}^{[i]}, \delta_{\mathrm{pre}}^{[i]}\right)$ so that

$$\mathrm{E}_\sigma\left(x_{\mathrm{pre}}^{[\cdot]}\right) = \mathrm{E}(x_{\mathrm{pre}}) \tag{5.74}$$

$$\mathrm{Var}_\sigma\left(x_{\mathrm{pre}}^{[\cdot]}\right) = \mathrm{Var}(x_{\mathrm{pre}})\,. \tag{5.75}$$

For each pre-reset sigma point, define a post-reset point as follows, where $R_{\mathrm{ref,post}}$ is a

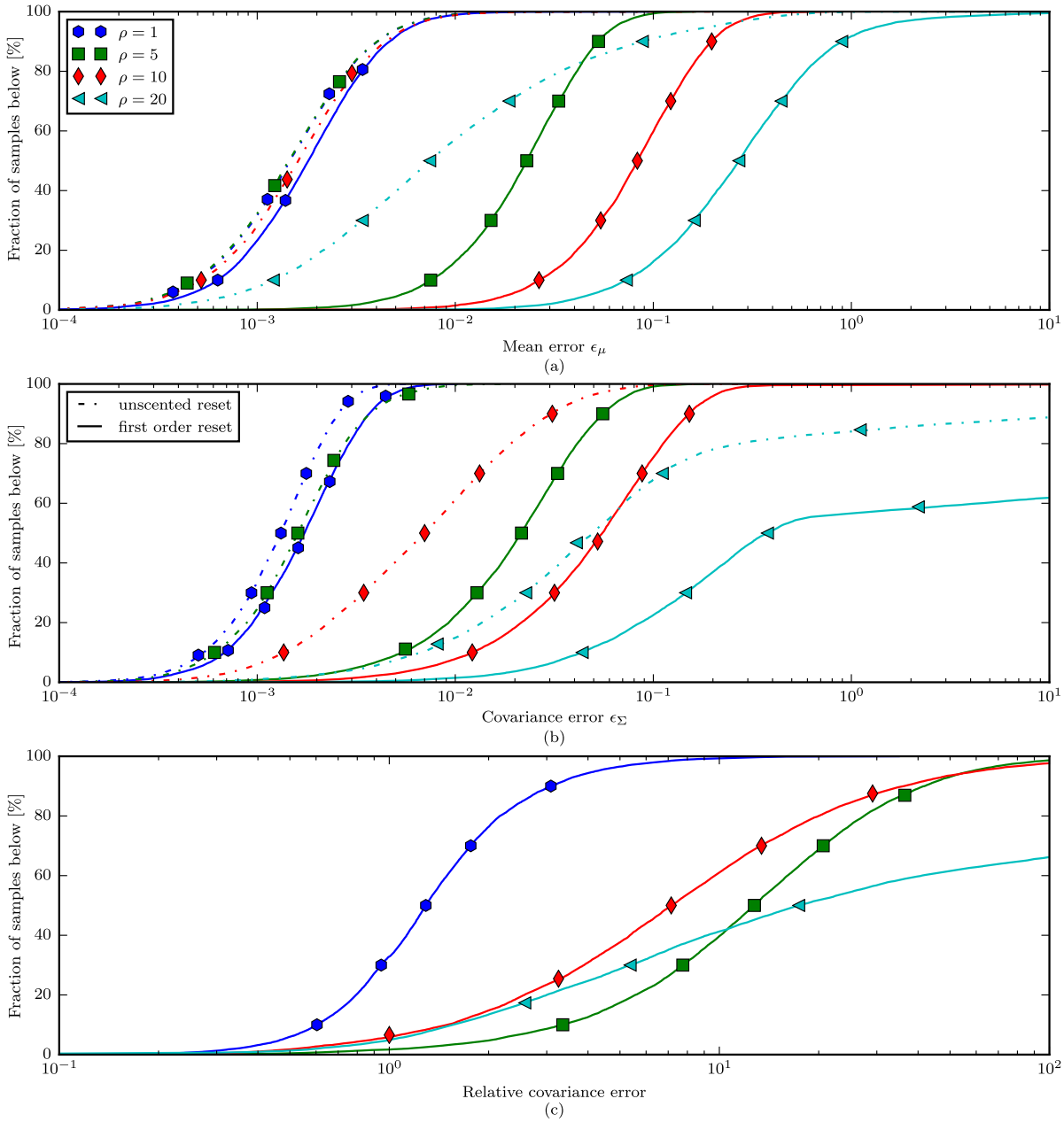

Figure 5.2. Unscented reset performance: The mean and covariance errors for ensembles of different magnitudes ρ, when comparing the post-reset sample mean and covariance to the predicted mean and covariance computed with the unscented attitude reset through Procedure 1. The iteration limit was set to $N_{\max} = 10$, and the error threshold was set to $e_{\min} = 10^{-9}$. The icons are used to indicate the magnitude ρ across all sub-plots. Sub-plot (a) shows the distribution of the mean error ϵ_μ as defined in (5.50). The covariance error ϵ_Σ as defined in (5.51) is shown in sub-plot (b) for the unscented attitude reset as solid lines, and with the first order attitude reset as dotted lines for comparison. The ratio of the covariance error made by the first order reset to the covariance error made by the unscented reset is shown in sub-plot (c).

post-reset reference attitude:

$$\delta^{[i]}_{\text{post}} = \text{rot}^{-1}\big(R^{-1}_{\text{ref,post}} R_{\text{ref,pre}}\, \text{rot}\big(\delta^{[i]}_{\text{pre}}\big)\big) \tag{5.76}$$

$$x^{[i]}_{\text{post}} = \Big(\xi^{[i]}_{\text{pre}}, \delta^{[i]}_{\text{post}}\Big)\,. \tag{5.77}$$

The problem is to compute $R_{\text{ref,post}}$ subject to

$$\text{E}_\sigma\Big(\delta^{[\cdot]}_{\text{post}}\Big) = 0. \tag{5.78}$$

The mean and covariance of the post-reset particles may then be approximated as $\text{E}_\sigma\Big(x^{[\cdot]}_{\text{post}}\Big)$ and $\text{Var}_\sigma\Big(x^{[\cdot]}_{\text{post}}\Big)$, respectively. This may again be solved by a numerical minimisation scheme. Alternatively, Procedure 1 may be extended to the full state, to compute an approximate solution to Problem 3, to yield Procedure 2. This takes as input a set of full-state sigma points and a reference attitude, and returns a post-reset set of sigma-points and a post-reset reference attitude. These post-reset sigma-points may be used to compute the post-reset covariance.

Procedure 2 Full state unscented reset

Input a set of sigma points $x^{[\cdot]}_{\text{pre}}$, where $x^{[k]}_{\text{pre}} = \Big(\xi^{[k]}_{\text{pre}}, \delta^{[k]}_{\text{pre}}\Big)$ and $R_{\text{ref,pre}}$.
`#generate post-reset attitude sigma points`
$\delta^{[\cdot]}_{\text{post}}, R_{\text{ref,post}} \leftarrow$ Procedure 1 $\Big(\delta^{[\cdot]}_{\text{pre}}, R_{\text{ref,pre}}\Big)$
for $k \in \{1, \ldots, 2n\}$ **do**
  $x^{[k]}_{\text{post}} = \Big(\xi^{[k]}_{\text{post}}, \delta^{[k]}_{\text{post}}\Big)$
end for
Return $x^{[\cdot]}_{\text{post}}$, $R_{\text{ref,post}}$

6. Algorithms

For the sake of completeness, the problem setup from Section 3 and some key results are repeated here, followed by the algorithms EKFA and UKFA.

6.1 Preliminaries

The underlying system state contains an attitude R and other states $\xi \in \mathbb{R}^{n_\xi}$. The stochastic state $x = (\xi, \delta) \in \mathbb{R}^{n_\xi+3}$ is introduced, where $\delta \in \mathbb{R}^3$ represents an attitude error parametrised through Rodrigues parameters. The attitude R associated with the system is written as the composition of a reference attitude R_{ref} and the attitude error δ:

$$R =: R_{\text{ref}}\, \text{rot}(\delta)\,. \tag{5.79}$$

The system dynamics (5.18)-(5.19) are combined and rewritten so that

$$x[k] = f(k{-}1, x[k{-}1], R_{\text{ref}}[k{-}1], \eta[k{-}1]) \tag{5.80}$$

where specifically the change in attitude of (5.19) affects only the attitude error δ, and not the reference attitude R_{ref}. Rewriting these equations requires compositions and kinematics of attitudes using the Rodrigues parameters, such as those in Section 2.

The measurement equation (5.20) is likewise rewritten as a function of the new state variables:

$$z[k] = h(k, x[k], R_{\text{ref}}[k], \zeta[k]). \tag{5.81}$$

Let $\hat{x}_m[k{-}1]$ represent the mean estimate of the state at time $k{-}1$, where specifically $\hat{\delta}_m[k{-}1] = 0$, and the associated covariance of the state is $\hat{P}_m[k{-}1]$. Let $R_{\text{ref},m}[k{-}1]$ be the associated reference attitude. These variables represent estimates of the mean and covariance of the state conditioned on all measurements up to and including time $k{-}1$.

6.2 Extended Kalman Filter with an Attitude (EKFA)

Step 0: Initialisation The filter state is initialised at time $k = 0$ using an initial mean $x_0 = (\xi_0, 0)$, covariance Σ_0, and reference attitude $R_{\text{ref},0}$:

$$\hat{x}_m[0] = x_0 \tag{5.82}$$

$$\hat{P}_m[0] = \Sigma_0 \tag{5.83}$$

$$R_{\text{ref},m}[0] = R_{\text{ref},0} \tag{5.84}$$

Step 1:Prediction The first step of the algorithm is the usual extended Kalman filter prediction step, as can be found in e.g. [5.29]. For the sake of completeness, the main ideas are presented here. This step propagates the estimate $\hat{x}_m[k{-}1]$ and $\hat{P}_m[k{-}1]$ through the dynamic equations, to yield predicted estimates $\hat{x}_p[k]$ and $\hat{P}_p[k]$, respectively. Important is that the reference attitude is left unchanged in this step.

$$\hat{x}_p[k] := f(k{-}1, \hat{x}_m[k{-}1], R_{\text{ref},m}[k{-}1], 0) \tag{5.85}$$

$$\hat{P}_p[k] := F[k{-}1]\hat{P}_m[k{-}1]F[k{-}1]^T + G[k{-}1]\Sigma_\eta[k{-}1]G[k{-}1]^T \tag{5.86}$$

$$R_{\text{ref},p}[k] := R_{\text{ref},m}[k{-}1] \tag{5.87}$$

where

$$F[k] := \left.\frac{\partial f}{\partial x}\right|_{x=\hat{x}_m[k], R_{\text{ref}}=R_{\text{ref},m}[k], \eta[k]=0} \tag{5.88}$$

$$G[k] := \left.\frac{\partial f}{\partial \eta}\right|_{x=\hat{x}_m[k], R_{\text{ref}}=R_{\text{ref},m}[k], \eta[k]=0} \tag{5.89}$$

$$\Sigma_\eta[k] := \text{Var}(\eta[k])\,. \tag{5.90}$$

Step 2: Prediction reset After the prediction step, the state estimate $\hat{x}_p[k]$ will in general contain a non-zero attitude error $\hat{\delta}_p[k]$. The goal of this step is to move the information from this attitude error into the reference attitude, following the method of Theorem 1. This yields the mean estimate $\hat{x}_q[k]$, associated covariance $\hat{P}_q[k]$ and reference attitude $R_{\text{ref},q}[k]$:

$$\hat{x}_q[k] := \hat{x}_p[k] - \left(0, \hat{\delta}_p[k]\right) \tag{5.91}$$

$$\hat{P}_q[k] := T_{\text{reset}}(\hat{\delta}_p[k])\hat{P}_p[k]T_{\text{reset}}(\hat{\delta}_p[k])^T \tag{5.92}$$

$$R_{\text{ref},q}[k] := R_{\text{ref},p}[k]\text{rot}\left(\hat{\delta}_p[k]\right) \tag{5.93}$$

where

$$T_{\text{reset}}\,(\delta) := \text{diag}\left(I, \text{rot}\left(-\tfrac{1}{2}\delta\right)\right). \tag{5.94}$$

Step 3: Measurement update This step is the usual Kalman filter measurement update step, again repeated here for completeness from [5.29]. Again, the reference attitude is left unchanged in this step.

$$L[k] := \hat{P}_q[k]H[k]^T\left(H[k]\hat{P}_q[k]H[k]^T + M[k]\Sigma_\zeta[k]M[k]^T\right)^{-1} \tag{5.95}$$

$$\hat{x}_r[k] := \hat{x}_q[k] + L[k]\left(z[k] - h(k, \hat{x}_q[k], R_{\text{ref},q}[k], 0)\right) \tag{5.96}$$

$$\hat{P}_r[k] := (I - L[k]H[k])\hat{P}_q[k] \tag{5.97}$$

$$R_{\text{ref},r}[k] := R_{\text{ref},q}[k] \tag{5.98}$$

where

$$H[k] := \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}_r[k], R_{\mathrm{ref}}=R_{\mathrm{ref},q}[k], \zeta[k]=0} \tag{5.99}$$

$$M[k] := \left.\frac{\partial h}{\partial \zeta}\right|_{x=\hat{x}_m[k], R_{\mathrm{ref}}=R_{\mathrm{ref},m}[k], \zeta[k]=0} \tag{5.100}$$

$$\Sigma_\zeta[k] := \mathrm{Var}(\zeta[k])\,. \tag{5.101}$$

Step 4: Measurement reset After the measurement update step, the state estimate $\hat{x}_r[k]$ will again in general contain a non-zero attitude error $\hat{\delta}_r[k]$. This is again moved to the reference attitude using the method of Theorem 1. This yields the mean estimate $\hat{x}_m[k]$, associated covariance $\hat{P}_m[k]$ and reference attitude $R_{\mathrm{ref},m}[k]$ that may be used for the next prediction step:

$$\hat{x}_m[k] := \hat{x}_r[k] - \left(0, \hat{\delta}_r[k]\right) \tag{5.102}$$

$$\hat{P}_m[k] := T_{\mathrm{reset}}(\hat{\delta}_r[k])\hat{P}_r[k]T_{\mathrm{reset}}(\hat{\delta}_r[k])^T \tag{5.103}$$

$$R_{\mathrm{ref},m}[k] := R_{\mathrm{ref},r}[k]\mathrm{rot}\left(\hat{\delta}_r[k]\right)\,. \tag{5.104}$$

The algorithm continues from Step 1.

6.3 Unscented Kalman Filter with an Attitude (UKFA)

The unscented Kalman filter offers much flexibility for encoding problems, e.g. allowing to encode process or measurement noise that enters the system in a nonlinear manner. For the sake of simpler exposition, here only the most straight-forward case will be used for the presentation of the UKFA: it will be assumed that the process noise η enters the dynamics in a purely additive way and that the dimension of η is the same as the state x, so that its effect on the covariance estimate is captured by a simple addition. Likewise, it is assumed that the measurement noise ζ enters in a purely additive way. If these assumptions do not hold, the algorithm may be straight-forwardly modified, see e.g. [5.30].

Step 0: Initialisation The filter state is initialised at time $k = 0$ using an initial mean $x_0 = (\xi_0, 0)$, covariance Σ_0, and reference attitude $R_{\mathrm{ref},0}$:

$$\hat{x}_m[0] = x_0 \tag{5.105}$$

$$\hat{P}_m[0] = \Sigma_0 \tag{5.106}$$

$$R_{\mathrm{ref},m}[0] = R_{\mathrm{ref},0} \tag{5.107}$$

Step 1: Prediction The first step of the algorithm is a modification of the usual unscented Kalman filter prediction step, as can be found in e.g. [5.1]. This step propagates the estimate $\hat{x}_m[k{-}1]$ and $\hat{P}_m[k{-}1]$ through the dynamic equations, to yield predicted estimates $\hat{x}_p[k]$ and $\hat{P}_p[k]$, respectively. Important is that the reference attitude is left unchanged in this step.

First a set of sigma-points $\hat{x}_p^{[\cdot]}$ is computed as

$$\hat{x}_m^{[\cdot]}[k{-}1]{:=}\text{GenerateSigmaPoints}\left(\hat{x}_m[k{-}1], \hat{P}_m[k{-}1]\right) \tag{5.108}$$

where $\text{GenerateSigmaPoints}\,(\mu, \Sigma)$ is a function that generates a set of sigma points as follows, where n is the dimension of μ:

$$s^{[i]} = \mu + \left(\sqrt{n\Sigma}\right)_i^T \qquad s \in \{1, \ldots, n\} \tag{5.109}$$

$$s^{[n+i]} = \mu - \left(\sqrt{n\Sigma}\right)_i^T \qquad s \in \{1, \ldots, n\}. \tag{5.110}$$

Each sigma-point is then propagated through the nonlinear dynamics equation as

$$\hat{x}_p^{[i]}[k]{:=}f(k{-}1, \hat{x}_m^{[i]}[k{-}1], R_{\text{ref},m}[k{-}1], 0). \tag{5.111}$$

Step 2: Prediction reset After the prediction step, the sigma-points $\hat{x}_p^{[\cdot]}[k]$ will in general contain a non-zero mean attitude error. The goal of this step is to move the information from this attitude error into the reference attitude, following the method derived in Procedure 2, such that for each particle $\hat{x}_p^{[i]}$ a new particle $\hat{x}_q^{[i]}$ is computed, as well as the reference attitude $R_{\text{ref},q}$. The effect of the process noise η is also taken into account.

$$\hat{x}_p^{[\cdot]}, R_{\text{ref},q} = \text{Procedure 2}\left(\hat{x}_p^{[\cdot]}, R_{\text{ref},p}\right) \tag{5.112}$$

From this, the estimated predicted statistics follow as

$$\hat{x}_q[k]{:=}\text{E}_\sigma\left(\hat{x}_q^{[\cdot]}[k]\right) \tag{5.113}$$

$$\hat{P}_q[k]{:=}\text{Var}_\sigma\left(\hat{x}_q^{[\cdot]}[k]\right) + \Sigma_\eta[k{-}1] \tag{5.114}$$

where

$$\Sigma_\eta[k]{:=}\text{Var}(\eta[k])\,. \tag{5.115}$$

Step 3: Measurement update This step is a modification of the usual unscented Kalman filter measurement update step, as can be found in [5.1]. Again, the reference attitude is

left unchanged in this step.

A set of sigma-points $\hat{x}_r^{[\cdot]}$ is generated as

$$\hat{x}_r^{[\cdot]}[k{-}1]{:=}\text{GenerateSigmaPoints}\left(\hat{x}_q[k{-}1], \hat{P}_q[k{-}1]\right). \tag{5.116}$$

Note that this may be omitted, and the sigma points from the previous step directly used, at the cost of reduced performance [5.1].

Each sigma-point is then propagated through the nonlinear measurement equation as

$$\hat{z}^{[i]}[k]{:=}h(k, \hat{x}_q^{[i]}[k], R_{\text{ref},q}[k], 0). \tag{5.117}$$

These sigma-points are combined to yield the predicted measurement statistics:

$$\hat{z}[k]{:=}\text{E}_\sigma\left(\hat{z}^{[\cdot]}\right) \tag{5.118}$$

$$\hat{P}_z[k]{:=}\text{Var}_\sigma\left(\hat{z}^{[\cdot]}\right) + \Sigma_\zeta[k] \tag{5.119}$$

where

$$\Sigma_\zeta[k]{:=}\text{Var}(\zeta[k]). \tag{5.120}$$

The cross covariance between $\hat{x}_r$ and $\hat{z}$ is estimated as

$$\hat{P}_{xz}[k]{:=}\frac{1}{2n}\sum_{i=1}^{2n}\left(\hat{x}_r^{[i]}[k]{-}\text{E}_\sigma\left(\hat{x}_r^{[\cdot]}[k]\right)\right)\left(\hat{z}^{[i]}[k]{-}\text{E}_\sigma\left(\hat{z}^{[\cdot]}[k]\right)\right)^T \tag{5.121}$$

The measurement $z[k]$ is then used as

$$L[k]{:=}\hat{P}_{xz}\hat{P}_z^{-1} \tag{5.122}$$

$$\hat{x}_r[k] := \hat{x}_q[k] + L[k]\left(z[k] - \hat{z}[k]\right) \tag{5.123}$$

$$\hat{P}_r[k] := \hat{P}_q[k] - L[k]\hat{P}_z[k]L[k]^T \tag{5.124}$$

$$R_{\text{ref},r}[k] := R_{\text{ref},q}[k]. \tag{5.125}$$

Step 4: Measurement reset After the measurement update step, the state estimate $\hat{x}_r[k]$ will again in general contain a non-zero attitude error $\hat{\delta}_r[k]$. This is again moved to the reference attitude by generating a new set of sigma-points, and using Procedure 2. The result of the reset step is the mean estimate $\hat{x}_m[k]$, the variance estimate $\hat{P}_m[k]$, and the reference attitude $R_{\text{ref},m}[k]$. These are used in the next iteration, and the algorithm continues from Step 1.

6.4 Alternative estimator structures

In addition to the presented EKFA and UKFA algorithms, alternative combinations are also possible. For example, the first order reset of Theorem 1 may be used in an unscented Kalman filter, or the unscented reset of Procedure 2 may be used with an extended Kalman filter. Alternatively, one form of reset may be used for the prediction reset, another for the measurement reset.

7. Conclusion

The presented algorithm is a generic framework for applying the extended or unscented Kalman filter to systems whose dynamic state includes an attitude. A three element attitude error is used in the estimate state vector (which has an associated covariance), in addition to a reference attitude (which does not have associated covariance). A key requirement for such an algorithm is a mathematical basis for moving information from the attitude error to the reference attitude, in a reset step. This is done by keeping track of the attitude error statistics during the reset step. Two approximations to the reset step are proposed, based on a first-order analysis and the unscented transform. Monte Carlo sampling is used to validate the reset steps, and it is shown that the first order reset has significantly lower errors than the method in the prior art. Furthermore, at the cost of increased computational complexity, the reset step based on the unscented transform is shown to significantly out-perform the first-order reset. These reset steps are used in two novel estimation algorithms, the Extended Kalman Filter with an Attitude (EKFA), and the Unscented Kalman Filter with an Attitude (UKFA). The reset steps may be extended straight-forwardly to problems containing multiple attitudes, e.g. a robotic arm with multiple serial joints.

Funding

This research was supported by the Swiss National Science Foundation (SNSF), under grant application 138112.

A. Comparison of EKFA and MEKF

This appendix compares the EKFA to the MEKF of [5.10]. It is shown that, for the system considered by the MEKF, the prediction step of the MEKF is equivalent to the prediction step of the EKFA. This section will not, however, attempt a detailed description of the MEKF.

A.1 Problem statement

The problem considered is to estimate an attitude of a rigid body, where the attitude R represents the transformation from a body-fixed frame to an inertial, world-fixed frame. The process model consists of integrating the output of a rate gyroscope g (which differs from the true angular velocity by a bias b). Additional vector measurements, used in the Kalman filter measurement update, are available. However, because the focus is on the Kalman prediction step, these are ignored here. Noise on the rate gyroscope output, and noise driving the bias, are also neglected here for simplicity.

The differential equation governing the dynamics of the bias and the orientation are then

$$\frac{d}{dt}b(t) = 0 \tag{5.126}$$

$$\frac{d}{dt}R(t) = R(t)\,\mathrm{S}(g(t) - b(t))\,. \tag{5.127}$$

Using Fact 4, the above may be discretised over a time period $\Delta t \ll 1$ by assuming that the angular velocity is constant over the sampling period, to yield

$$b[k] = b[k{-}1] \tag{5.128}$$

$$R[k] = R[k{-}1]\exps((g[k] - b[k{-}1])\,\Delta t)\,. \tag{5.129}$$

A.2 MEKF

The MEKF state vector is six dimensional, and includes a three-dimensional attitude error representation δ, and a three dimensional rate gyroscope bias b. The derivation in [5.10] is done on a continuous time process model: to aid comparison to the EKFA, the MEKF is here discretised with a zero-order hold over the time step Δt. The rate gyroscope output at time k is $g[k]$, and the state estimate is $\hat{x} = \left(\hat{b}, \hat{\delta}\right)$.

It should be noted that in [5.10] the reference attitude is encoded using the Euler symmetric parameters, this however has no effect on the following. Furthermore, the attitude error is encoded using twice the Rodrigues parameters, so that some factors of a half appear additionally below.

The prediction of the MEKF may then be written as below, using the per-step symbols of Section 6:

$$\hat{x}_{q,\mathrm{M}}[k] := \left(\hat{b}_{m,\mathrm{M}}[k{-}1], 0\right) \tag{5.130}$$

$$R_{\mathrm{ref},q,\mathrm{M}}[k] = R_{\mathrm{ref},p,\mathrm{M}}[k]\exps\left(\left(g[k] - \hat{b}_{q,\mathrm{M}}[k]\right)\Delta t\right) \tag{5.131}$$

That is, the bias estimate remains constant, the attitude error remains zero, and the reference attitude is updated using the bias-corrected rate gyroscope output. The covariance

prediction is:

$$\hat{P}_{q,\mathrm{M}}[k] = F_{\mathrm{M}}(g[k] - \hat{b}_{m,\mathrm{M}}[k{-}1])\hat{P}_{P,\mathrm{M}}[k]F_{\mathrm{M}}(g[k] - \hat{b}_{m,\mathrm{M}}[k{-}1])^T \tag{5.132}$$

where $F_{\mathrm{M}}(\omega)$ is given as below [5.10, see (38)]

$$F_{\mathrm{M}}(\omega) := \begin{bmatrix} I & 0 \\ -\frac{1}{2}I\Delta t & I - \mathrm{S}(\omega\Delta t) \end{bmatrix}. \tag{5.133}$$

A.3 EKFA

This can be compared to the prediction step of the EKFA, from Section 6: The first step (prediction) yields the below, using again a first-order Euler sampling to discretise the dynamics:

$$\hat{x}_p[k] := \left(\hat{b}_m[k{-}1], \mathrm{rot}^{-1}\left(\exp\left(\left(g[k] - \hat{b}_m[k{-}1]\right)\Delta t\right)\right)\right) \tag{5.134}$$

$$\hat{P}_p[k] := F[k{-}1]\hat{P}_m[k{-}1]F[k{-}1]^T \tag{5.135}$$

$$R_{\mathrm{ref},p}[k] := R_{\mathrm{ref},m}[k{-}1] \tag{5.136}$$

where F is defined as below, using the differential equation (5.17) for the partial derivatives

$$F[k] := \begin{bmatrix} I & 0 \\ -\frac{1}{2}I\Delta t & I - \frac{1}{2}\mathrm{S}\left(\left(g[k] - \hat{b}[k{-}1]\right)\Delta t\right) \end{bmatrix}. \tag{5.137}$$

The prediction reset step then follows as

$$\begin{aligned} \hat{x}_q[k] &:= \left(\hat{b}_p[k], 0\right) \\ &= \left(\hat{b}_m[k{-}1], 0\right) \end{aligned} \tag{5.138}$$

$$\begin{aligned} \hat{P}_q[k] &:= T_{\mathrm{reset}}\left(\hat{\delta}_p[k]\right)\hat{P}_p[k]T_{\mathrm{reset}}\left(\hat{\delta}_p[k]\right)^T \\ &:= T_{\mathrm{reset}}F[k]\left(\hat{\delta}_p[k]\right)\hat{P}_m[k{-}1]F[k]^T T_{\mathrm{reset}}\left(\hat{\delta}_p[k]\right)^T \end{aligned} \tag{5.139}$$

$$\begin{aligned} R_{\mathrm{ref},q}[k] &:= R_{\mathrm{ref},p}[k{-}1]\mathrm{rot}\left(\hat{\delta}_p[k]\right) \\ &= R_{\mathrm{ref},p}\exp\left(\left(g[k] - \hat{b}_m[k{-}1]\right)\Delta t\right) \end{aligned} \tag{5.140}$$

where $\mathrm{rot}\Big(\hat{\delta}[k]\Big)$ can be approximated to first order, yielding

$$\mathrm{rot}\Big(\hat{\delta}[k]\Big) = I - \mathrm{S}\Big(g[k] - \hat{b}[k{-}1]\Big)\,\Delta t + o(\Delta t) \tag{5.141}$$

$$T_{\mathrm{reset}}\left(\hat{\delta}[k]\right) F[k] = \begin{bmatrix} I & 0 \\ -\frac{1}{2}I\Delta t & I - \mathrm{S}\Big(g[k] - \hat{b}[k{-}1]\Big)\,\Delta t \end{bmatrix} + o(\Delta t)\,. \tag{5.142}$$

A.4 Comparison

By comparing (5.133) and (5.142) it can be seen that the MEKF contains an implicit covariance reset in the prediction step, equivalent to that of Theorem 1. Therefore, for this system, the MEKF and the EKFA are equivalent for the prediction step. Note, however, that the EKFA contains an additional reset after the measurement update. Furthermore, the reset of Theorem 1 may be straight-forwardly extended to alternative dynamic models, with more complex state vectors.

Remark. The innovation (measurement residual) sequence for a Kalman filter is a zero-mean, white sequence [5.8, Sec. 5.2.6]. Because the state updates due to the measurements are linear in the innovations, the state update sequence is thus also zero-mean and white. Therefore, for a properly tuned MEKF, with "slow" dynamics and low uncertainty (so that nonlinear effects are negligible), the measurement covariance reset will be zero-mean and white, so that failure to do the measurement reset may be expected to have a small effect. If instead the process/measurement noise characteristics are not known exactly, or nonlinear effects are significant, the state updates will no longer be white (and may also be non-zero mean), so that the measurement reset step may significantly improve performance.

Acknowledgements

The authors would like to thank Mark Psiaki and Landis Markley for their feedback and helpful comments to this paper.

References

[5.1] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[5.2] J. Stuelpnagel, "On the parametrization of the three-dimensional rotation group", *SIAM review*, vol. 6, no. 4, pp. 422–430, 1964.

[5.3] Y. Yoon, A. Kosaka, and A. C. Kak, "A new Kalman-filter-based framework for fast and accurate visual tracking of rigid objects", *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1238–1251, 2008.

[5.4] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera", in *International Conference on Computer Vision*, IEEE, 2003, pp. 1403–1410.

[5.5] M. D. Shuster, "A survey of attitude representations", *Journal of the Astronautical Sciences*, vol. 41, no. 4, pp. 439–517, 1993.

[5.6] S. J. Julier and J. J. LaViola Jr, "On Kalman filtering with nonlinear equality constraints", *IEEE Transactions on Signal Processing*, vol. 55, no. 6, pp. 2774–2784, 2007.

[5.7] R. Zanetti, M. Majji, R. H. Bishop, and D. Mortari, "Norm-constrained Kalman filtering", *Journal of guidance, control, and dynamics*, vol. 32, no. 5, pp. 1458–1465, 2009.

[5.8] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation*. John Wiley & Sons, 2004.

[5.9] J. L. Farrell, "Attitude determination by Kalman filtering", *Automatica*, vol. 6, no. 3, pp. 419–430, 1970.

[5.10] F. L. Markley, "Attitude error representations for Kalman filtering", *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[5.11] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation", *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[5.12] J. L. Crassidis and F. L. Markley, "Unscented filtering for spacecraft attitude estimation", *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 4, pp. 536–542, 2003.

[5.13] F. L. Markley, J. L. Crassidis, and Y. Cheng, "Nonlinear attitude filtering methods", in *AIAA Guidance, Navigation, and Control Conference*, 2005, pp. 15–18.

[5.14] J. L. Crassidis, F. L. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods", *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[5.15] F. L. Markley and J. L. Crassidis, "Filtering for attitude estimation and calibration", in *Fundamentals of Spacecraft Attitude Determination and Control*, Springer, 2014, pp. 235–285.

[5.16] S. Steffes, J. P. Steinbach, and S. Theil, "Investigation of the attitude error vector reference frame in the INS EKF", in *Advances in Aerospace Guidance, Navigation and Control*, Springer, 2011, pp. 345–357.

[5.17] R. G. Reynolds, "Asymptotically optimal attitude filtering with guaranteed convergence", *Journal of guidance, control, and dynamics*, vol. 31, no. 1, pp. 114–122, 2008.

[5.18] F. L. Markley, "Lessons learned", English, *The Journal of the Astronautical Sciences*, vol. 57, no. 1-2, pp. 3–29, 2009.

[5.19] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation", *University of Minnesota, Deptartment of Computer Science & Engineering, Report 2005-002*, Jan. 2005.

[5.20] J. K. Hall, N. B. Knoebel, and T. W. McLain, "Quaternion attitude estimation for miniature air vehicles using a multiplicative extended kalman filter", in *IEEE/ION Position, Location and Navigation Symposium*, IEEE, 2008, pp. 1230–1237.

[5.21] N. Filipe, M. Kontitsis, and P. Tsiotras, "Extended Kalman filter for spacecraft pose estimation using dual quaternions", *Journal of Guidance, Control, and Dynamics*, 2015.

[5.22] J. C. Springmann, A. J. Sloboda, A. T. Klesh, M. W. Bennett, and J. W. Cutler, "The attitude determination system of the RAX satellite", *Acta Astronautica*, vol. 75, pp. 120–135, 2012.

[5.23] M. D. Truch, P. A. Ade, J. J. Bock, E. L. Chapin, M. J. Devlin, S. R. Dicker, M. Griffin, J. O. Gundersen, M. Halpern, P. C. Hargrave, *et al.*, "The balloon-borne large aperture submillimeter telescope (BLAST) 2006: Calibration and flight performance", *The Astrophysical Journal*, vol. 707, no. 2, p. 1723, 2009.

[5.24] F. M. Mirzaei, S. Roumeliotis, *et al.*, "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation", *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1143–1156, 2008.

[5.25] A. Mourikis, N. Trawny, S. Roumeliotis, A. E. Johnson, A. Ansar, L. Matthies, *et al.*, "Vision-aided inertial navigation for spacecraft entry, descent, and landing", *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 264–280, 2009.

[5.26] M. D. Shuster, "Constraint in attitude estimation part I: Constrained estimation", *Journal of the Astronautical Sciences*, vol. 51, no. 1, pp. 51–74, 2003.

[5.27] M. D. Shuster, "Constraint in attitude estimation part II: Unconstrained estimation", *Journal of the Astronautical Sciences*, vol. 51, no. 1, pp. 75–102, 2003.

[5.28] D. S. Bernstein, *Matrix mathematics: Theory, facts, and formulas*. Princeton University Press, 2009.

[5.29] B. D. O. Anderson and J. B. Moore, *Optimal filtering*. Prentice-Hall, 1979.

[5.30] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation", *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[5.31] G. Casella and R. L. Berger, *Statistical inference*. Duxbury Pacific Grove, CA, 2002, vol. 2.

Paper P6

Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation

Mark W. Mueller, Michael Hamer, and Raffaello D'Andrea

Abstract

A state estimator for a quadrocopter is presented, using measurements from an accelerometer, angular rate gyroscope, and a set of ultra-wideband ranging radios. The estimator uses an extended aerodynamic model for the quadrocopter, where the full 3D airspeed is observable through accelerometer measurements. The remaining quadrocopter states, including the yaw orientation, are rendered observable by fusing ultra-wideband range measurements, under the assumption of no wind. The estimator is implemented on a standard microcontroller using readily-available, low-cost sensors. Performance is experimentally investigated in a variety of scenarios, where the quadrocopter is flown under feedback control using the estimator output.

Published in *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA).*

1. Introduction

Quadrocopters offer agile and mechanically simple testbeds for research on flying robots. They also promise to be useful for commercial applications, for example as package delivery robots [6.1] or as mobile sensing platforms.

State estimation is a fundamental requirement for the autonomous operation of these vehicles. Typical estimation strategies fuse a variety of sensors, typically combining inertial measurement units with sensors providing absolute measurements.

A number of solutions for absolute state measurement currently exist. Off-board vision is often used to measure the position and orientations of flying vehicles, using either commercial motion-capture systems [6.2]–[6.4], or lower-cost cameras [6.5], [6.6]. On-board vision systems are also very popular, with [6.7]–[6.9] being examples of monocular-, and [6.10], [6.11] of stereo-vision systems. Finally, many outdoor systems rely on GPS systems for position measurements, as e.g. [6.12]. These different localisation solutions present different trade-offs between cost, measurement accuracy, robustness to external influences, computational burden, and ease of deployment.

A relatively new method of indoor localization utilizes low cost, low power, ultra-wideband (UWB) radio modules to estimate inter-module distance by measuring the transmission and reception time of UWB pulses, see e.g. [6.13]–[6.15]. This paper presents a quadrocopter state estimation strategy, which uses these range measurements to localize the quadrocopter. This is enabled by a mobile UWB radio connected to the quadrocopter and a set of fixed modules with known position (anchors) placed in the environment. Inter-module distance is measured using a time-of-arrival (TOA) approach. The layout of the system is illustrated in Fig. 6.1.

This paper demonstrates that closed loop control of a quadrocopter during agile manoeuvres is possible using UWB range measurements fused with a dynamic model of the quadrocopter and with measurements from on-board accelerometers and rate gyroscopes. The state estimator, controller, and trajectory generator all run on-board the quadrocopter's microcontroller.

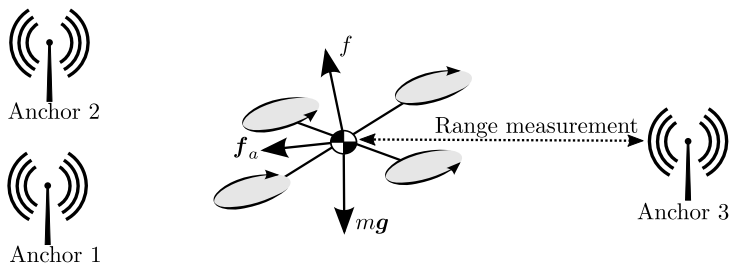

Figure 6.1. A quadrocopter uses time of flight measurements from an UWB radio to measure the distance to a set of stationary UWB anchors in a round-robin fashion. An accelerometer effectively measures the static thrust f and other aerodynamic effects $\boldsymbol{f}_a$, thereby providing information on the quadrocopter's translational velocity. Rate gyroscopes are used to measure the quadrocopter's angular velocity.

The remainder of this paper is organised as follows: the dynamics of the quadrocopter are presented in Section 2, with a special focus on aerodynamic effects. Section 3 then presents sensor models for each of the three sensors used, and Section 4 presents a state estimator fusing these measurements with a dynamic model of the quadrocopter. An algorithm to estimate the range between two UWB radios is then given in Section 5. The approach is validated in experiment in Section 6, and the paper concludes with Section 7.

2. System dynamics

This section presents the equations of motion governing the flight of a quadrocopter. These equations will later be used in the prediction step of the state estimator. Furthermore, specific attention will be paid to an accelerometer model, showing how the vehicle's three dimensional velocity may be directly inferred from the accelerometer measurement.

The quadrocopter is modelled as a rigid body, governed by the Newton-Euler equations [6.16]. Denoting the quadrocopter's position in an inertial reference frame with $\boldsymbol{x}$ and the orientation of the quadrocopter's body with respect to the inertial reference frame with $\mathbf{R}$, the equations of motion of the quadrocopter are as below. The total thrust produced by the propellers is written as f, and the acceleration due to gravity is expressed as $\boldsymbol{g}$. All remaining aerodynamic effects are captured in the vector $\boldsymbol{f}_a$, whose components are expressed in the body-fixed frame. The vehicle's mass is given by m, and $\boldsymbol{e}_3 = (0, 0, 1)$. The notation (x, y, z) will be used throughout this paper to compactly express the elements of a vector. The forces are illustrated in Fig. 6.1.

$$m\ddot{\boldsymbol{x}} = \mathbf{R}\ (f\boldsymbol{e}_3 + \boldsymbol{f}_a) + m\boldsymbol{g} \tag{6.1}$$

$$\dot{\mathbf{R}} = \mathbf{R}\ [\![\boldsymbol{\omega}\times]\!] \tag{6.2}$$

Note that this uses the matrix form of the cross product, given by

$$[\![\boldsymbol{\omega}\times]\!] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{6.3}$$

where $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)$ is the angular velocity of the quadrocopter as expressed in the body fixed frame.

The angular acceleration of the quadrocopter evolves as a function of the torques acting on the vehicle, the current angular velocity, and the quadrocopter's inertia. These equations are given for example in [6.17], and will not be repeated here as these equations are not required for the estimator design in Section 4.

2.1 Static thrust force

It is assumed that the angular velocity $\dot{\theta}_i$ of each propeller i is known, for example as a measurement returned by the electronic speed controller. The thrust produced by a stationary propeller may be modelled as quadratic in its angular velocity, with proportionality constant κ [6.18], so that the total thrust of the the vehicle is

$$f = \sum_{i=1}^{4} \kappa \dot{\theta}_i^2. \tag{6.4}$$

This static thrust points along the propellers' axis of rotation.

2.2 Aerodynamic effects

The force generated by a propeller translating with respect to the free stream will typically be significantly different from the static thrust force f. This deviation is given by $\boldsymbol{f}_a$, which is taken to be a function of the quadrocopter's relative airspeed.

The effect of the component of the airspeed in the rotor plane is well documented, e.g. [6.17], [6.19]–[6.21]. Typically, blade flapping is invoked to motivate a linear dependence between the quadrocopter's velocity in the rotor plane to the components of $\boldsymbol{f}_a$ in the rotor plane.

There is rich literature on the aerodynamics of propellers translating along their axial directions [6.22], typically in the context of large, fixed-wing aeroplanes. The dependence of the thrust produced by a quadrocopter translating in the thrust direction has also been studied in the literature, for example [6.23], [6.24].

Here, the aerodynamic force is modelled as an interaction between the rotating propellers, and the vehicle's airspeed. The force is assumed to be linear in the product of the airspeed and the propeller speeds, with $\kappa_\perp$ the proportionality constant for the force in the plane of the rotors, and $\kappa_\parallel$ the proportionality constant in the direction of the thrust vector. It is assumed that there is no wind, so that the quadrocopter's relative airspeed equals the quadrocopter's velocity with respect to the inertial frame.

The force can then be calculated as

$$\boldsymbol{f}_a = \mathbf{K}_{\text{aero}}\, \dot{\theta}_\Sigma\, \mathbf{R}^{-1} \dot{\boldsymbol{x}} \tag{6.5}$$

where

$$\mathbf{K}_{\text{aero}} = \operatorname{diag}\left(\kappa_\perp, \kappa_\perp, \kappa_\parallel\right) \tag{6.6}$$

$$\dot{\theta}_\Sigma = \sum_{i=1}^{4} \left|\dot{\theta}_i\right|. \tag{6.7}$$

3. Sensors

The estimator presented in Section 4 relies on the measurements of three distinct types of sensors: angular rate gyroscopes, accelerometers, and UWB range sensors. Each of these sensors will be briefly discussed here, and the sensor output will be linked to the quadrocopter dynamics of Section 2.

3.1 Angular rate gyroscopes

The angular rate gyroscopes measure the quadrocopter's angular velocity in the body frame. The measurement will be modelled here as

$$\boldsymbol{z}_{\text{gyro}} = \boldsymbol{\omega} + \boldsymbol{\eta}_{\text{gyro}} \tag{6.8}$$

where $\boldsymbol{\eta}_{\text{gyro}}$ is assumed to be zero-mean white noise. More complete models exist, which include for example scale errors or biases [6.25], here however it will be assumed that the sensor is well calibrated and these effects may be neglected.

3.2 Accelerometers

An accelerometer measures the specific acceleration of a body, that is the difference between the acceleration and gravitational acceleration, as expressed in the body frame. A good tutorial may be found in [6.21].

The accelerometer measurements are assumed to be corrupted by zero-mean white noise $\boldsymbol{\eta}_{\text{acc}}$ so that the accelerometer measurement can be derived from (6.1) as

$$\boldsymbol{z}_{\text{acc}} = \mathbf{R}^{-1}\left(\ddot{\boldsymbol{x}} - \boldsymbol{g}\right) + \boldsymbol{\eta}_{\text{acc}} = \frac{1}{m}\left(\boldsymbol{e}_3 f + \boldsymbol{f}_a\right) + \boldsymbol{\eta}_{\text{acc}}. \tag{6.9}$$

Thus, given knowledge of the propellers' angular velocities and through (6.4) the static thrust force f, the accelerometer yields information about the aerodynamic force $\boldsymbol{f}_a$. This will be exploited in the design of the estimator, as it renders all three components of the vehicle's airspeed observable.

3.3 UWB range measurements

The UWB radio mounted on the quadrocopter uses a TOA-based algorithm, presented in Section 5, to calculate the distance between the quadrocopter at position $\boldsymbol{x}$ and an anchor i at position $\boldsymbol{p}_{\text{uwb,i}}$. This measurement is assumed to be a perfect measurement of distance, corrupted by zero mean white noise η_{uwb}, such that the measurement is given as

$$z_{\text{uwb,i}} = \left\|\boldsymbol{p}_{\text{uwb,i}} - \boldsymbol{x}\right\| + \eta_{\text{uwb}} \tag{6.10}$$

where $\|\cdot\|$ represents the Euclidean norm.

4. State estimator

An extended Kalman filter is used to estimate the state of the quadrocopter in flight. This section will present the necessary equations for the Kalman filter, derived from the system's dynamic model of Section 2 and the sensor models of Section 3.

To estimate the state of the quadrocopter the estimator will use sensor measurements from the accelerometer, angular rate gyroscope, and the UWB range sensors, and combine these with the motor commands. Throughout this section, a caret will be used to indicate an estimate, for example $\hat{\boldsymbol{\omega}}$ is the estimate of the body's angular velocity.

The goal of the estimator is to estimate the twelve-dimensional state of the rigid body, consisting of the quadrocopter's position, velocity, orientation, and angular velocity. The estimation is split into two parts: estimating the vehicle's angular velocity, and estimating the remaining states. As the computational complexity of a Kalman filter scales approximately as n^3, where n is the number of states [6.26], this reduces the computational cost to estimate the twelve states by more than 50%.

For the sake of brevity, the standard Kalman filter equations available in a standard reference (such as [6.26]) will not be repeated in this paper. Instead the focus will be on the system dynamics equations, and on the measurement equations.

The angular rate gyroscope measurement is used directly as the estimate of the quadrocopter's angular velocity, so that

$$\hat{\boldsymbol{\omega}} = \boldsymbol{z}_{\text{gyro}}. \tag{6.11}$$

This bases on the assumption that the measurement noise of the rate gyroscopes is negligible. The covariance of this estimate is equal to the covariance of $\boldsymbol{\eta}_{\text{gyro}}$.

An extended Kalman filter is used to estimate the quadrocopter's position, velocity, and orientation. The estimator has a nine-dimensional stochastic state ξ:

$$\xi = (\boldsymbol{x}, \boldsymbol{\rho}, \boldsymbol{\delta})\,. \tag{6.12}$$

where $\boldsymbol{x}$ is the quadrocopter's position expressed in the inertial coordinate system, and $\boldsymbol{\rho} = \mathbf{R}^{-1}\dot{\boldsymbol{x}}$ is the velocity of the quadrocopter expressed in the body frame. The three dimensional vector $\boldsymbol{\delta}$ is an attitude error representation, used to encode the uncertainty about the vehicle's orientation.

In addition to the stochastic state, the estimator also contains a reference orientation $\mathbf{R}_{\text{ref}}$. The reason for this parametrisation is as follows: although the orientation of one frame with respect to another contains only three degrees of freedom, there exists no global three-dimensional representation without singular points [6.27]. Thus, when using a three dimensional representation alone (such as Euler angles), the estimator will have to deal with singularities. However, using parametrisations with more than three elements (such as the four dimensional Euler symmetric parameters, or the nine dimensional rota-

tion matrix) requires applying constraints, which would in turn imply rank deficiency in the associated covariance matrix [6.28]. Higher dimensional representations also imply a higher computational cost.

For this reason, the estimated attitude is encoded using both the reference orientation and the three attitude error components. The error components are assumed to be infinitesimal, and then the estimator's attitude estimate is given as below [6.16], where all orientations are expressed with rotation matrices:

$$\hat{\mathbf{R}} = \hat{\mathbf{R}}_{\text{ref}} \left(\mathbf{I} + [\![\hat{\boldsymbol{\delta}} \times]\!] \right) \tag{6.13}$$

with $\mathbf{I} \in \mathbb{R}^{3\times3}$ the identity matrix. This formulation is sometimes referred to as a multiplicative Kalman filter, and the key idea is to maintain the orientation estimate in the reference attitude, use the three attitude error components to encode the covariance associated with the orientation. After each Kalman filter step, the reference rotation is updated with the attitude errors, and $\hat{\boldsymbol{\delta}}$ is reset to zero. The covariance is left unchanged during this reset step. More information on this approach can be found in e.g. [6.29].

4.1 Prediction equations

During the Kalman filter prediction step, the estimated states evolve as follows, from Section 2:

$$\dot{\hat{\boldsymbol{x}}} = \hat{\mathbf{R}}_{\text{ref}} \left(\mathbf{I} + [\![\hat{\boldsymbol{\delta}} \times]\!] \right) \hat{\boldsymbol{\rho}} \tag{6.14}$$

$$\dot{\hat{\boldsymbol{\rho}}} = \frac{1}{m} f \boldsymbol{e}_3 + \left(\frac{1}{m} \mathbf{K}_{\text{aero}} \dot{\theta}_\Sigma - [\![\hat{\boldsymbol{\omega}} \times]\!] \right) \hat{\boldsymbol{\rho}} - \|\boldsymbol{g}\| \left(\mathbf{I} - [\![\hat{\boldsymbol{\delta}} \times]\!] \right) \hat{\mathbf{R}}_{\text{ref}}^{-1} \boldsymbol{e}_3 \tag{6.15}$$

$$\dot{\hat{\boldsymbol{\delta}}} = \hat{\boldsymbol{\omega}} \tag{6.16}$$

$$\dot{\hat{\mathbf{R}}}_{\text{ref}} = 0 \tag{6.17}$$

The angular rate gyroscope is used as an input to this system, entering through (6.16). The zero-mean noise on the rate gyroscope is then trivially encoded as process noise using the standard extended Kalman filter formulation [6.26]. Additionally, a zero-mean acceleration is assumed to act on the system, so that an additional three dimensional process noise is taken to act upon $\boldsymbol{\rho}$.

By stacking (6.14)-(6.16), taking the derivative with respect to ξ, and evaluating at the filter's current estimate $\hat{\xi}$ and $\hat{\mathbf{R}}_{\text{ref}}$, the Jacobian necessary for performing the Kalman filter covariance prediction can be computed.

4.2 Accelerometer measurement update equation

The accelerometer allows to infer the vehicle's airspeed, through the aerodynamic forces. It is assumed that the propellers' angular rates $\dot{\theta}_i$ are known, so that the static thrust f can be computed. This can then be subtracted from the accelerometer measurement, so

that the result $\tilde{\boldsymbol{z}}_{\text{acc}}$ is a measurement of $\boldsymbol{f}_a$, scaled by the vehicle mass and corrupted by noise, by (6.9) and (6.5):

$$\begin{aligned}\tilde{\boldsymbol{z}}_{\text{acc}} &= \boldsymbol{z}_{\text{acc}} - \frac{1}{m}\boldsymbol{e}_3\kappa\sum_{i=1}^{4}\dot{\theta}_i = \frac{1}{m}\boldsymbol{f}_a + \boldsymbol{\eta}_{\text{acc}} \\ &= \frac{1}{m}\mathbf{K}_{\text{aero}}\,\dot{\theta}_\Sigma\,\boldsymbol{\rho} + \boldsymbol{\eta}_{\text{acc}}\end{aligned} \tag{6.18}$$

which is linear in the quadrocopter's airspeed, and may thus be easily encoded in the Kalman filter.

4.3 UWB range measurement update equation

A range measurement to an anchor can be modelled by (6.10). This may again easily be linearised about the estimator's current state, and incorporated in the filter.

1) Ranging outlier rejection Outliers from the UWB ranging radios may be detected by computing likelihood of a given ranging measurement. Because the measurements are scalar, the resulting innovation covariance [6.26] will also be scalar, and may thus be inverted at low computational cost. Squaring the difference between the actual range measurement and the measurement expected given the current state estimate, and multiplying by the inverse innovation yields a squared normalised distance called the Mahalanobis distance [6.26]. The larger this normalised distance the less likely a particular measurement is to result from the statistical properties of the expected error. A measurement with a distance larger than some given threshold may thus be rejected as an outlier.

5. Range measurement using time-of-arrival measurements

As discussed in Section 3.3, a UWB radio mounted to the quadrocopter is used to measure distances to anchors placed within the environment. Many different methods exist to measure distance using UWB radio [6.13], [6.30]–[6.32]; for the purposes of this paper, a TOA-based method known as two way ranging is used[6.31].

5.1 The two way ranging algorithm

The high temporal resolution of UWB pulses enables accurate range measurement; however, it also poses challenges, such as compensating for clock frequency differences between UWB modules [6.13], [6.30]. Compensating for these differences is achieved using a variation of the two way ranging algorithm, which employs a repeated reply to allow measurement of anchor delay relative to the quadrocopter's clock [6.31].

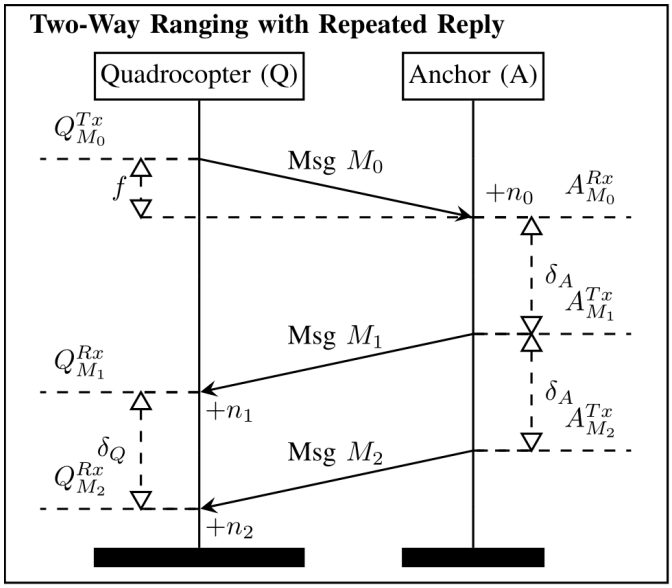

Figure 6.2. A two-way ranging with repeated reply algorithm is used to measure the time of flight between a quadrocopter and an anchor. By subtracting the locally-measured processing time ($Q^{Rx}_{M_2} - Q^{Rx}_{M_1}$) from the round-trip time ($Q^{Rx}_{M_1} - Q^{Tx}_{M_0}$), the time of flight ($f$) can be estimated in the quadrocopter's local clock.

1) Measurement of clock frequency difference As shown in Fig. 6.2, after sending its first reply, the anchor waits for a predetermined amount of time $\delta_A = \delta$, where δ is a predefined constant, here chosen as 1 ms. After this time delay, the anchor's reply message is repeated, allowing the anchor's delay to be measured by the quadrocopter as

$$z_{\delta_Q} = \delta_Q + n_2 - n_1 = Q^{Rx}_{M_2} - Q^{Rx}_{M_1}, \tag{6.19}$$

where n_i are samples of a noise distribution and affect the reception timestamp. Fig. 6.3 shows the difference between the anchor's delay and the expected delay, as measured by the quadrocopter ($\delta_Q - \delta$). This measurement is time-varying due to clock frequency drift and is affected by noise.

It is assumed that the noise term $n_2 - n_1$ is zero mean, and thus δ_Q tracks the mean of the time-varying measurements. By applying a low-pass filter over successive measurements, δ_Q can thus be estimated as $\hat{\delta}_Q$. Analysing the distribution of the noise term $n_2 - n_1$ around this estimate showed that this term is uncorrelated between distance measurements and distributed with standard deviation 0.17 ns. Fig. 6.3 further shows the existence of outliers, which motivates the inclusion of outlier detection as previously discussed in Section 4.3.

2) Time of flight measurement Using the estimate $\hat{\delta}_Q$, the quadrocopter is able to mea-

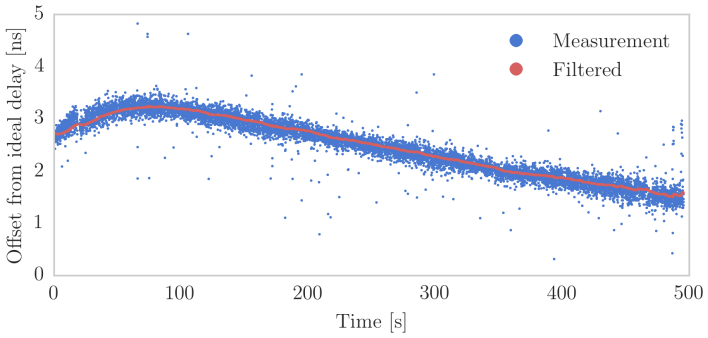

Figure 6.3. A plot showing the difference between the anchor's delay and the expected delay, as measured by the quadrocopter. In the ideal case, this would equal zero; the discrepancy is due to a difference in clock frequencies between the quadrocopter and anchor. This function is time varying, due to clock frequency drift, and is corrupted by an uncorrelated noise with standard deviation 0.17 ns. Note that 1 ns corresponds to a distance measurement error of 30 cm.

sure the time of flight f as

$$z_{\text{TOF}} = f + \frac{n_1 + n_0}{2} = \frac{Q_{M_1}^{Rx} - Q_{M_0}^{Tx} - \hat{\delta}_Q}{2}. \tag{6.20}$$

Multiplying the time of flight by the speed of light c and comparing with (6.10) yields

$$\begin{aligned} z_{\text{uwb}} &= cf + \frac{c}{2}(n_1 + n_0) \\ &= \|\boldsymbol{p}_{\text{uwb,i}} - \boldsymbol{x}\| + \eta_{\text{uwb}} \end{aligned} \tag{6.21}$$

Assuming the noise term n_1+n_0 has the same statistical properties as n_2-n_1, we conclude that η_{uwb} is zero mean with standard deviation 0.025 m. This result was experimentally verified by recording range measurements at a constant distance and calculating their statistical properties. Furthermore, this result reflects a similar analysis performed in [6.32].

6. Experimental validation

The estimation approach is validated in experiment in the Flying Machine Arena [6.4]. Ascending Technologies Hummingbird quadrocopters [6.33] are used, modified to use the Pixhawk PX4 flight management unit [6.10]. The estimator is implemented in C++ and runs on the microcontroller on the flight management unit.

The flight management unit features a three axis accelerometer and gyroscope, each sampled at 1000Hz for the estimator.

Six UWB radios were used for the experiments: one mounted on the quadrocopter,

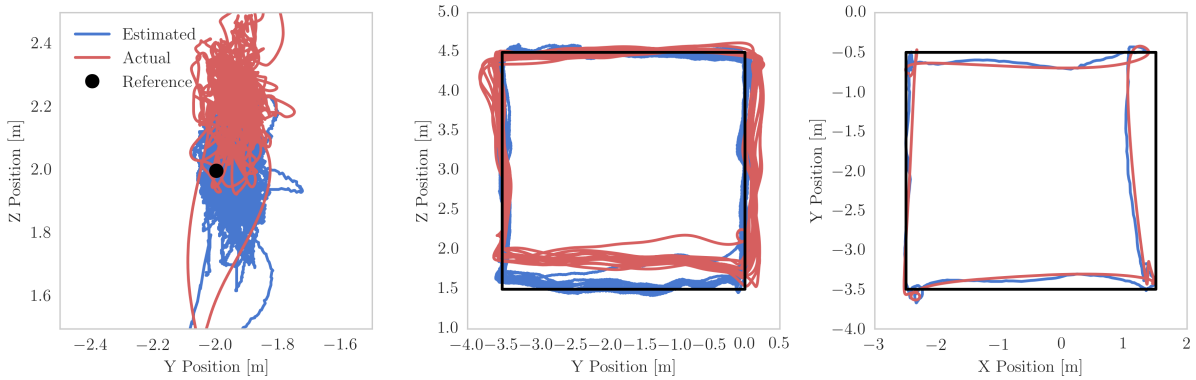

Figure 6.4. Three experiments evaluating the quadrocopter performance under feedback control using the estimator, from left to right: the quadrocopter hovers at a position for 13 minutes; the quadrocopter flies a slow vertical manoeuvre; and the quadrocopter flies a fast horizontal manoeuvre. Gravity points along negative Z. For the hover experiment, the results were comparable in both horizontal directions, therefore only the Y-Z plane is shown – a history of the estimation errors is given in Fig. 6.5. The standard deviation of the estimation error during the hover experiment was 36 mm horizontally, and 53 mm vertically, with the corresponding mean values of 50 mm and 244 mm, respectively. The setpoint was moved at a constant $1\,\mathrm{m\,s^{-1}}$ during the slow manoeuvre, and the result clearly shows systematic biases present in the system, especially along the lower edge of the trajectory. The maximum speed along the fast trajectory was $3.4\,\mathrm{m\,s^{-1}}$. Eleven sequential rounds of the fast trajectory were flown, however only one is shown. The attached video shows each of the experiments.

and five used as anchors. Of the anchors, three were placed in an approximately isosceles triangle on the floor, and two were placed at a height of approximately 1.7 m above the ground. At 200 Hz the quadrocopter requests a range to an anchor, starting at the first anchor and proceeding sequentially.

A controller also runs on the microcontroller, which computes desired motor angular velocities as a function of the estimator's state, and a desired state. A position setpoint is periodically transmitted to the vehicle from a base station over a dedicated wireless channel.

The Flying Machine Arena is additionally equipped with an overhead motion capture system, which measures the position and orientation of the quadrocopter at 200Hz, with precision on the order of millimetres and degrees, respectively. The output from this motion capture system is used as ground truth for these experiments. Note that the quadrocopter's closed loop control is based solely on the output of the presented estimator, and the motion capture system is used exclusively for performance evaluation.

6.1 Estimator parameters

The quadrocopter has a mass of 0.56 kg. The propeller force coefficient is given by $\kappa = 6.41\,\mu\mathrm{N\,s^2\,rad^{-2}}$.

The aerodynamic force coefficients $\kappa_{\parallel}$ and $\kappa_{\perp}$ were estimated by analysing data from

quadrocopter flights in the Flying Machine Arena. The coefficients are estimated as:

$$\kappa_{\perp} = -0.000\,11\,\mathrm{N\,s^2\,rad^{-1}\,m} \tag{6.22}$$

$$\kappa_{\parallel} = -0.000\,23\,\mathrm{N\,s^2\,rad^{-1}\,m}. \tag{6.23}$$

The noise covariance for the angular rate gyroscope and the accelerometer were set as follows:

$$\mathrm{Var}\left(\boldsymbol{\eta}_{\mathrm{gyro}}\right) = \mathrm{diag}\left(0.01, 0.01, 0.25\right)\mathrm{rad^2\,s^{-2}} \tag{6.24}$$

$$\mathrm{Var}\left(\boldsymbol{\eta}_{\mathrm{acc}}\right) = \mathrm{diag}\left(9, 9, 81\right)\mathrm{m^2 s^{-4}} \tag{6.25}$$

These covariances are partly based on experimental data, which showed a significantly higher accelerometer noise in the direction of the quadrocopter's thrust, and partly on experimental tuning. The observed higher noise in the direction of the thrust could possibly be explained by the quadocopter's asymmetric mechanical structure, and the transmission of vibrations.

The covariance of the UWB range measurements was set to $\mathrm{Var}\left(\eta_{\mathrm{uwb}}\right) = 0.0625\,\mathrm{m^2}$. This is an order of magnitude larger than suggested in Section 5, however due to a possible unmodelled measurement bias (discussed further in this section), was necessary for smooth flight. The Mahalanobis distance for rejecting a UWB range measurement was set to 3.

The acceleration process noise acting to increase the estimator's covariance in the prediction state of the Kalman filter was set to $9\,\mathbf{I}\,\mathrm{m^2/s^{-4}}$.

6.2 Experiments

Three different experiments were performed: hovering in one spot for an extended period of time, flying a slow vertical manoeuvre, and flying a fast horizontal manoeuvre. The experiments are shown in Fig. 6.4, and each experiment is shown in the attached video. Errors are quantified by their mean (ϵ_μ) and standard deviation (ϵ_σ) – these are related to the root-mean-squared (RMS) error (ϵ_{RMS}) as $\epsilon_{\mathrm{RMS}}^2 = \epsilon_\mu^2 + \epsilon_\sigma^2$ [6.26].

1) Hovering The quadrocopter was commanded to hold a position in space for a period of 13 minutes. The quadrocopter's yaw angle (that is, the rotation about the thrust axis) was commanded to be constant throughout this period.

The mean position estimation error was approximately 50 mm in the horizontal direction, and 244 mm in the vertical direction. The standard deviation of the error was 36 mm horizontally and 53 mm vertically.

The RMS closed loop position tracking error was 302 mm, and the RMS closed loop yaw tracking error was 5.3°.

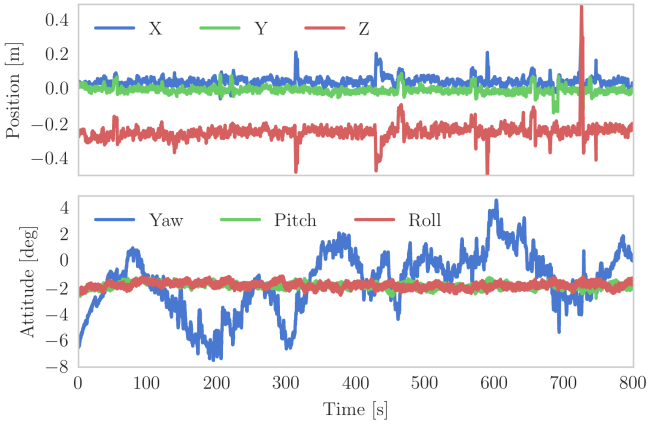

Figure 6.5. The estimation error over approximately 13 minutes of hovering. The actual and estimated position trajectories are shown in Fig. 6.4. The attitude error is reported as differences in the true and estimated Z-Y-X yaw-pitch-roll angles [6.16].

2) Slow vertical manoeuvre A second experiment was performed where the quadrocopter flies along a vertically oriented rectangle of size $3.5 \times 3\,\mathrm{m}$. The position setpoint was moved along slowly at $1\,\mathrm{m\,s^{-1}}$ to minimize the influence dynamic effects, and interactions of the controller. The quadrocopter flew twelve times along the rectangle. A discussion of the system's performance is given below.

3) Fast horizontal manoeuvre The performance of the estimator during more dynamic trajectories was investigated by having the quadrocopter fly along a $4 \times 3\,\mathrm{m}$ horizontal rectangle. Trajectories to the corners were generated using the method of [6.34], with the quadrocopter commanded to start and end at rest, at the corners. The duration along the long edge of the rectangle was 2.2 s, and along the short edge 1.9 s, so that the maximum commanded velocity along the trajectory is $3.4\,\mathrm{m\,s^{-1}}$. Over 11 rounds around the rectangle, the position estimation error had a mean of 41 mm, and standard deviation of 123 mm. One such round around the rectangle is shown in Fig. 6.4, and the attached video visualises an ensemble of such rounds.

4) Discussion and interpretation The experimental results show that the quadrocopter's full state is observable to the state estimator. Specifically it appears that the quadrocopter's corrective motions when holding a constant position set point are sufficient for observing the quadrocopter's orientation about its thrust axis, also over prolonged periods of time.

Both the hover experiment, and the slow vertical experiment, showed significant non-zero-mean estimation errors in position. For the vertical rectangle, as can be clearly seen in Fig. 6.4, there is a significant systematic vertical estimation error along the lower edge of the rectangle. The relatively large mean position estimation errors (on the order of 250 mm) stand in contrast to the much lower standard deviation on the position estimate (64 mm when hovering, and 123 mm along the fast horizontal rectangle).

These systematic estimation errors are most likely explained by systematic errors in the UWB ranging system. Systematic ranging errors could be due to the effects of occlusion and multi-path on the radio signal [6.35], however the experimental setup had no occlusions. An alternative explanation is given in [6.14], [6.36], where the group-delay of an antenna, corresponding to a time-delay of the signal, is a function of antenna orientation. This appears to match the observation that the magnitude and direction of the estimation errors vary as a function of the position in space.

7. Conclusion

The strategy presented in this paper utilises accelerometers, gyroscopes, and ultra-wideband radios to estimate the dynamic state of a quadrocopter, under the assumption of no wind. The estimator is shown to perform sufficiently well for the quadrocopter to maintain a position for an extended period of time, and to fly dynamic manoeuvres. Furthermore, the computational complexity is low enough that the estimator may be run on a typical microcontroller.

Inertial measurement sensors are already widely used on flying vehicles, and ultra-wideband radios may be added at low cost and at little additional mass. This system thus appears to be a low-cost, easily-implementable method to improve the autonomy of quadrocopter systems.

Acknowledgement

The Flying Machine Arena is the result of contributions of many people, a full list of which can be found at `http://flyingmachinearena.org`.

This research was supported by the Swiss National Science Foundation (SNSF) and the NCCR Digital Fabrication (Agreement # 51NF40-141853), also funded by the SNSF.

References

[6.1] R. D'Andrea, "Can drones deliver?", *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 3, pp. 647–648, 2014.

[6.2] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment", *IEEE Control Systems Magazine*, vol. 28, no. 2, pp. 51–64, Apr. 2008.

[6.3] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed", *IEEE Robotics Automation Magazine*, vol. 17, no. 3, pp. 56–65, Sep. 2010.

[6.4] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena", *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[6.5] S. Klose, J. Wang, M. Achtelik, G. Panin, F. Holzapfel, and A. Knoll, "Markerless, vision-assisted flight control of a quadrocopter", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2010, pp. 5712–5717.

[6.6] C. Martinez, P. Campoy, I. Mondragon, and M. A. Olivares-Mendez, "Trinocular ground system to control UAVs", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2009, pp. 3361–3367.

[6.7] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: A compendium", *Journal of Field Robotics*, vol. 30, no. 5, pp. 803–831, 2013.

[6.8] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2012, pp. 2815–2821.

[6.9] I. Sa and P. Corke, "100Hz onboard vision for quadrotor state estimation", in *Australasian Conference on Robotics & Automation*, 2012.

[6.10] L. Meier, P. Tanskanen, L. Heng, G. Lee, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision", English, *Autonomous Robots*, vol. 33, no. 1-2, pp. 21–39, 2012.

[6.11] K. Schauwecker and A. Zell, "On-board dual-stereo-vision for autonomous quadrotor navigation", in *International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2013, pp. 333–342.

[6.12] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Precision flight control for a multi-vehicle quadrotor helicopter testbed", *Control engineering practice*, vol. 19, pp. 1023–1036, June 2011.

[6.13] S. Gezici, Z. Tian, G. Giannakis, H. Kobayashi, A. Molisch, H. Poor, and Z. Sahinoglu, "Localization via ultra-wideband radios: A look at positioning aspects for future sensor networks", *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 70–84, 2005.

[6.14] M. Mahfouz, C. Zhang, B. Merkl, M. Kuhn, and A. Fathy, "Investigation of high-accuracy indoor 3-D positioning using UWB technology", *IEEE Transactions on Microwave Theory and Techniques*, vol. 56, no. 6, pp. 1316–1330, 2008.

[6.15] A. Prorok, A. Arfire, A. Bahr, J. R. Farserotu, and A. Martinoli, "Indoor navigation research with the Khepera III mobile robot: An experimental baseline with a case-study on ultra-wideband positioning", in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2010, pp. 1–9.

[6.16] P. H. Zipfel, *Modeling and Simulation of Aerospace Vehicle Dynamics Second Edition*. AIAA, 2007.

[6.17] R. Mahony, V. Kumar, and P. Corke, "Aerial vehicles: Modeling, estimation, and control of quadrotor", *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[6.18] P. Pounds, R. Mahony, P. Hynes, and J. Roberts, "Design of a four-rotor aerial robot", in *Australasian Conference on Robotics and Automation*, vol. 27, 2002, p. 29.

[6.19] P. Martin and E. Salaun, "The true role of accelerometer feedback in quadrotor control", in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2010, pp. 1623–1629.

[6.20] D. Abeywardena, S. Kodagoda, G. Dissanayake, and R. Munasinghe, "Improved state estimation in quadrotor MAVs: A novel drift-free velocity estimator", *IEEE Robotics & Automation Magazine*, pp. 32–39, 2013.

[6.21] R. Leishman, J. Macdonald, R. Beard, and T. McLain, "Quadrotors and accelerometers: State estimation with an improved dynamic model", *Control Systems, IEEE*, vol. 34, no. 1, pp. 28–41, 2014.

[6.22] B. W. McCormick, *Aerodynamics Aeronautics and Flight Mechanics*. John Wiley & Sons, Inc, 1995.

[6.23] H. Huang, G. Hoffmann, S. Waslander, and C. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering", in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2009, pp. 3277–3282.

[6.24] M. Bangura, H. Lim, H. Kim, and R. Mahony, "Aerodynamic power control for multirotor aerial vehicles", in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 529–536.

[6.25] J. L. Crassidis, F. L. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods", *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[6.26] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: Theory algorithms and software*. John Wiley & Sons, 2004.

[6.27] J. Stuelpnagel, "On the parametrization of the three-dimensional rotation group", *SIAM review*, vol. 6, no. 4, pp. 422–430, 1964.

[6.28] M. D. Shuster, "Constraint in attitude estimation Part I: Constrained estimation", *Journal of the Astronautical Sciences*, vol. 51, no. 1, pp. 51–74, 2003.

[6.29] F. L. Markley, "Attitude error representations for Kalman filtering", *Journal of guidance, control, and dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[6.30] H. Soganci, S. Gezici, and H. Poor, "Accurate positioning in ultra-wideband systems", *IEEE Wireless Communications*, vol. 18, no. 2, pp. 19–27, 2011.

[6.31] R. Dalce, "Comparison of indoor localization systems based on wireless communications", *Wireless Engineering and Technology*, vol. 02, no. 04, pp. 240–256, 2011. [Online]. Available: `http://www.scirp.org/journal/PaperDownload.aspx?DOI=10.4236/wet.2011.24033`.

[6.32] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative localization in wireless networks", *Proceedings of the IEEE*, vol. 97, no. 2, pp. 427–450, 2009. [Online]. Available: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4802193`.

[6.33] D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz", in *IEEE International Conference on Robotics and Automation (ICRA)*, Apr. 2007, pp. 361–366.

[6.34] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3480–3486.

[6.35] A. Shahi, A. Aryan, J. S. West, C. T. Haas, and R. C. G. Haas, "Deterioration of UWB positioning during construction", *Automation in Construction*, vol. 24, pp. 72–80, 2012. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0926580512000246`.

[6.36] W. Soergel and W. Wiesbeck, "Influence of the antennas on the ultra-wideband transmission", *EURASIP J. Appl. Signal Process.*, vol. 2005, pp. 296–305, 2005.

Curriculum Vitae

Mark Wilfried Mueller
born 22 August 1986

2011 – 2015	*ETH Zurich, Switzerland* Ph.D. Candidate at the Institute for Dynamic Systems and Control (adviser: Prof. Raffaello D'Andrea), Department of Mechanical and Process Engineering.
2011	*I3S, Sophia Antipolis, France* Academic internship.
2009 – 2011	*ETH Zurich, Switzerland* Master studies in Mechanical Engineering; graduated with M.Sc. ETH Mechanical Engineering.
2009	*Denel Dynamics, Centurion, South Africa* Junior software practitioner.
2007	*Rensselaer Polytechnic Institute, Troy NY, USA* Exchange semester.
2005 – 2008	*University of Pretoria, South Africa* Bachelor studies in Mechanical Engineering; graduated with B.Eng. Mechanical Engineering.

List of publications

Journal publications

- M. W. Mueller and R. D'Andrea, "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles", *International Journal of Robotics Research*, 2015
- M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation", *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015
- F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines", *Control Systems, IEEE*, vol. 34, no. 4, pp. 46–64, 2014
- S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena", *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014
- M. W. Mueller, L. Liebenberg, E. H. Mathews, and P. W. Young, "Quick estimates for analysis and prediction of the flight mechanics of unmanned aerial vehicles", *International Journal of Mechanical Engineering Education*, vol. 40, no. 2, pp. 121–145, 2012

Conference proceedings (peer-reviewed)

- W. Zhang, M. W. Mueller, and R. D'Andrea, "A controllable flying vehicle with a single moving part", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016
- M. W. Mueller, M. Hamer, and R. D'Andrea, "Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015
- M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 45–52
- M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3480–3486
- M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception", in *European Control Conference (ECC)*, 2013, pp. 1383–1389

- M. W. Mueller and R. D'Andrea, "Critical subsystem failure mitigation in an indoor uav testbed", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 780–785
- R. Ritz, M. W. Mueller, M. Hehn, and R. D'Andrea, "Cooperative quadrocopter ball throwing and catching", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 4972–4978
- M. W. Mueller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 5113–5120

Patents

- M. W. Mueller, S. Lupashin, R. D'Andrea, and M. Waibel, "Controlled flight of a multicopter experiencing a failure affecting an effector", *Patent pending, WO 2014/198641 A1*, 2014
- M. W. Mueller, S. Lupashin, R. D'Andrea, and M. Waibel, "Volitant vehicle rotating about an axis and method for controlling the same", *Patent pending, WO 2014/198642 A1*, 2014