Modelling and Control of the Crazyflie Quadrotor for Aggressive and Autonomous Flight by Optical Flow Driven State Estimation

Marcus Greiff

LUND
UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6026
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2017

Abstract

The master thesis seeks to develop a control system for the Crazyflie 2.0 unmanned aerial vehicle to enable aggressive and autonomous flight. For this purpose, different rigid-body models are considered, differing primarily in their parametrisation of rotation. The property of differential flatness is explored and several means of parametrising trajectories in flat output space are implemented. A new method of rotor control with parameter estimation is developed and geometric controllers are implemented for rigid-body control. Finally, state estimation is accomplished through a scalar-update extended Kalman filter, where information from the internal measurement unit is fused with positional information from camera systems, ultra-wide band systems, optical flow measurements and laser ranging measurements. Capable of sustaining flight with any combination of the previously mentioned sensors, the real-time implementation is showcased using polynomial motion-planning to avoid known obstacles.

Acknowledgements

I would like to express my sincere thanks to the Professors Anders Robertsson and Bo Bernhardsson as well as Ph.D. student Fredrik Bagge Karlsson for valuable advice and helpful theoretical discussions throughout the course of the thesis. I also wish to thank Professor Karl-Johan Åström for helpful discussions on the complementary filtering and enthusiastic encouragement. In addition, would like to thank the members of Bitcraze AB individually. Firstly, Arnaud Taffanel, for his patience and help with the *EKF* implementation and input on the firmware architecture. Secondly, Marcus Eliasson, for help with the design and development of the expansion boards relating to optical flow and laser ranging. Thirdly, Tobias Antonsson, for helpful discussions on the rotor-loops and debugging of the Crazyflie radio. Fourthly, Kristoffer Richardsson, for discussions on the firmware implementation and ideas relating to organisation of labour. Finally, Björn Mauritz, for support and encouragement throughout the project. I would also like to thank Michael Hamer, for his prior work on the extended Kalman filter, helpful discussions on the control system throughout the thesis and help with tuning the geometric controller. Finally, I would like to thank Daniel Nilsson, for his contribution to the ROS implementation in the Kinect vision project.

Contents

1. Introduction **1**
- 1.1 Outline of thesis . . . 2
- 1.2 Goals of thesis . . . 3

2. Modelling **4**
- 2.1 Tait-Bryan rigid-body dynamics . . . 4
- 2.2 Quaternion rigid-body dynamics . . . 11
- 2.3 Rotor dynamics and coupling . . . 14
- 2.4 Implementation considerations . . . 17
- 2.5 Open loop response . . . 21
- 2.6 Summary . . . 23

3. Motion planning **24**
- 3.1 Differential flatness . . . 26
- 3.2 Generation of flat output trajectories . . . 32
- 3.3 Parametrization of flat outputs . . . 38
- 3.4 Summary . . . 42

4. Rotor control **44**
- 4.1 Open loop control . . . 44
- 4.2 Closed loop rotor control . . . 45
- 4.3 Rotor adaptation and estimation . . . 50
- 4.4 Summary . . . 53

5. Rigid-body control **54**
- 5.1 Saturations and controllability . . . 55
- 5.2 Tait-Bryan parametrised control . . . 57
- 5.3 Geometric control . . . 63
- 5.4 Summary . . . 68

6. Inner state estimation **69**
- 6.1 Model independent estimation . . . 70
- 6.2 Model based state estimation . . . 75
- 6.3 MOCAP positioning . . . 78

6.4 UWB positioning 84
6.5 Optical flow and laser positioning 92

7. Conclusions and summary **102**

Bibliography **104**

A. Modelling appendix **111**
A.1 Identification of mappings 111
A.2 Continuous time DC motor parameters 113
A.3 Coriolis matrix definition 115
A.4 Gimbal lock avoidance with Tait-Bryan angles 116
A.5 Quaternion rotations and relation to Tait-Bryan angles . . . 119
A.6 Quaternion rate of change 121
A.7 Quaternion state-space representation 122
A.8 Linearised systems 123
A.9 Closed form system integration with constant terms 124
A.10 Cross product identities 125
A.11 Time derivative of the rotation matrix 126

B. Controller appendix **127**
B.1 Set-point weighed PID 127
B.2 MRAC with MIT synthesis 128
B.3 Linear quadratic reaulators 130

C. State estimation appendix **132**
C.1 Cramer-Rao lower bound in TOA 132
C.2 Robust protocols considering clock drift 133
C.3 The Extended kalman Filter 136
C.4 Multi-camera LS regression 138

Index of Acronyms

Acronym	Description
UAV	Unmanned Aerial Vehicle
EKF	Extended Kalman Filter, a state estimator using system dynamics
PID	Proportional-Integral-Derivative regulator
LQR	Linear-Quadratic Regulator
UWB	Ultra-Wideband, referring to radio communication with a large bandwith
TOA	Time Of Arrival, a term used to describe a class of *UWB*-positioning algorithms
MRAC	Model Reference Adaptive Control, here for *SISO* systems with *MIT*-rule synthesis
GA	Genetic Algorithm, used in the motion-planning of the rotor-craft
CIR	Channel Impulse Response, used to determine time-stamps in the *TOA* methods
RLS	Recursive Least-Squares, here for on-linear parameter identification
LS	Least-Squares, referring to any problem formulated and solved in a least-squares sense
TSP	The Travelling Salesman Problem, an integer-programming problem
IMU	Inertial Measurement Unit, contains a 3-axis gyroscope, a 3-axis accelerometer and a 3-axis magnetometer

Acronym	Description
PD	Proportional-Derivative control
PWM	Pulse-Width Modulation, used in the rotor control
MOCAP	Motion capture, here referring to camera systems
TWR	Two-Way Ranging, a protocol for communication in the *UWB*-system
TDOA	Time Difference Of Arrival, a term used to describe a class of *UWB*-positioning algorithms
SO(3)	Isometric rotations in $\mathbb{R}^3$ with fixed origin
CRLB	Cramer-Rao Lower Bound, a lower bound for any unbiased estimator
DW1000	The radio chip implemented in the *LPS*
SISO	Single-Input Single-Output, a class of dynamical systems
SFD	Start of Frame Delimiter, used to time-stamp packets in the *UWB*-network
SDS-TWR	Symmetric-Double-Sided Two-Way Ranging, a communication protocol
QR	Referring to the *QR*-decomposition of matrices
MSE	Mean Squared Error
LPS	Loco Positioning System, an *UWB*-system being developed by Bitcraze
DC	Direct Current, with respect to the brushed motors implemented in the *UAV*
SE(3)	Euclidean group of isometric rotations in $\mathbb{R}^3$ with fixed origin and fixed-sign determinant
ROS	The Robot Operating System, a meta-operating system for robotics
LP	Low-Pass, referring to a filter which attenuates high frequency spectral content
AHRS	Attitude Heading and Reference Systems, a category of complementary filters
MIT	Massachusetts Institute of Technology
ZOH	Zero-Order Hold, an approximation used when discretising continuous systems

Acronym	Description
TOF	Time Of Flight, referring to communication in the *UWB*-network or laser ranging
PCB	Printed Circuit Board, a prototype board for electronics
FIM	The Fischer Information Matrix, evaluated in order to compute the *CRLB*
TV	Time-Varying, either with respect to a dynamical system or to a controller operating on a time-variant system
TI	Time-Invariant, with respect to a dynamical system or to a controller operating on a time-variant system
TDMA	Time Division Multiple Access, a method of distributing resources in a real-time system
SPR	Strictly Positive Real, a condition for stability in Lyapunov-rule *MRAC*
SPI	Serial Peripheral Interface, a interface used for short distance communication
SLAM	Simultaneous Localisation And Mapping
SISO	Single-Input Multiple-Output, a class of dynamical systems
DWM1000	A module from Decawave implementing a resonance antenna and the *DW1000 IC*
WLS	Weighted Least-Squares, a variation of the standard *LS* formulation
RSS	Received Signal Strength, a method used in *UWB*-positioning
QP	Quadratic Programming, a class of optimisation problems
OWR	One-Way Ranging, a communication protocol used in *UWB*-positioning
MIMO	Multiple-Input Multiple-Output, a class of dynamical system
LE	Leading-Edge algorithm, used to detect peaks in time series
JPL	Jet Propulsion Laboratory
DF	Differential Flatness
AW	Anti-Windup, implemented in integrating controllers

1

Introduction

Autonomous aircraft control is an interesting and widely researched topic, as it poses great theoretical challenges and can be considered for a vast amount of applications. For instance, small unmanned aerial vehicles (UAV) have previously been used to build temporary rope bridges between gorges, with the intention of eventually aiding aid fire brigades and rescue workers [Augugliaro et al., 2015]. Other, more distant prospects include UAV's that navigate public spaces safely to perform basic tasks such as inventorying and refilling shelves in supermarkets. With the rapid development of machine learning and technological innovations in embedded systems, applications concerning human interaction that were unimaginable ten years ago are almost possible today and will be readily implementable in the near future.

With development of such applications limited mainly by ceilings and human imagination, a good method of progression is through open-source collaboration. However, some significant hurdles need to be overcome before moving research forward and contributing to any UAV project. Certain knowledge of control theory, electrical engineering and computer science is needed, but a much greater constraint is the cost related to motion capture systems ($MOCAP$). In well funded research labs, millimeter precision in positional estimation can be attained with high performance camera systems, but this is certainly not an option for the individual researcher as the price typically amounts to tens of thousands of dollars. Instead, more affordable ultra-wideband (UWB) systems may be used for positional estimation with precision on the decimetre scale at a tenth of the cost. Alternatively, cheaper solutions can be envisioned with simple household cameras, offering good precision in a very small flyable space. Finally, we may consider positional estimation free from external systems altogether by using optical flow, where a camera and laser is mounted directly on the UAV enabling autonomous flight in a virtually unbounded space at a fraction of any other motion capture system.

This thesis seeks to provide basic building blocks for developers and researchers, taking all of the above methods of positioning into account such that the final product can be used in high performance labs and hobby implementations alike. Simultaneously, the work aspires to decrease the knowledge threshold for developing algorithms on the *UAV* platform by contributing well tested models to the open source community. The work is divided into five main sections, each touching on current research including original findings and conclusions based on either simulations or physical experiments. The thesis contains the theoretical developments required to understand the implementation, but only concerns the embedded inner control system used for autonomous flight of the Crazyflie *UAV*. The sections on *SLAM*, the *ROS* implementation and the outer control system are omitted entirely and will be published in a more extensive second report.

1.1 Outline of thesis

In **Chapter 2**, general *UAV* rigid-body dynamics are derived with a discussion on the choice of rotation parametrisation and their consequences for controllability. This section seeks to provide developers with well tested models to aid testing of future control systems. Three significant contributions to the pre-existing research body include (i) an approximation enabling the Tait-Bryan angle model to operate free of singularities, (ii) an analytically derived error-synamics of the quaternion quad-rotor system free of trigonometric terms and (iii) a method of sampling linear temporal invariant systems with constant terms, such as a gravitational force field.

In **Chapter 3**, four distinct methods of motion planning are detailed and and implemented to simplify development of future applications. This includes (i) a novel solver for the travelling salesman problem to find the shortest distance through a subset of points in space, (ii) a new projective algorithm for obstacle avoidance (iii) an implementation of a previously defined quadratic program [Richter et al., 2013], and finally (iv) a derivation of the differential flatness equations for the quaternion system for generating system-compliant state trajectories.

In **Chapter 4**, a potential improvement of the current control system is presented by introducing separate control loops with respect to each rotor. Different approaches are simulated and compared with reference to error metrics, finding good methods of parameter estimation and control with open invitation to future development.

In **Chapter 5**, we present four of many tested rigid-body controllers which were subsequently implemented in the *UAV* firmware to enable autonomous flight. This includes a discussion on necessary saturations required to operate the quad-rotor through aggressive looping manoeuvres in a real-

time context, as well as the presentation of a singularity free geometrical controller [Lee et al., 2010] which has previously been implemented a real-time context at Penn University [Mellinger and Kumar, 2011]. In addition, an extension is proposed to the standard Tait-Bryan *LQR*-type controllers [Landry, 2015] [Tedrake, 2009], were integrating states with conditional anti-windup are used to improve disturbance attenuation.

In **Chapter 6**, discussed the implemented system for state estimation, building on previous work by Bitcraze AB and research by Michael Hamer at the Swiss Federal Institute of Technology (*ETH*) [Mueller et al., 2016] [Mueller, 2016] [Mueller et al., 2015]. This section discusses the theory of enabling sensor fusion of inertial measurement unit *IMU*, *MOCAP*, *UWB*, optical flow and laser ranging measurements in a variety of filters and estimators. Our contributions to the implementation is primarily with (i) development of algorithms for inclusion of optical flow and laser ranging in the Extended Kalman filter, (ii) alternative static methods of estimation with least-squares, (iii) means of *MOCAP* using simple Kinect and web cameras, (iv) multi-path compensation in the *UWB*-measurements and (v) implementation of bias compensated *IMU* filters based on the research of Madgwick [Madgwick et al., 2011].

1.2 Goals of thesis

The goals of this thesis are many, but primarily concern two main points. The first is to decrease knowledge barrier in developing applications with the Crazyflie *UAV* and the second is to decrease the cost of positional state estimation by implementing various means of positioning in a real-time context. The primary goals are to

- Implement rigid-body and rotor models for the Crazyflie.
- Derive the differential flatness equations for the dynamics.
- Devise a robust rotor control system.
- Create means of loading and evaluating trajectories synchronously across multiple *UAV*s.
- Implement an embedded *SE(3)* controller in the *UAV* firmware for rigid-body control, as originally theorised in [Lee et al., 2010].
- Enamble embedded state estimation through a fusion of optical flow, laser ranging, *UWB* and *IMU* measurements in an *EKF*.

All above goals have been met and will here be described in theoretical terms, referring the interesting reader to [Greiff, 2017] for code and supplementing notes on the real-time implementation.

2

Modelling

In this section, models will be derived to describe the rigid body dynamics of the Crazyflie *UAV*. The intention is to not only use the models for simulating and comparing controllers pre-implementation, but also to find a good model for predicting the system time evolution in the Crayflie firmware, necessary for model based state estimation. The goal of this section is to make evaluation of the discrete time model computationally efficient, while maintaining well conditioned dynamics regardless of the system state.

We will first present a common approach where the system equations are derived with the Euler-Lagrange equations using a Tait-Bryan representation of rotation, here in the extrinsic ZYX form. In contrast to the common Euler angles, this parametrisation uses three different rotational axis and has been used in various *UAV* applications for non-aggressive flight. It suffers on account of the trigonometric terms, which may lead to a Gimbal lock [Lepetit and Fua, 2005] and a complete breakdown of the model. A method of getting around this issue will be presented while maintaining the human readable Tait-Bryan angles, however, due to it's trigonometric terms, the Tait-Bryan angle model remains a poor choice for any real-time implementation. An alternative model will therefore be derived with the Newton-Euler equations using the formalism of quaternions. The models will be compared in terms of controllability and computational complexity with statements on when they should be used. The state-space form of each model will be presented, as well as the analytically derived error dynamics. Finally, means of model reduction and efficient discretisation methods will be presented.

2.1 Tait-Bryan rigid-body dynamics

General quad-rotor dynamics with a Tait-Bryan angle parametrisation of rotation have been the subject of much previous research [Luukkonen, 2011] [Castillo et al., 2005] [Raffo et al., 2010] [Castillo et al., 2004], and

will be presented to clarify nomenclature, enable a discussion on singularities and to serve as a benchmark when developing alternative models. For future reference, we let $\mathbf{p}$ [m] denote the position of the *UAV* centre of mass in a global Cartesian coordinate system, $\boldsymbol{\eta}$ [rad] denote the extrinsic ZYX Tait-Bryan angles of the rigid-body rotation in the global coordinate system and $\boldsymbol{\omega}$ [rad/s] denote the angular velocities of the body coordinate system relative the global coordinate system. Furthermore, Ω_i [rad/s] denotes the angular speed of the rotor i and finally, the vector $\mathbf{s} \in \mathbb{R}^{6\times 1}$ denotes the combined positional and angular of the rigid-body model.

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \boldsymbol{\eta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \quad \boldsymbol{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}, \quad \boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} \mathbf{p} \\ \boldsymbol{\eta} \end{bmatrix} \tag{2.1}$$

For absolute clarity, basis vectors in the respective coordinate systems are written in bold font with a hat $\hat{\cdot}$, where the sub-indexing $\cdot_{\mathcal{G}}$, $\cdot_{\mathcal{I}}$ and $\cdot_{\mathcal{B}}$ refers to a vectors defined in the global, inertial and body coordinate sysrems respectively. The inertial system, sub-indexed $\cdot_{\mathcal{I}}$, is centred at a position $\mathbf{p}$ in the global frame, in the quadcopter centre of mass. Finally, the body frame, sub-indexed $\cdot_{\mathcal{B}}$, is defined with origin in the centre of mass and, rotated by from the inertial frame. This rotation is defined as an isometry with fixed origin in a three-dimensional Euclidean space, $SO(3)$, sub-indexed with $\cdot_{\mathcal{GB}}$ if rotating vectors from the the inertial or global frame to the body frame. Conversely, maps from the body frame to the global or inertial frames are denoted $\cdot_{\mathcal{BG}}$ (see Figure 2.1).

With our choice of Tait-Bryan angles, rotation from the inertial coordinate system to the body frame is done by a sequence of three orthogonal operators in matrix form, which when combined represent a rotation $\mathbf{R} \in SO(3)$ [Palais and Palais, 2007]. By denoting $c_i = \cos(i)$, $s_i = \sin(i)$, the rotation matrices can be defined according to the ZYX order, referring to the sequence in which the rotations are applied. Using rotations around three different axis is commonly referred to as the extrinsic Tait-Bryan representation, in which

$$\mathbf{R}(\psi) = \begin{bmatrix} c_\psi & s_\psi & 0 \\ -s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{R}(\theta) = \begin{bmatrix} c_\theta & 0 & -s_\theta \\ 0 & 1 & 0 \\ s_\theta & 0 & c_\theta \end{bmatrix}, \quad \mathbf{R}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & s_\phi \\ 0 & -s_\phi & c_\phi \end{bmatrix} \tag{2.2}$$

such that the complete rotational operator from the global to body frame is

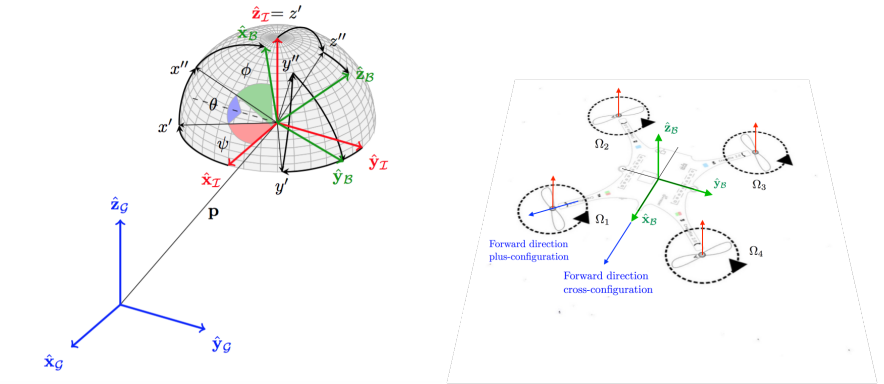

Figure 2.1 *Left:* The global (blue), inertial (*red*) and body (*green*) coordinate system used to describe the quadcopter rotations with the Tait-Bryan angles of roll (ϕ), pitch (θ) and yaw (ψ) with the Tait-Bryan ZYX-convention. *Right:* The Crazyflie with thrusts (red) and body basis vectors in the plus- and cross-configurations. Rotor speeds, $\mathbf{\Omega}$, are defined with Ω_1 and Ω_3 right-handed with respect to $\hat{\mathbf{z}}_{\mathcal{B}}$ while, Ω_2 and Ω_4 are left-handed.

given by

$$\mathbf{R}_{\mathcal{GB}} = \mathbf{R}(\phi)\mathbf{R}(\theta)\mathbf{R}(\psi) = \begin{bmatrix} c_\phi c_\psi & s_\psi c_\phi & -s_\theta \\ c_\psi s_\theta s_\phi - s_\psi c_\phi & s_\psi s_\theta s_\phi + c_\phi c_\psi & c_\theta s_\phi \\ c_\psi s_\theta c_\phi + s_\phi s_\psi & s_\psi s_\theta c_\phi - c_\psi s_\phi & c_\phi c_\theta \end{bmatrix} \in SO(3). \tag{2.3}$$

We here note that by consequence of Euler's rotational theorem [Palais and Palais, 2007], the operator satisfies

$$\boldsymbol{\omega}_{\mathcal{G}} = \mathbf{R}_{\mathcal{GB}}^{-1}\boldsymbol{\omega}_{\mathcal{B}} = \mathbf{R}_{\mathcal{GB}}^{T}\boldsymbol{\omega}_{\mathcal{B}} = \mathbf{R}_{\mathcal{BG}}\boldsymbol{\omega}_{\mathcal{B}}. \tag{2.4}$$

Furthermore, in solving the Newton-Euler and Euler-Lagrange equations in later sections, we require angular rate vector of the quadcopter in the body frame, $\boldsymbol{\omega}_{\mathcal{B}}$, to be expressed in terms of a change in Tait-Bryan angles [Castillo et al., 2005] [Raffo et al., 2010]. The discrepancy is a result of of $\dot{\boldsymbol{\eta}}$ being extrinsic, defined with respect to the inertial coordinate system. The map from the angular rate vector to Tait-Bryan is computed by taking small increments of the Tait-Bryan angles in time, computing the effect on the

rotation vector to find the inverse mapping. Let

$$\boldsymbol{\omega}_{\mathcal{B}} = \mathbf{I}\begin{Bmatrix}\frac{\Delta\phi}{\Delta t}\\0\\0\end{Bmatrix}+\mathbf{R}(\phi)\begin{Bmatrix}0\\\frac{\Delta\theta}{\Delta t}\\0\end{Bmatrix}+\mathbf{R}(\phi)\mathbf{R}(\theta)\begin{Bmatrix}0\\0\\\frac{\Delta\psi}{\Delta t}\end{Bmatrix} = \begin{bmatrix}1 & 0 & -s_\theta\\0 & c_\phi & c_\theta s_\phi\\0 & -s_\phi & c_\theta c_\phi\end{bmatrix}\begin{Bmatrix}\frac{\Delta\phi}{\Delta t}\\\frac{\Delta\theta}{\Delta t}\\\frac{\Delta\psi}{\Delta t}\end{Bmatrix} \tag{2.5}$$

which can be expressed in terms of $\boldsymbol{\eta}$ as

$$\boldsymbol{\omega}_{\mathcal{B}} = \begin{bmatrix}1 & 0 & -s_\theta\\0 & c_\phi & c_\theta s_\phi\\0 & -s_\phi & c_\theta c_\phi\end{bmatrix}\dot{\boldsymbol{\eta}} = \mathbf{W}(\boldsymbol{\eta})\dot{\boldsymbol{\eta}} \tag{2.6}$$

where $\mathbf{W}(\boldsymbol{\eta})$ is the mapping from $\dot{\boldsymbol{\eta}}$ to $\boldsymbol{\omega}_{\mathcal{B}}$ [Castillo et al., 2005]. Here it should be noted that the inverse mapping exists if and only if $\det(\mathbf{W}(\boldsymbol{\eta})) = \cos(\theta) \neq 0$ which is not the case when $\theta = \pi(n+1/2),\ \ n \in \mathbb{N}$, where $\mathbf{W}^{-1}(\boldsymbol{\eta})$ becomes singular, a point which will be addressed in later sections.

Simplified rotor force dynamics From blade element theory, the force generated by the rotor i with a rotor speed, Ω_i $[rad/s]$, is roughly proportional to the rotor speed squared [Bangura et al., 2016]. Consequently, we let

$$f_i \approx k_i\Omega_i^2 \tag{2.7}$$

in the positive $\hat{\mathbf{z}}_{\mathcal{B}}$ direction with some constant, k_i, where the thrust, T $[N]$, is the sum of the rotor forces. The above assumption is common when modelling the forces of rotor UAV's [Chovancová et al., 2014] and was validated by experimental data in **Appendix A.1**. Similarly, the torque around each motor axis is assumed to be approximately

$$\tau_{M_i} \approx b_i\Omega_i^2 + I_M\dot{\Omega}_i \tag{2.8}$$

where b_i is a drag constant and I_M is the rotor inertia [Luukkonen, 2011].

The considered system is symmetric in the $\hat{\mathbf{x}}_{\mathcal{B}}\hat{\mathbf{y}}_{\mathcal{B}}$-plane, where each motor axis is l $[m]$ from the centre of mass. Assuming that $k_i \approx k \in \mathbb{R}^+ \ \ \forall i$, $b_i \approx b \in \mathbb{R}^+ \ \ \forall i$ and that we are using the cross-configuration (see Figure 2.1), the thrust and torque vectors in the body coordinate system can be written

$$\mathbf{T}_{\mathcal{B}} = T\hat{\mathbf{z}}_{\mathcal{B}} \triangleq \begin{bmatrix}0\\0\\k\sum_{i=1}^{4}\Omega_i^2\end{bmatrix},\ \ \boldsymbol{\tau}_{\mathcal{B}}^{\times} = \begin{bmatrix}\tau_\phi\\\tau_\theta\\\tau_\psi\end{bmatrix} \triangleq \begin{bmatrix}kl(-\Omega_1^2-\Omega_2^2+\Omega_3^2+\Omega_4^2)/\sqrt{2}\\kl(-\Omega_1^2+\Omega_2^2+\Omega_3^2-\Omega_4^2)/\sqrt{2}\\\sum_{i=1}^{4}\tau_{M_i}\end{bmatrix} \tag{2.9}$$

Alternatively, we could fly in the more conventional plus configuration, where the $\hat{\mathbf{x}}_{\mathcal{B}}$-axis instead aligns with the arm on which motor 1 is mounted

$$\boldsymbol{\tau}_{\mathcal{B}}^{+} = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \triangleq \begin{bmatrix} kl(-\Omega_2^2 + \Omega_4^2) \\ kl(-\Omega_1^2 + \Omega_3^2) \\ \sum_{i=1}^{4} \tau_{M_i} \end{bmatrix} \tag{2.10}$$

in which special care needs to be taken in the state estimation as the sensory frame is defined with respect to the $\times$-configuration. For future reference, we define the affine maps $\mathbf{M}^{+}_{\Omega^2 \to \tau} \in \mathbb{R}^{4\times4}$ and $\mathbf{M}^{\times}_{\Omega^2 \to \tau} \in \mathbb{R}^{4\times4}$ such that

$$\begin{bmatrix} T & \boldsymbol{\tau}_{\mathcal{B}}^{+} \end{bmatrix}^T = \mathbf{M}^{+}_{\Omega^2 \to \tau} \boldsymbol{\Omega}^2 \quad \text{and} \quad \begin{bmatrix} T & \boldsymbol{\tau}_{\mathcal{B}}^{\times} \end{bmatrix}^T = \mathbf{M}^{\times}_{\Omega^2 \to \tau} \boldsymbol{\Omega}^2 \tag{2.11}$$

respectively, which are both are invertible at all times, as $\det(\mathbf{M}^{\times}_{\Omega^2 \to \tau}) \neq 0$ and $\det(\mathbf{M}^{+}_{\Omega^2 \to \tau}) \neq 0$ if $l, b, k \neq 0$. If a configuration is omitted, the cross-configuration is used by default. Finally, the system identification of the mappings is omitted for brevity, referring to **Appendix A.1** for details on the experiments.

Euler-Lagrange equations In order to use the above definitions and express the rotor dynamics using the principle of conservation of energy, we first define the inertia tensor in the body coordinate system as

$$\mathbf{I}_{\mathcal{B}} = \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix} \tag{2.12}$$

such that the total rotational energy of the *UAV* can be written $E_{rot} = \frac{1}{2}\boldsymbol{\omega}_{\mathcal{B}}^T \mathbf{I}_{\mathcal{B}} \boldsymbol{\omega}_{\mathcal{B}}$. Finally, to make the model more accurate, air resistance is introduced as reactionary force increasing with $\dot{\mathbf{p}}$, similar to viscous friction. This drag matrix is defined in the body frame as

$$\mathbf{D}_{\mathcal{B}} = \begin{bmatrix} D_{11} & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & D_{33} \end{bmatrix} \tag{2.13}$$

where $D_{11} \approx D_{22} < D_{33}$ on account of symmetry. In practice, the drag matrix is usually small enough to a point where it can be disregarded completely, but it is included here to make the simulations more accurate and to make a point commonly missed in the modelling of *UAV* dynamics. In the work of [Luukkonen, 2011] and [Raffo et al., 2010], the drag term is defined with regards to the global translational velocity $\dot{\mathbf{p}}_{\mathcal{G}}$, when it in reality should be defined in the body frame based on $\dot{\mathbf{p}}_{\mathcal{B}}$, as assymetries of the quadcopter geometry makes the drag term dependent on the rigid-body rotation as well as translational speed in the global frame.

With the above definitions, the complete Lagrangian can be written in terms of the systems kinetic, rotational and potential energy as

$$\mathcal{L}(\mathbf{s}, \dot{\mathbf{s}}) = E_{kin} + E_{rot} + E_{pot} = \frac{1}{2} m \dot{\mathbf{p}}_{\mathcal{G}}^T \dot{\mathbf{p}}_{\mathcal{G}} + \frac{1}{2} \boldsymbol{\omega}_{\mathcal{B}}^T \mathbf{I}_{\mathcal{B}} \boldsymbol{\omega}_{\mathcal{B}} - mg \mathbf{p}_{\mathcal{G}} \hat{\mathbf{z}}_{\mathcal{G}} \quad (2.14)$$

for which Euler-Lagrange equations

$$\begin{bmatrix} \mathbf{f} \\ \boldsymbol{\tau} \end{bmatrix} = \frac{d}{dt} \Big(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{s}}} \Big) - \frac{\partial \mathcal{L}}{\partial \mathbf{s}}. \quad (2.15)$$

are solved in accordance with Hamilton's principle [Castillo et al., 2005] [Raffo et al., 2010]. The drag term should not be included, as it pertains to the dissipation rather than the conservation of energy.

Now, if we for the sake of simplicity assume that the drag matrix has a close to constant diagonal such that $D_{11} = D_{22} = D_{33} = d$, which will be validated by in the section on system identification, then it follows that the drag term reduces to

$$m \mathbf{R}_{\mathcal{BG}} \mathbf{D}_{\mathcal{B}} \mathbf{R}_{\mathcal{BG}}^T = md \mathbf{R}_{\mathcal{BG}} \mathbf{I} \mathbf{R}_{\mathcal{BG}}^T = md\mathbf{I} = m\mathbf{D}_{\mathcal{B}} \quad (2.16)$$

by Euler's rotational theorem [Palais and Palais, 2007]. With this approximation, then there are no terms in the energy expression coupling the positional and angular derivatives, and we may proceed by solving the system in terms of (i) translation and (ii) angles independently as pointed out in [Castillo et al., 2005]. If we first consider the translation with the force defined in the global coordinates, then by Newton's second law, equation (2.15) becomes

$$m \ddot{\mathbf{p}}_{\mathcal{G}} = \mathbf{R}_{\mathcal{BG}} \mathbf{T}_{\mathcal{B}} - \mathbf{D}_{\mathcal{B}} \dot{\mathbf{p}}_{\mathcal{G}} - mg \hat{\mathbf{z}}_{\mathcal{G}} \quad (2.17)$$

In similar fashion, we write the rotational energy in the global frame by equation (2.6) as

$$E_{rot} = \frac{1}{2} \boldsymbol{\omega}_{\mathcal{B}}^T \mathbf{I}_{\mathcal{B}} \boldsymbol{\omega}_{\mathcal{B}} = \frac{1}{2} \dot{\boldsymbol{\eta}}^T \mathbf{W}^T(\boldsymbol{\eta}) \mathbf{I}_{\mathcal{B}} \mathbf{W}(\boldsymbol{\eta}) \dot{\boldsymbol{\eta}} = \frac{1}{2} \dot{\boldsymbol{\eta}}^T \mathbf{J}(\boldsymbol{\eta}) \dot{\boldsymbol{\eta}} \quad (2.18)$$

denoting the matrix $\mathbf{J}(\boldsymbol{\eta}) = \mathbf{W}^T(\boldsymbol{\eta}) \mathbf{I}_{\mathcal{B}} \mathbf{W}(\boldsymbol{\eta})$ for convenience. The angular part of the Euler-Lagrange equation then takes the form

$$\boldsymbol{\tau} = \frac{d}{dt} \Big(\frac{1}{2} \dot{\boldsymbol{\eta}}^T \mathbf{J}(\boldsymbol{\eta}) + \frac{1}{2} \mathbf{J}(\boldsymbol{\eta}) \dot{\boldsymbol{\eta}} \Big) - \frac{1}{2} \frac{\partial}{\partial \boldsymbol{\eta}} \Big(\dot{\boldsymbol{\eta}}^T \mathbf{J}(\boldsymbol{\eta}) \dot{\boldsymbol{\eta}} \Big) \quad (2.19)$$

$$= \mathbf{J}(\boldsymbol{\eta}) \ddot{\boldsymbol{\eta}} + \dot{\mathbf{J}}(\boldsymbol{\eta}) \dot{\boldsymbol{\eta}} - \frac{1}{2} \frac{\partial}{\partial \boldsymbol{\eta}} \Big(\dot{\boldsymbol{\eta}}^T \mathbf{J}(\boldsymbol{\eta}) \dot{\boldsymbol{\eta}} \Big) \quad (2.20)$$

$$= \mathbf{J}(\boldsymbol{\eta}) \ddot{\boldsymbol{\eta}} + \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \dot{\boldsymbol{\eta}} \quad (2.21)$$

where the elements of $\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})$ was found using Matlab's symbolic toolbox (see **Appendix A.3**). In summary, the equations of motion in the Tait-Bryan angle representation of rotation is given by

$$\begin{cases} \ddot{\mathbf{p}} = -g\hat{\mathbf{z}}_{\mathcal{G}} + \frac{1}{m}\mathbf{R}_{\mathcal{BG}}\mathbf{T}_{\mathcal{B}} - \frac{1}{m}\mathbf{D}_{\mathcal{B}}\dot{\mathbf{p}} \\ \ddot{\boldsymbol{\eta}} = \mathbf{J}^{-1}(\boldsymbol{\eta})(\boldsymbol{\tau}_{\mathcal{B}} - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}}), \end{cases} \tag{2.22}$$

The same derivation has been done in much previous well cited work, using slightly different nomenclature but arriving at similar results [Luukkonen, 2011] [Raffo et al., 2010] [Castillo et al., 2005]. While making assumptions such as (i) independence between generated rotor thrusts and the translational speeds (2.7), (ii) time-invariance of the rotor model in (2.9) and (iii) simplifications of both the inertial and drag tensors (2.12) (2.13), the model is accurate enough to perform aggressive model based control in real-time applications [Landry, 2015]. As such, the points (i) to (iii) could be addressed to improve performance, but a more pressing issue with respect to aggressive flight is in the parametrisation of rotation which may give rise to dynamical singularities.

Singularities and Gimbal lock The Tait-Bryan angle representation is widely used but may give rise to dynamical singularities during aggressive flights. This is commonly referred to as Gimbal lock [Lepetit and Fua, 2005], and is a symptom of the parametrisation of the $SO(3)$-rotation which is reflected in the map $\mathbf{W}(\boldsymbol{\eta})$ becoming singular for $\theta = (n + 1/2)\pi$, implying a singular matrix $\mathbf{J}(\boldsymbol{\eta})$ by (2.18). Recalling that $\mathbf{R} = \mathbf{R}(\psi)\mathbf{R}(\theta)\mathbf{R}(\phi)$, it is clear that a pitch of $\theta = (n + 1/2)\pi,\ n \in \mathbb{N}$ yields

$$\mathbf{R}(\phi, (n + 1/2)\pi, \psi) = \begin{bmatrix} 0 & 0 & 1 \\ \sin(\phi + \psi) & \cos(\phi + \psi) & 0 \\ -\cos(\phi + \psi) & \sin(\phi + \psi) & 0 \end{bmatrix} \tag{2.23}$$

with the double angle formulae. Here, In this state, changing ψ and ϕ will only rotate the the coordinate system around the z-axis and and we effectively lose one degree of freedom, giving some intuition as to why the singularities arise.

In order to simulate and evaluate control in the entire $\boldsymbol{\eta}$-space, the system (2.22) was augmented to keep the pitch constant at it's most recently feasible value θ_f when sufficiently close to the singularity. The modified system dynamics become

$$\ddot{\boldsymbol{\eta}} = \begin{cases} \mathbf{J}^{-1}(\phi, \theta, \psi)(\boldsymbol{\tau} - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}}), & \text{if} \quad |\cos(\theta)| > \epsilon \\ \mathbf{J}^{-1}(\phi, \theta_f, \psi)(\boldsymbol{\tau} - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}}), & \text{if} \quad |\cos(\theta)| \leq \epsilon \end{cases} . \tag{2.24}$$

In the **Appendix A.4**, it is shown that this is equivalent of bounding the condition number, $\kappa_{\mathbf{J}}(\boldsymbol{\eta}) < \kappa_{max}$, of the $\mathbf{J}$-matrix. As a consequence, the re-

sulting system contains no dynamical singularities in finite time. To demonstrate this modified Tait-Bryan angle model in critical flight, the continuous time model is run with a discrete time controller which will be discussed in later sections. The system is set to follow a lowpass filtered ramp reference in pitch, $\theta_r(t)$, starting at $\theta_r(t = 4) = 0$ $[rad]$ and reaching a full revolution at $\theta_r(t = 8) = 2\pi$. Implementing the feasibility conditions with $\epsilon = 0.1$, we may then simulate flight through two singular regions where the original Tait-Bryan angle dynamics blow up. The epsilon was chosen such that the bound on the condition number of $\mathbf{J}(\boldsymbol{\eta})$ never exceeds 800, but can be made smaller if need be (see Figure 2.2).

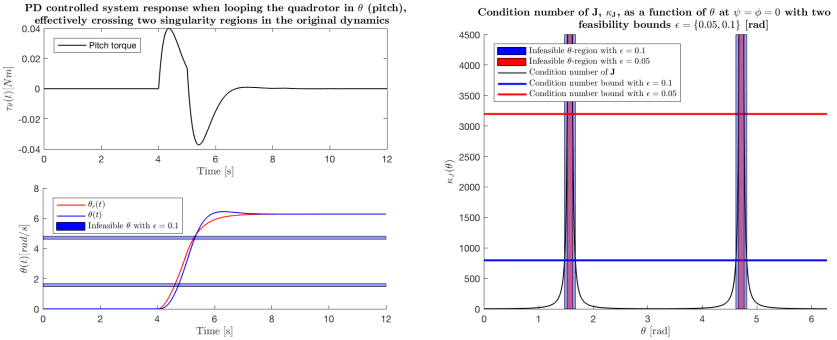

Figure 2.2 *Left:* Simulated system response (blue) when generating a control signal sequence (black) to follow a reference in pitch (red) effectively looping the quadcopter in the θ-direction, passing two singularity regions (blue) where previous models break down. *Right:* The condition number of $\kappa_{\mathbf{J}}(\theta)$ (black) as a function of θ for two numerical limits, $\epsilon_1 = 0.1$ (blue) and $\epsilon_1 = 0.05$ (red) and corresponding bounds, κ_{max}.

2.2 Quaternion rigid-body dynamics

The Tait-Bryan angles are intuitive and simple to use, but the computational complexity required to evaluate the trigonometric functions makes the model suboptimal for a real time implementation. To get around this issue entirely, we will instead make use of the quaternion formalism, which represents the rotation as a four dimensional complex number with one redundant degree of freedom. For this discussion, we will use the Cayley-Dickson construc-

tion [Baez, 2005], where a quaternion is defined as

$$Q = a + bi + cj + dk \in \mathbb{H} \tag{2.25}$$

with $\{a, b, c, d\} \in \mathbb{R}$ and $\{i, j, k\}$ being imaginary units. In this particular construction, some noteworthy and useful algebraic identities are

$$i^2 = j^2 = k^2 = ijk = -1, \quad ij = -ji = k, \quad jk = -kj = i, \quad ki = -ik = j. \tag{2.26}$$

To discuss the geometrical implications of the quaternion, we define it in polar form by Euler's identity

$$Q = e^{(v_x i + v_y j + v_z k)\theta/2} = \cos(\theta/2) + iv_x \sin(\theta/2) + jv_y \sin(\theta/2) + kv_z \sin(\theta/2) \tag{2.27}$$

expressed in the compact notation

$$\mathbf{q} = \begin{bmatrix} \cos(\theta/2) \\ \mathbf{v} \sin(\theta/2) \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ v_x \sin(\theta/2) \\ v_y \sin(\theta/2) \\ v_z \sin(\theta/2) \end{bmatrix} = \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} q_w \\ \mathbf{q}_v \end{bmatrix} \in \mathbb{R}^{4\times 1} \tag{2.28}$$

with a unit vector $\mathbf{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T \in \mathbb{R}^3$. In addition, the real and imaginary parts of the quaternion are notated $\Re\{\mathbf{q}\} = q_w \in \mathbb{R}$ and $\Im\{\mathbf{q}\} = \mathbf{q}_v \in \mathbb{R}^{3\times 1}$ respectively.

Here we may take two distinct approaches in defining the rotation, the Hamilton form, commonly used in the field of robotics and compatible with programs such as *ROS* and *Eigen* [Tully Foote, 2016], or we may opt for the *JPL* form, developed for *NASA* and commonly used in the field of aerospace engineering. The major difference between the two is the definition of the product ij. In the Hamilton form the product is defined as $ij = k$ as in equation (2.26), resulting in a right-handed rotation of θ around $\mathbf{v}$, the rotational axis. However, in the case of the *JPL* formulation, the product is defined as $ij = -k$ making the rotation left handed [Sola, 2012]. For the remainder of this document, we will adopt the Hamilton form, resulting in the quaternion product

$$\mathbf{p} \otimes \mathbf{q} = \begin{bmatrix} p_w q_w - \mathbf{p}_v^T \mathbf{q}_v \\ p_w \mathbf{q}_v + q_w \mathbf{p}_v + \mathbf{p}_v \times \mathbf{q}_v \end{bmatrix} = \begin{bmatrix} p_w q_w - p_x q_x - p_y q_y - p_z q_z \\ p_w q_x + p_x q_w + p_y q_z - p_z q_y \\ p_w q_y - p_x q_z + p_y q_w - p_z q_x \\ p_w q_z + p_x q_y - p_y q_x - p_z q_w \end{bmatrix} \tag{2.29}$$

when using the Cayley-Dickson construction (2.25) obtained using its basic algebraic identities in the Hamilton form (2.26). The operation admits a simple matrix form, as

$$\mathbf{p} \otimes \mathbf{q} = [\mathbf{p}]_L \mathbf{q} = [\mathbf{q}]_R \mathbf{p} \tag{2.30}$$

with

$$[\mathbf{q}]_L = q_w\mathbf{I} + \begin{bmatrix} 0 & -\mathbf{q}_v^T \\ \mathbf{q}_v & [\mathbf{q}_v]_\times \end{bmatrix}, \quad [\mathbf{q}]_R = q_w\mathbf{I} + \begin{bmatrix} 0 & -\mathbf{q}_v^T \\ \mathbf{q}_v & -[\mathbf{q}_v]_\times \end{bmatrix} \tag{2.31}$$

and $[\cdot]_\times$ denoting the skew symmetric operator

$$[\mathbf{q}_v]_\times = \begin{bmatrix} 0 & -q_z & q_y \\ q_z & 0 & -q_x \\ -q_y & q_x & 0 \end{bmatrix}. \tag{2.32}$$

with the inverse map $[\cdot]^\vee$, such that $[[\mathbf{q}_v]_\times]^\vee = \mathbf{q}_v$.

Furthermore, we define the complex conjugate and absolute value of the quaternion similarly to standard complex numbers

$$\mathbf{q}^* = \begin{bmatrix} q_w \\ -\mathbf{q}_v \end{bmatrix}, \qquad ||\mathbf{q}|| = \sqrt{\mathbf{q}^* \otimes \mathbf{q}} = \sqrt{q_w^2 + q_x^2 + q_y^2 + q_z^2} \tag{2.33}$$

which by using equation (2.29) can be shown to satisfy

$$\mathbf{q}^* \otimes \mathbf{q} = \mathbf{q} \otimes \mathbf{q}^* = \begin{bmatrix} ||\mathbf{q}||^2 \\ \mathbf{0} \end{bmatrix}. \tag{2.34}$$

With this result, the inverse of the quaternion can be expressed as

$$\mathbf{q}^{-1} \otimes \mathbf{q} = \mathbf{q} \otimes \mathbf{q}^{-1} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} \Rightarrow \mathbf{q}^{-1} = \frac{\mathbf{q}^*}{||\mathbf{q}||^2}. \tag{2.35}$$

Finally, the quaternion rotation operation from $\mathbf{x}_\mathcal{B}$ to $\mathbf{x}_\mathcal{G}$ in relation to the standard rotation matrix $\mathbf{R}_{\mathcal{BG}} \in SO(3)$ can be written

$$\mathbf{x}_\mathcal{G} = \mathbf{R}_{\mathcal{BG}}\mathbf{x}_\mathcal{B}, \qquad \begin{bmatrix} 0 \\ \mathbf{x}_\mathcal{G} \end{bmatrix} = \mathbf{q}_{\mathcal{BG}} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_\mathcal{B} \end{bmatrix} \otimes \mathbf{q}_{\mathcal{BG}}^* \tag{2.36}$$

as shown using the vector rotation formula in the **Appendix A.5**.

Newton-Euler equations

When representing the system in terms of the previously defined quaternions instead of Tait-Bryan angles, some aspects of the model require new thought. Instead of using the map $\mathbf{W}(\cdot)$, we now need to define the quaternion time derivative in terms of the angular rates vector in the body frame, $\boldsymbol{\omega}_\mathcal{B}$. Through a series of developments, shown in **Appendix A.6**, this relationship is given by

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega}_\mathcal{B} \end{bmatrix}. \tag{2.37}$$

With this result, we may express the angular dependence in the quadcopter model by means of quaternions using the Newton-Euler equations [Fresk and Nikolakopoulos, 2013]. Similarly to the previous derivation with the Euler-Lagrange equations, we denote the externally applied forces by the forces in the global frame, $\mathbf{F}_{\mathcal{G}}$, and torques in the body frame and $\boldsymbol{\tau}_{\mathcal{B}}$ respectively, resulting in

$$\begin{bmatrix}\mathbf{F}_{\mathcal{G}}\\ \boldsymbol{\tau}_{\mathcal{B}}\end{bmatrix} = \begin{bmatrix}m\mathbf{I} & 0\\ 0 & \mathbf{I}_{\mathcal{B}}\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{p}}_{\mathcal{G}}\\ \boldsymbol{\alpha}_{\mathcal{B}}\end{bmatrix} + \begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{B}} \times \mathbf{I}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}}\end{bmatrix}. \tag{2.38}$$

where $\boldsymbol{\alpha}_{\mathcal{B}} = \dot{\boldsymbol{\omega}}_{\mathcal{B}}$. We start by expressing the force equation in the global frame, including the thrust in the body frame, the gravitational acceleration and the drag from the air resistance, which yields the same equation as in the Euler-Lagrange case. Using the relationship (2.37) we see that the angular part of the Newton equation can be expressed by means of the quaternion product, resulting in the complete system

$$\begin{cases}\ddot{\mathbf{p}}_{\mathcal{G}} = \dfrac{1}{m}\Im\left\{\mathbf{q}\otimes\begin{bmatrix}0\\ \mathbf{T}_{\mathcal{B}}\end{bmatrix}\otimes\mathbf{q}^*\right\} - g\hat{\mathbf{z}}_{\mathcal{G}} - \dfrac{1}{m}\mathbf{D}_{\mathcal{G}}\dot{\mathbf{p}}_{\mathcal{G}}\\ \dot{\mathbf{q}} = \dfrac{1}{2}\mathbf{q}\otimes\begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{B}}\end{bmatrix}\\ \dot{\boldsymbol{\omega}}_{\mathcal{B}} = \mathbf{I}_{\mathcal{B}}^{-1}(\boldsymbol{\tau}_{\mathcal{B}} - [\boldsymbol{\omega}_{\mathcal{B}}]_{\times}\mathbf{I}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}})\end{cases} \tag{2.39}$$

which admits a simple state-space representation if writing out the quaternion terms using (2.35) (2.31) (2.33) presented in **Appendix A.7**.

At this point, we have two non-linear rigid-body models of the *UAV* which both may be written in state space form

$$\begin{aligned}\dot{\mathbf{x}}(t) =& \mathbf{A}(\mathbf{x})\mathbf{x}(t) + \mathbf{B}(\mathbf{x})\mathbf{u}(t) + \mathbf{G}\\ \mathbf{y}(t) =& \mathbf{C}(\mathbf{x})\mathbf{x}(t)\end{aligned} \tag{2.40}$$

with different dimensions and definitions of $\{\mathbf{x}, \mathbf{A}, \mathbf{B}, \mathbf{C}\}$ (see **Appendix A.7 A.8**). Despite describing the same reality, the mathematical properties of the models result in cases when they should not be considered. Before these developments, we will first present the rotor dynamics of the *UAV*.

2.3 Rotor dynamics and coupling

While the simplified rotor dynamics outlined in **Section 2.1** may be used for real-time control, the approximations of time-invariance and independence between generated thrust and translational speeds may be crude during aggressive flights [Powers et al., 2013]. Consequently, we will in this section

develop a non-linear rotor model to yield more accurate simulations in evaluating control and enable discussions on current feedback rotor control.

The Crazyflie UAV implements a set of brushed motors (DC), controlled by standard pulse-width modulation (PWM). The amplitude of the PWM signal is $U \in [0, 3.7]$ $[V]$, its carrier frequency is 300 $[kHz]$ with a frequency duty cycle by $d \in [0, 1]$. For the purposes of modelling the complete UAV dynamics, we will in this section describe the dynamics of a common DC motor using Kirchhoff's and Newton's laws [Åström and Murray, 2010] before investigating the effect of attaching a rotor.

Assume that the effective torque generated by the motor, T $[Nm]$, is proportional to the armature current $i(t)$ by some constant K_t $[Nm/A]$ and that there exists viscous friction with some constant b $[Nms]$. Furthermore, we define the rotor moment of inertia as J $[kgm^2]$ and the angular position of the rotor is as $\mu(t)$ $[rad]$, such that $\dot{\mu}(t) = \Omega(t)$ by the previous definitions. Newton's second law of motion then results in

$$J\ddot{\mu}(t) = T - b\dot{\mu}(t) \Leftrightarrow \ddot{\mu}(t) = \frac{1}{J}\Big(K_t i(t) + f_c(\mathbf{p}, \dot{\mathbf{p}}, \hat{\mathbf{z}}_{\mathcal{B}}) - b\dot{\mu}(t)\Big). \tag{2.41}$$

where $f_c(\cdot)$ is a term coupling the previously discussed quad-rotor dynamics with the rotor dynamics. This coupling can be defined to capture effects of translational speeds along the $\hat{\mathbf{z}}_{\mathcal{B}}$-axis and the ground effect during proximity flights [Powers et al., 2013], but is generally disregarded and exact modelling of aerodynamical effects remains an open research question.

Similarly to the mechanical equation, we assume that electro-motive force $e(t)$ to be proportional to $\dot{\mu}(t)$ by a constant K_e $[V/rad/s]$ and let the motor have an inductance of L $[H]$, and resistance of R $[Ohm]$. Applying a voltage $U(t)$ $[V]$ to the motor, Kirchhoff's law results in

$$L\dot{i}(t) = -Ri(t) + U(t) - e(t) \Leftrightarrow \dot{i}(t) = \frac{1}{L}\Big(- Ri(t) + U(t) - K_e\dot{\mu}(t)\Big). \tag{2.42}$$

Combining (2.41) and (2.42) and disregarding coupling terms results in a continuous time system with a state vector $\mathbf{x}(t) = \begin{bmatrix}\mu(t) & \dot{\mu}(t) & i(t)\end{bmatrix}^T$, and a single control signal $u(t) = U(t)$, which may be used for state estimation in the real time implementation for model based rotor speed control.

When attaching the rotor to the shaft, the system exhibits behaviours which that not captured in the linear dynamics. Measuring the rotor speed response when setting the duty cycle from low to high shows that some system parameters change significantly, seemingly with the rotor acceleration (see Figure 2.3).

Consequently, we add a non-linear function to equation (2.41) and assume this non-linearity to be proportional to the angular acceleration by some constant α and dependant on the sign of the velocity, such that the

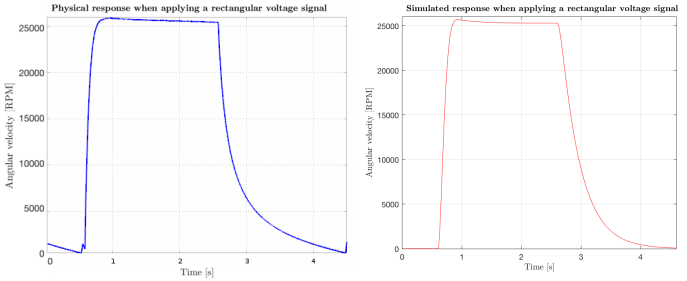

Figure 2.3 *Left:* The rotor speed step response of the physical brushless *DC* rotor subject to a pulse where the duty cycle is set to $d = 1$ over two seconds. *Right:* The rotor speed step response of the modelled non-linear brushed *DC* rotor.

contribution differs based on whether $|\dot{\theta}|$ is increasing or decreasing. The mechanical equation then becomes

$$J\ddot{\mu}(t) = T - b\dot{\mu}(t) + f(\dot{\mu}(t), \ddot{\mu}(t)) \Rightarrow (J - \alpha \text{sgn}(\dot{\mu}(t)\ddot{\mu}(t)))\ddot{\mu}(t) = T - b\dot{\mu}(t) \tag{2.43}$$

which effectively can be thought of as variable rotor moment of inertia, with

$$J = \begin{cases} J^+ & \text{if} \quad \text{sgn}(\dot{\mu}(t) \cdot \ddot{\mu}(t)) > 0 \\ J^- & \text{if} \quad \text{sgn}(\dot{\mu}(t) \cdot \ddot{\mu}(t)) < 0 \end{cases} . \tag{2.44}$$

The single-input single-output (*SISO*) model of the n^{th} rotor is then defined by the system

$$\dot{\mathbf{x}}_n^r(t) = \mathbf{A}_n^r \mathbf{x}_n^r(t) + \mathbf{B}_n^r u_n^r(t) \tag{2.45}$$

$$\mathbf{y}_n^r(t) = \mathbf{C}_n^r \mathbf{x}_n^r(t) \tag{2.46}$$

with a state vector $\mathbf{x}_n^i(t) = \begin{bmatrix} \mu_n(t) & \dot{\mu}_n(t) & i_n(t) \end{bmatrix}^T$, and a single control signal $u_n(t) = U_n(t)$, and

$$\mathbf{A}_n^r = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -b/J^{\pm} & K_t/J^{\pm} \\ 0 & -K_e/L & -R/L \end{bmatrix}, \quad \mathbf{B}_n^r = \begin{bmatrix} 0 \\ 0 \\ 1/L \end{bmatrix}, \quad \mathbf{C}_n^r = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^T . \tag{2.47}$$

The system identification is omitted for brevity but described in **Appendix A.2**, resulting in the parameters used to generate the simulated step response (see Figure 2.3).

2.4 Implementation considerations

Incorporating both a rigid-body model (2.24) (2.39) and four rotor *SISO* models (2.45), the entire *UAV* system is described by the extended state vector,

$$\mathbf{x}^e \triangleq \begin{bmatrix}(\mathbf{x}^r)^T & \mathbf{x}^T\end{bmatrix}^T \triangleq \begin{bmatrix}(\mathbf{x}_1^r)^T & (\mathbf{x}_2^r)^T & (\mathbf{x}_3^r)^T & (\mathbf{x}_4^r)^T & \mathbf{x}^T\end{bmatrix}^T \tag{2.48}$$

containing *UAV* with > 24 states. Such a model will render embedded model based controllers and estimators infeasible if used in its entirety. In this section, a set of assumptions will be made in order to reduce the model order and the analytically linearised systems will be presented for model based control. Finally, a cheap method of discretisation will be given, including constant gravitational terms.

Model reduction If we assume that the translational rigid-body movement does not affect the rotors in any way, then the rotor- and rigid-body *UAV* dynamics may be decoupled. This results in one multiple-input multiple-output (*MIMO*) rigid-body system and four decoupled *SISO* rotor systems which may all be controlled independently (see Figure 2.4). For the rotor control, we may simplify things further by assuming the approximation of the rotor thrust being proportional to the rotor speed squared (2.7). This enables open loop rotor control if restricting the rotation of the rotors to a predefined direction, such that the inverse relationship is unique.

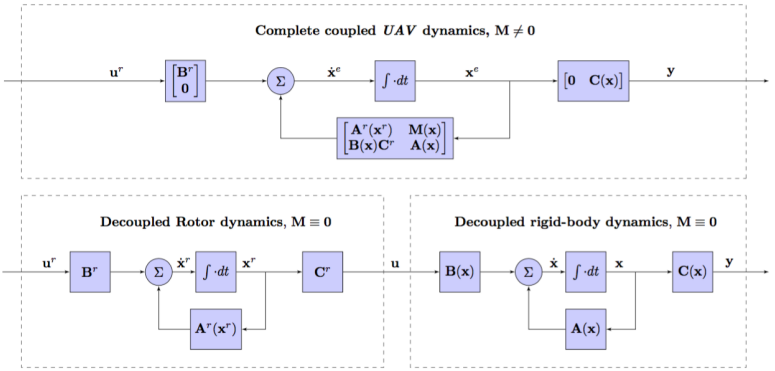

Figure 2.4 *Top*: The complete *UAV* model (top) with a coupling term $\mathbf{M}(\mathbf{x})$ affecting the rotor dynamics. *Bottom*: Decoupling of rotor- and rigid-body systems made possible if assuming that $\mathbf{M}(\mathbf{x}) \equiv \mathbf{0}$.

Linearisation In many considered controllers, the linearized system dynamics are used. Both the Tait-Bryan and quaternion rigid-body models, here represented as a general non-linear system $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$, solely depend on the attitude and its derivative, the quadcopter mass, m, the inertial tensor $\mathbf{I}_{\mathcal{B}}$ and the drag matrix $\mathbf{D}_{\mathcal{B}}$. Consequently, any controller operating using the linearized system will require very little system identification. Given a point in the rigid-body state-space, $\mathbf{x}_0$, either of the two systems can be linearised around a control signal trajectory, $\mathbf{u}_0$, such that the linearised system error dynamics

$$\tilde{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_0(t), \qquad \tilde{\mathbf{u}}(t) = \mathbf{u}(t) - \mathbf{u}_0(t) \tag{2.49}$$

are governed by

$$\dot{\tilde{\mathbf{x}}}(t) = \tilde{\mathbf{A}}_{\Delta\mathbf{x}}\tilde{\mathbf{x}}(t) + \tilde{\mathbf{B}}_{\Delta\mathbf{u}}\tilde{\mathbf{u}}(t). \tag{2.50}$$

the linearized system matrices are then computed as

$$\tilde{\mathbf{A}}_{\Delta\mathbf{x}} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}_0,\mathbf{u}_0}, \qquad \tilde{\mathbf{B}}_{\Delta\mathbf{x}} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}\Big|_{\mathbf{x}_r 0 \mathbf{u}_0}. \tag{2.51}$$

As we are only interested in the Euler-Lagrange system close to a stable hovering point, the linearised system is given in **Appendix A.8** where in addition an analytical expression of the linearised quaternion Newton-Euler system is given for all possible combinations $\{\mathbf{x}_0, \mathbf{u}_0, \mathbf{I}_{\mathcal{B}}, \mathbf{D}_{\mathcal{B}}\}$.

Controllability The synthesis of many controllers, such as the large family of LQR controllers, require complete controllability. For arbitrary linear temporal-invariant (LTI) systems such as the linearised rigid-body dynamics in **Appendix A.8**, we may determine the controllability by showing positive definiteness of the controllability Gramian

$$\mathbf{W}_c(0,\infty) = \int_0^\infty e^{\tilde{\mathbf{A}}_{\Delta\mathbf{x}}\tau}\tilde{\mathbf{B}}_{\Delta\mathbf{x}}\tilde{\mathbf{B}}_{\Delta\mathbf{x}}^T(e^{\tilde{\mathbf{A}}_{\Delta\mathbf{x}}\tau})^T d\tau \in \mathbb{R}^{n\times n}. \tag{2.52}$$

which by the Cayley-Hamilton theorem [Vilfan, 1973] is equivalent to full row-rank of the controllability matrix

$$\mathcal{C} = \begin{bmatrix}\tilde{\mathbf{B}}_{\Delta\mathbf{x}} & \tilde{\mathbf{A}}_{\Delta\mathbf{x}}\tilde{\mathbf{B}}_{\Delta\mathbf{x}} & \cdots & \tilde{\mathbf{A}}_{\Delta\mathbf{x}}^{n-1}\tilde{\mathbf{B}}_{\Delta\mathbf{x}}\end{bmatrix} \in \mathbb{R}^{n\times nm}. \tag{2.53}$$

By this criteria, the Tait-Bryan angle system linearised around a stable hovering state is completely controllable, satisfying $\text{rank}(\mathcal{C}) = 12$. Attempts at proofs of complete controllability for the non-linear Tait-Bryan angle dynamics have been done in a more general sense by Sato et.al., where the property was presumably shown for all attitudes $\boldsymbol{\eta}$ [Sato, 2014]. However, this is not the case for practical purposes. In a simple heuristic experiment, uniformly distributing

$$\boldsymbol{\eta} \sim \mathcal{U}(0, 2\pi), \quad \dot{\boldsymbol{\eta}} \sim \mathcal{U}(-1, 1), \quad T \sim \mathcal{U}(0.1, 0.5) \tag{2.54}$$

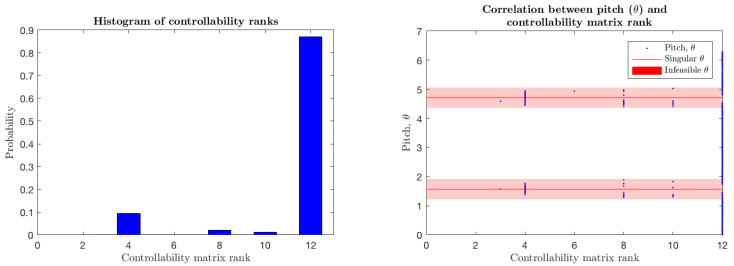

Figure 2.5 *Left:* The histogram of the linearised system controllability matrix rank. *Right*: The correlation between the pitch θ and the system controllability matrix rank, the infeasible region $(n+1)\pi/2 \pm \arccos(\epsilon)$ marked red, and the red lines indicating the dynamical singularities in the $\boldsymbol{\eta}$-space.

and analysing a total of 10^5 linearizations, the probability of finding a fully controllable system is ≈ 0.9 (see Figure 2.5).

The controllability is strongly correlated with the pitch, θ, which greatly reduces system controllability close to $\cos(\theta) = 0$ due to poor conditioning in the angular equations (2.22). In **Section 2.1**, a method was presented to of get around this issue such that complete controllability can be assumed with high probability, making the model viable for time-varying model based control in the entire $\boldsymbol{\eta}$-space.

In contrast, the Newton-Euler quaternion model is not fully controllable, as $\text{rank}(\mathcal{C}) = 12 \neq 13$, caused by the extra degree of freedom introduced by the quaternion. A remedy to this may be to parametrise $q_w = \sqrt{1 - \mathbf{q}_v^T \mathbf{q}_v}$ in the dynamics, but then information is lost regarding the sign of the real part, only allowing for control on one hemisphere of the hypersphere of $\mathbf{q} \in \mathbb{H}$. The issue of guaranteeing controllability in the Newton-Euler equations for LQR synthesis is left as an open topic for future research.

Non-uniqueness and dynamical unwinding There is a fundamental problem in considering multiple means of parametrising the rigid body rotation in the same application. It is easily verified that the quaternion representation is anti-podal due to the quaternion-product being non-commutative, meaning that $\mathbf{R}\{\mathbf{q}\} = \mathbf{R}\{-\mathbf{q}\}$. Creating a rotation matrix from a quaternion is therefore a two-to-one map, while creating the quaternion from a rotation matrix is non-unique and implies a loss of information. As a result, a quaternion generated from a rotation matrix may at times switch between $\mathbf{q}$ and $-\mathbf{q}$, commonly referred to as dynamical unwinding. Similarly, Tait-Bryan parameterisations suffer on account of the trigonometric terms which make

any conversion from Tait-Bryan angles to rotation matrices a or quaternions a many-to-one map.

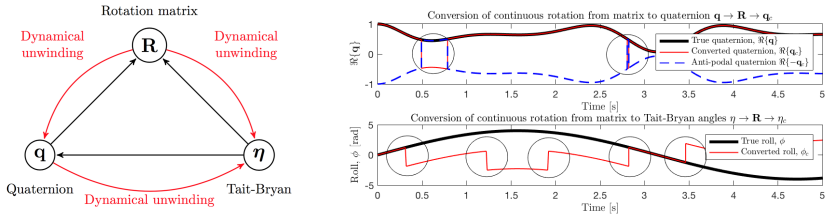

Figure 2.6 *Left*: Safe rotation conversions (black) and conversions susceptible to dynamical unwinding (red). *Top Right*: Two instances of dynamical unwinding marked by circles when creating a quaternion attitude representation, $\mathbf{q}_c(t)$, from a rotation matrix representation, $\mathbf{R}(t)$. *Bottom Right*: Dynamical unwinding in roll $\phi_c(t)$ marked with circles occurring when recreating a Tait Bryan attitude from a continuous rotation matrix, $\mathbf{R}(t)$.

From this we conclude that any application hoping to perform loops and aggressive manoeuvres, must define a controller which operates on a rotational parametrisation containing containing as much, or more, information than the parametrisation used in the estimator. If for instance using a controller operating on the Tait-Bryan angles, the estimator must contain a Tait-Bryan model. If only using a quaternion or rotational matrix formulation with Tait-Bryan control, the system becomes susceptible to dynamical unwinding and may fail. As such, the most general controller will act on the rotation matrix directly, and the most general estimator will operate on the Tait-Bryan angles. With that said, in the real-time application we should instead consider the quaternion formulation as it is more computationally efficient, demonstrating once more why Tait-Bryan angles unsuitable for the aggressive manoeuvres.

Discretization Regardless of the parametrisation of rotation, discrete time integration of the system is required in both the model based control and the state estimators. For this purpose, we assume that the system changes very little over a time interval $[t_k, t_k + h]$ and apply the zero-order hold *ZOH* time integration [Chen and Francis, 2012]. In **Appendix A.9**, it is shown that an arbitrary system with constant terms

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G} \tag{2.55}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{y}(t) \tag{2.56}$$

with $\mathbf{A} \in \mathbb{R}^{M\times M}$, $\mathbf{B} \in \mathbb{R}^{M\times N}$, $\mathbf{G} \in \mathbb{R}^{M\times 1}$, can be discretised to the form

$$\dot{\mathbf{x}}(hk+h) = \mathbf{\Phi}\mathbf{x}(hk) + \mathbf{\Gamma}\mathbf{u}(hk) + \mathbf{\Psi} \tag{2.57}$$

$$\mathbf{y}(hk) = \mathbf{C}\mathbf{x}(hk). \tag{2.58}$$

by letting

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} & \mathbf{G} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{(M+N+1)\times(M+N+1)} \tag{2.59}$$

where then

$$e^{\mathbf{M}\Delta t} = \sum_{k=0}^{\infty} \frac{1}{k!}(\mathbf{M}\Delta t)^k \approx \mathbf{I} + \mathbf{M}\Delta t + O(||(\mathbf{M}\Delta t)||_2^2) = \begin{bmatrix} \mathbf{\Phi} & \mathbf{\Gamma} & \mathbf{\Psi} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}. \tag{2.60}$$

No sub-matrix in the Euler angle or Quaternion state space representations are nilpotent, but the definition of the matrix exponential is still a powerful tool in solving the above equation numerically as the elements of the matrices $\mathbf{I}_{\mathcal{B}}$ and $\mathbf{D}_{\mathcal{B}}$ are $\ll 1$. Higher order terms in the exponential matrix sum will rapidly approach zero, and only including the first order terms in the discretizations will be shown to be a good approximation of the continuous time system.

2.5 Open loop response

To demonstrate the derived Newton-Euler quaternion dynamics and the implementation, the model is simulated with identified parameters of drag and inertia in the plus configuration. The reference rotor speeds, $\mathbf{\Omega}_r$, are defined as sinusoidal splines on five time intervals, deviating only slightly from the signals required to maintain stable hovering position at a thrust of $T = mg$ $[N]$. Furthermore, the model response of translational speed in the global frame $\dot{\mathbf{p}}_{\mathcal{G}}$ is presented, as well as the ZYX Tait-Bryan angle parametrisation, $\boldsymbol{\eta}$, of the attitude quaternion, $\mathbf{q}$ for basic intuition. The example showcases rotor dynamics as well as the derived method of system discretisation using the first order exponential matrix formulation including the constant gravitational term. The response of continuous time system (2.39) in translational velocity and attitude (black) is compared to the discrete time model (2.57) simulated at a rate of 60 $[Hz]$ only including the first order terms in the exponential matrix sum (red, blue and green) (see Figure 2.7).

On the first interval, $I_1 = [0, 0.5)$ $[s]$, rotor speed is increased synchronously across all rotors, effectively accelerating the quadcopter to a slightly elevated position which is reflected in the z-component of the translational response. As the integral of the applied thrust is zero over I_1, the

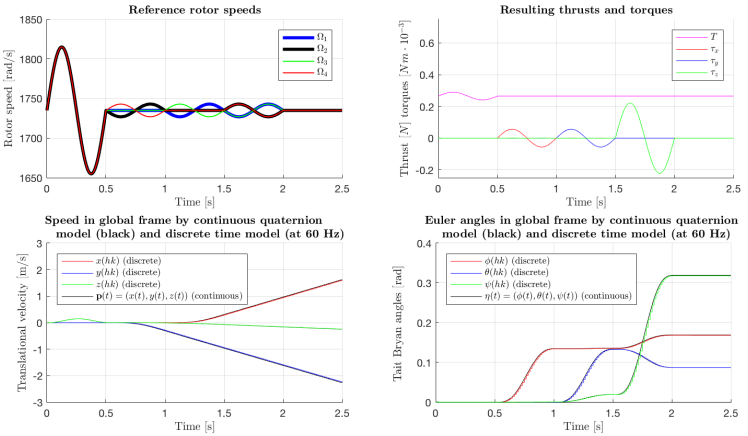

Figure 2.7 Simulation of the continuous and discrete time quaternion *UAV* dynamics. *Top left:* Rotor speeds, $\mathbf{\Omega}$ [rad/s], deviating slightly from a stable hovering state by sinusoid splines. *Top right:* Resulting thrust $[N]$ and torques $[Nm]$ of the rigid body system. *Bottom left:* Translational speed in the global frame with the continuous time model (black) and with the discrete time model (red, blue and green). *Bottom right:* The quaternion attitude response, transformed into Tait-Bryan angles in continuous time (black) and discrete time (red, blue and green).

speed is clearly $\dot{z}(0) = \dot{z}(0.5) = 0$ as a consequence of the conservation of energy. Furthermore, the sequence will not generate any difference in rotor speeds and thereby no torques, which is accurately reflected by the force and torque response.

On the second interval, $I_2 = [0.5, 1.0)$ $[s]$, the rotor speeds of motor 2 and 4 are varied sinusoidally with a phase offset of π $[rad]$, which generates a torque about the x-axis where the integral of the torque is zero over I_2, hence the roll increases while the remaining Tait-Bryan angles remain fixed. The same reasoning applies to the second interval, $I_3 = [1.0, 1.5)$ $[s]$, where a torque about the y-axis rotates the already rotated frame and thereby induces a change in not only pitch θ but also yaw ψ. The reason ψ is affected by a rotation about the y-axis is due to the discrepancy between our choice of Tait-Bryan angles and the body rates with respect to which the torques are defined, recall $\boldsymbol{\omega}_{\mathcal{B}} = \mathbf{W}(\boldsymbol{\eta})\boldsymbol{\eta}_{\mathcal{G}}$.

On the fourth interval, $I_4 = [1.5, 2.0)$ $[s]$, the rotor speeds induce a positive torque about the z-axis which cases the attitude to shift more dramatically.

This is partly due to all four rotors are cooperating in generating the torque, which around the z-axis is governed by different parameters than the x and y axes as shown in (2.9), (2.7) and(2.8). In addition, the inertia tensor $\mathbf{I}_{\mathcal{B}}$ does not have a constant diagonal due to geometric asymmetries of the quad-rotor.The attitude remains constant at some $\mathbf{R} \neq \mathbf{I}$ and in this state the thrust required to hover is not counteracting gravity. As a consequence the z-component of the velocity grows negatively, eventually causing a crash with the ground. Furthermore, the velocities increase linearly which is to be expected during constant accelerations induced by constant thrust, and the discretisation developed to its first order approximation provides an efficient tool for evaluating the model in a real-time context at rates of 60 $[Hz]$.

2.6 Summary

In this section, we have provided the definitions necessary to understand the dynamics of the quad-rotor *UAV* in the context of both a Tait-Bryan and quaternion parametrisation of the *SO(3)* rotation, detailing both the Newton-Euler and Euler-Lagrange approaches. We have performed basic system identification of the maps from duty cycle to rotor thrust which validated results from blade element theory [Bangura et al., 2016], and devised the maps from thrust and torques to desired *PWM* duty cycle in the brushless motors in both the "+"- and "×"-configurations respectively.

Furthermore, a continuous time rotor model has been derived to more accurately simulate the system when synthesising and comparing controllers. We have shown that the Tait-Bryan angle model may fail, and therefore proposed a method for preserving dynamical feasibility and controllability throughout the entire $\boldsymbol{\eta}$-space. The quaternion model on the other hand is not fully controllable and therefore not suited for synthesis of *LQR* controllers. The rotation were discussed to illustrate potential problems when using multiple parameterisations of rotation in the same system during aggressive flight. Finally, analytical state space linearisations of the quaternion and Tait-Bryan systems were given, and a computationally efficient method of discretisation was derived.

3

Motion planning

In all conceivable quadcopter applications, robust motion planning is of outmost importance. From an architectural perspective, this is arguably the most challenging aspect of the thesis, as it strives to meet the needs of Bitcraze *AB*, as well as the Robotics and Autonomous Systems Center (*RASC*) at *USC*, the *CDS* departement of *ETH* and the Media Lab at *MIT*. Naturally, a wide variety of problems had to be solved, and the specifications of the system can be summarised in the following seven points (i)-(vii).

(i) Compliance with the rigid-body dynamics.

(ii) Being economical in terms of power usage.

(iii) Avoiding paths which intersect known static obstacles.

(iv) Avoiding dynamic obstacles.

(v) Loading and on-line evaluation of precomputed trajectories represented with minimal information.

(vi) Compliance with current academic- and industry standards with .

(vii) Synchronising trajectory evaluation across multiple *UAV*'s.

(viii) Scheduling events.

The above specifications have all been met in our implementation, and therefore serves as holistic view of the real-time application. The specifications (i)-(iv) are handled on a host host computer, and (v-viii) are accomplished in the embedded system. This chapter mainly detalils the mathematics required to solve the problems, referring to [Greiff, 2017] for the code, documentation and supplemental notes on the real-time implementation.

The first section concerns the model compliance, (i), and representation of trajectories in minimal information, (v), with a discussion on differential flatness. Here it will be shown that all states and control signals in the systems

can be determined from a set of flat outputs, $\boldsymbol{\gamma}$, presenting the equations as implemented in the Crazyflie firmware. The theory, while requiring the fourth order derivative of $\boldsymbol{\gamma}$ to be defined, allows for cheap on-line computation of feed forward terms in any considered control system.

The second section concerns the generation of trajectories in flat output space. Here we handle specification (ii) by equating the power consumption to a distance travelled and solving a multi-dimensional travelling salesman problem (*TSP*). For this purpose, a genetic algorithm (*GA*) is proposed to mutate an initial feasible solution to a close-to-optimal solution with an upper bound on computational time. The derived algorithm is capable of assigning different priorities to subsets of M-dimensional points, $\mathcal{P} \in \mathbb{R}^{M \times N}$, with corresponding times, $\mathbf{T} \in \mathbb{R}^{N}$ such as the higher priority points are guaranteed to be passed first. This algorithm is particularly useful if the points are considered as "jobs" at which the quadcopter needs to perform a task, such as reading a bar code or otherwise interacting with the environment.

The third section outlines how the points $\mathcal{P}$ from the *TSP-GA* solution may be used to parametrise trajectories in flat output space, allowing the safe evaluation of the flatness equations in the real-rime application and simultaneously satisfying (ii), (iii) and (vi). For this purpose, we consider a method of polynomial interpolation [Richter et al., 2013] with a polynomial degree n, where a constrained quadratic program (*QP*) is solved to minimise the fourth positional derivative of a trajectory. The resulting "minimum snap" trajectory is represented by a set of N polynomial coefficients, $\mathbf{P} \in \mathbb{R}^{N(n+1)}$, and times, $\mathbf{T}$. When choreographing many Crazyflies, programs such as Blender is often used [Blender Online Community, 2016]. Here cubic Bezier curves are used to create coordinated movement instead of polynomial splines, creating set of control points $\mathcal{P}_B \in \mathbb{R}^{N \times 4}$ and times $\mathbf{T}$ characterising the Bezier curve [Jolly et al., 2009]. In addition, many users parametrise circular movement by sinusoid trajectories described in terms of an amplitude $\mathbf{A}$, initial offset $\mathbf{B}$, frequency and phase $\boldsymbol{\omega}$, $\boldsymbol{\varphi}$ as well as a time during which the trajectory should be followed $\mathbf{T}$. Finally, we also provide a method of evaluating the linear segments from the *GA* directly by means of high-order smoothing.

The final specifications of synchronisation (vii) and event scheduling (viii) were both met by modifying the Crazyflie driver, originally implemented in [Hoenig et al., 2015], referring to [Greiff, 2017] for additional details on the modifications. In summary, the proposed architecture will be shown to satisfy the specifications (i)-(viii) and is general enough to allow the flat outputs to be defined not only by compositions of the above mentioned trajectories, but also have them start and finish at different points in time. Furthermore, it allows the pre-loading, re-loading and synchronisation of trajectories, enabling autonomous flight completely free from delays in communication between the host computer and the Crazyflie.

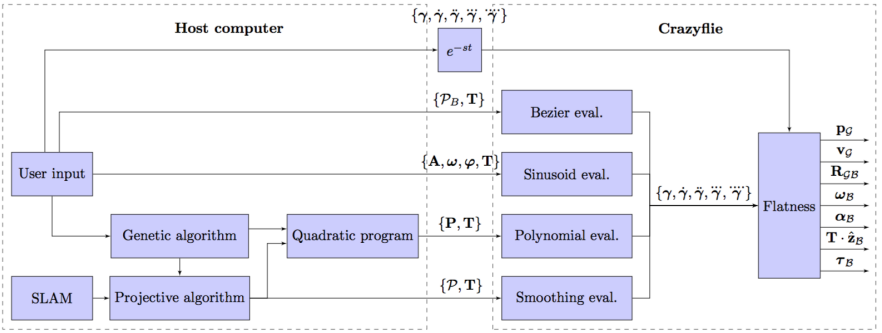

Figure 3.1 Embedded motion planning architecture in the Crazyflie.

3.1 Differential flatness

Differential flatness was first introduced by means of differential algebra [Fliess et al., 1992], and has more recently been explored using Lie-Bäckman transformation theory [Fliess et al., 1999]. Consider a very general non-linear system, defined by $\mathbf{x} \in \mathbb{R}^{M\times 1}$, $\mathbf{u} \in \mathbb{R}^{N\times 1}$, and $M \geq N$, which has the property of being differentially flat. We may the find a set of flat outputs $\boldsymbol{\gamma} \in \mathbb{R}^{N\times 1}$,

$$\boldsymbol{\gamma} = \mathbf{h}(\mathbf{x}, \mathbf{u}, \dot{\mathbf{u}}, \cdots, \mathbf{u}^{(r)}) \tag{3.1}$$

such that

$$\mathbf{x} = \mathbf{f}_\alpha(\boldsymbol{\gamma}, \dot{\boldsymbol{\gamma}}, \cdots, \boldsymbol{\gamma}^{(q)}), \qquad \mathbf{u} = \mathbf{f}_\beta(\boldsymbol{\gamma}, \dot{\boldsymbol{\gamma}}, \cdots, \boldsymbol{\gamma}^{(q)}). \tag{3.2}$$

which allows the identification of all system states, including control signals from the $\boldsymbol{\gamma}$ parameters without integration. If this property may be proven, it provides a cheap method of evaluating a trajectory's dynamical feasibility. In addition, our specific rigid-body system (2.39) contains four input signals, implying that motion planning need only be done in at most four dimensions, minimising the information with which the trajectory is represented.

The Newton-Euler quaternion dynamics are fully actuated, symmetric and holonomic, the system is likely therefore likely differentially flat in it's centre of mass [Murray et al., 1995]. As shown by many before us, flatness can indeed be proven with $q = 4$ for simplified Newton Euler equations excluding drag using a ZXY Tait-Bryan angle representation [Mellinger and Kumar, 2011]. However, it should be noted that this need not be the case when adopting the quaternion rotations due to the extra degree of freedom introduced. Consequently, we will derive the flatness equations using the ZYX Tait-Bryan angles and quaternion parametrization of the $SO(3)$. First

for the angular states, the for the angular rates, and finally for the angular accelerations and control signal torques.

The equations will be tested by simulating the *PD*-stabilised continuous time quaternion quad-rotor dynamics, numerically deriving the system response of some chosen flat output $\boldsymbol{\gamma}$ to the fourth derivative and then attempting to recreate the original system response using the flatness equations. If accurate, we should in each subsection be able to derive the exact states and control signals based solely on $\boldsymbol{\gamma}$, yielding a very visual accuracy test and interpretation of the theory. The controllers will be discussed in later sections and are omitted here, but when simulating the system, we let roll, pitch and yaw follow first order lowpass-filtered unit steps, such that $(\phi_r(t), \theta_r(t), \psi_r(t)) \in [-0.8, 0.8]\ [rad]$, and let the set-point in elevation follow a sinusoidal reference with that $z_r(t) = 0.5\sin(0.75t)\ [m]$ (see Figure 3.2).

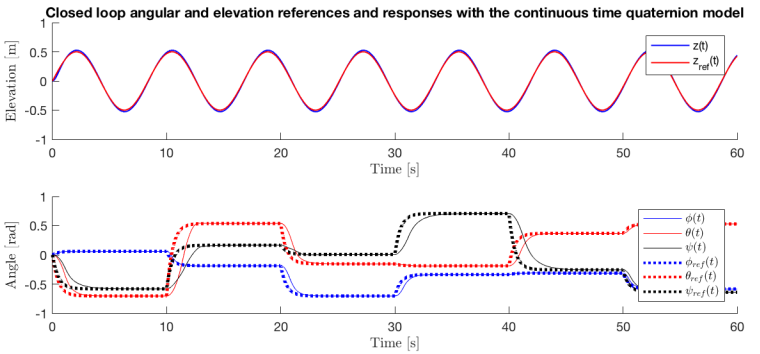

Figure 3.2 Closed PD-loop simulation of attitude and elevation of the quaternion dynamics used to validate the flatness equation derivation and implementation.

For future reference in the derivation of the *DF*-equations, we notate for body rates and body accelerations relative the inertial frame by,

$$\dot{\boldsymbol{\omega}}_{\mathcal{B}} = \begin{bmatrix}\dot{\omega}_x & \dot{\omega}_y & \dot{\omega}_z\end{bmatrix} = \begin{bmatrix}\alpha_x & \alpha_y & \alpha_z\end{bmatrix} = \boldsymbol{\alpha}_{\mathcal{B}}\ \ [rad/s^2], \tag{3.3}$$

recall the cross product identity

$$\mathbf{u} \cdot (\mathbf{v} \times \mathbf{u}) = \mathbf{v} \cdot (\mathbf{u} \times \mathbf{u}) = 0\ \ \forall\ \ \mathbf{u}, \mathbf{v} \in \mathbb{R}^3, \tag{3.4}$$

and note that the time derivative of a rotation $\mathbf{R}_{\mathcal{GB}} \in SO(3)$ can be written

$$\dot{\mathbf{R}}_{\mathcal{BG}} = \mathbf{R}_{\mathcal{BG}}[\boldsymbol{\omega}_{\mathcal{B}}]_\times \Leftrightarrow \dot{\mathbf{R}}_{\mathcal{GB}} = -[\boldsymbol{\omega}_{\mathcal{B}}]_\times \mathbf{R}_{\mathcal{GB}} \tag{3.5}$$

as shown in **Section A.11**.

Angular states and attitude As we are likely to control the position of the quad-rotor in space, the quadcopter position $\mathbf{p}(t) = (x(t), y(t), z(t))$ in the global frame is included in $\boldsymbol{\gamma}$. The last output is chosen as the yaw angle $\psi(t)$ as in [Mellinger and Kumar, 2011], for reasons which will become apparent. For this and all future discussion, we define the outputs

$$\boldsymbol{\gamma}(t) = \begin{bmatrix}\gamma_1(t) & \gamma_2(t) & \gamma_3(t) & \gamma_4(t)\end{bmatrix}^T \triangleq \begin{bmatrix}x(t) & y(t) & z(t) & \psi(t)\end{bmatrix}^T \tag{3.6}$$

and assume that $\boldsymbol{\gamma}^{(q)}$ exists for $q \leq 4$. To find the attitude, we simply need to define $\phi(t)$ and $\theta(t)$ as a function of the flat outputs, which can be done by the $\ddot{\mathbf{p}}$-terms. The unit vector $\hat{\mathbf{z}}_{\mathcal{B}}$ can be expressed in terms of $\tilde{\mathbf{a}} = [\ddot{\gamma}_1, \ddot{\gamma}_2, \ddot{\gamma}_3 + g]^T$ as

$$\hat{\mathbf{z}}_{\mathcal{B}} = \frac{\tilde{\mathbf{a}}}{||\tilde{\mathbf{a}}||_2}, \tag{3.7}$$

as the rotors, by design, are incapable of rotating with negative rotor speeds. Analogously, the thrust control signal is simply

$$T = m||\tilde{\mathbf{a}}||_2. \tag{3.8}$$

Now, if we denote the unit vector $\hat{\mathbf{y}}_{\mathcal{C}} = [\cos(\gamma_4 + \pi/2), \sin(\gamma_4 + \pi/2), 0]^T$ as the $\hat{\mathbf{y}}_{\mathcal{G}}$ rotated ψ around $\hat{\mathbf{z}}_{\mathcal{G}}$, we may construct the unit vectors of the body frame as

$$\hat{\mathbf{x}}_{\mathcal{B}} = \frac{\hat{\mathbf{y}}_{\mathcal{C}} \times \hat{\mathbf{z}}_{\mathcal{B}}}{||\hat{\mathbf{y}}_{\mathcal{C}} \times \hat{\mathbf{z}}_{\mathcal{B}}||_2}, \qquad \hat{\mathbf{y}}_{\mathcal{B}} = \hat{\mathbf{z}}_{\mathcal{B}} \times \hat{\mathbf{x}}_{\mathcal{B}} \tag{3.9}$$

recalling that we are using the conventional ZYX Tait-Bryan angle representation of rotation, differing from the derivation in [Mellinger and Kumar, 2011]. Determining the parameterisations of rotation can then be done by solving the equation

$$\mathbf{R}_{\mathcal{GB}}\begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{G}} & \hat{\mathbf{y}}_{\mathcal{G}} & \hat{\mathbf{z}}_{\mathcal{G}}\end{bmatrix} = \mathbf{R}_{\mathcal{GB}}\mathbf{I} = \mathbf{R}(\psi)\mathbf{R}(\theta)\mathbf{R}(\phi) = \begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{B}} & \hat{\mathbf{y}}_{\mathcal{B}} & \hat{\mathbf{z}}_{\mathcal{B}}\end{bmatrix} \tag{3.10}$$

giving the rotation matrix and implicitly the quaternion attitude, $\mathbf{q}$, though not uniquely due to the quaternion being anti-podal (see **Section 2.4**). Consequently, the system is not completely differentially flat, however, the same can be said with the commonly used Euler angle parametrisation which is used successfully in many applications [Mellinger and Kumar, 2011] [Landry, 2015]. The only time caution must be taken is when performing looping manoeuvres, in which case controllers acting directly on the rotation matrix should be implemented. In addition, the equations become singular when $\hat{\mathbf{y}}_{\mathcal{C}}||\hat{\mathbf{z}}_{\mathcal{B}}$ by (3.9) which should be avoided at all times. The derived equations are here demonstrated by finding the real and imaginary parts of the attitude quaternion $\mathbf{q}(t)$ from the flat outputs $\boldsymbol{\gamma}(t)$, yielding a perfect reconstruction (see Figure 3.3).

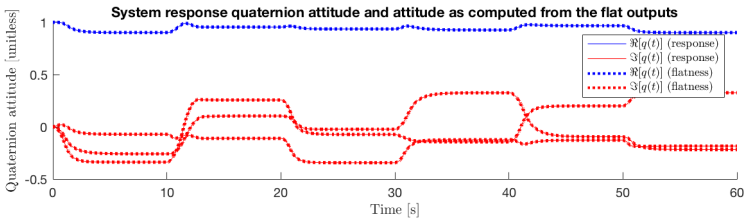

Figure 3.3 Closed PD-loop simulated attitude (line) and re-created attitude as computed by the flatness equations (dotted).

Angular velocities In determining the angular velocities of the system, $\boldsymbol{\omega}_{\mathcal{B}}$, we may use the translational equation in the Newton-Euler system

$$m\ddot{\mathbf{p}} = T\hat{\mathbf{z}}_{\mathcal{B}} - mg\hat{\mathbf{z}}_G. \tag{3.11}$$

The system may be differentiated with respect to time, where then

$$\frac{d}{dt}\left(m\ddot{\mathbf{p}}\right) = \frac{d}{dt}\left(T\hat{\mathbf{z}}_{\mathcal{B}} - mg\hat{\mathbf{z}}_G\right) \Rightarrow m\mathbf{p}^{(3)} = \dot{T}\hat{\mathbf{z}}_{\mathcal{B}} + \boldsymbol{\omega}_{\mathcal{B}} \times T\hat{\mathbf{z}}_{\mathcal{B}}. \tag{3.12}$$

By projecting the expression along the $\hat{\mathbf{z}}_{\mathcal{B}}$ unit vector and using the cross product identity (3.4), the thrust derivative in the body frame can be written as

$$\dot{T} = \hat{\mathbf{z}}_{\mathcal{B}} \cdot \dot{T}\hat{\mathbf{z}}_{\mathcal{B}} = m\hat{\mathbf{z}}_{\mathcal{B}} \cdot \mathbf{p}^{(3)}. \tag{3.13}$$

Combining equations (3.12) and (3.13), the cross product with angular rates can be written

$$\boldsymbol{\omega}_{\mathcal{B}} \times \hat{\mathbf{z}}_{\mathcal{B}} = \frac{m}{T}(\mathbf{p}^{(3)} - (\hat{\mathbf{z}}_{\mathcal{B}} \cdot \mathbf{p}^{(3)})\hat{\mathbf{z}}_{\mathcal{B}}) \tag{3.14}$$

which can be thought of as the projection of $\mathbf{p}^{(3)}$ onto the plane spanned by $\hat{\mathbf{x}}_{\mathcal{B}}$ and $\hat{\mathbf{y}}_{\mathcal{B}}$. It is then evident that the angular components are

$$\omega_x = -\hat{\mathbf{y}}_{\mathcal{B}} \cdot (\boldsymbol{\omega}_{\mathcal{B}} \times \hat{\mathbf{z}}_{\mathcal{B}}) \tag{3.15}$$

$$\omega_y = \hat{\mathbf{x}}_{\mathcal{B}} \cdot (\boldsymbol{\omega}_{\mathcal{B}} \times \hat{\mathbf{z}}_{\mathcal{B}}) \tag{3.16}$$

where the final component, ω_z, can be found by solving the associated angular equation

$$\mathbf{R}_{\mathcal{GB}}\begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{G}} & \hat{\mathbf{y}}_{\mathcal{G}} & \hat{\mathbf{z}}_{\mathcal{G}}\end{bmatrix}\begin{bmatrix}\omega_x \\ \omega_y \\ \omega_z\end{bmatrix} = \mathbf{R}_{\mathcal{GB}}\mathbf{I}\begin{bmatrix}\omega_x \\ \omega_y \\ \omega_z\end{bmatrix} = \begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{B}} & \hat{\mathbf{y}}_{\mathcal{C}} & \hat{\mathbf{z}}_{\mathcal{G}}\end{bmatrix}\begin{bmatrix}\dot{\phi} \\ \dot{\theta} \\ \dot{\psi}\end{bmatrix} \tag{3.17}$$

where the yaw derivative is known as $\dot{\psi} = \dot{\gamma}_4$ is a flat output. Denoting the matrix product $\mathbf{A} = \begin{bmatrix}\hat{\mathbf{x}}_\mathcal{B} & \hat{\mathbf{y}}_\mathcal{C} & \hat{\mathbf{z}}_\mathcal{G}\end{bmatrix}^{-1} \mathbf{R}_{\mathcal{GB}}$, with elements a_{ij}, we may express the missing Tait-Bryan angles and body rates as

$$\omega_z = \frac{\dot{\psi} - a_{31}\omega_x - a_{32}\omega_y}{a_{33}} \tag{3.18}$$

$$\dot{\phi} = a_{11}\omega_x + a_{12}\omega_y + a_{13}\omega_z \quad . \tag{3.19}$$

$$\dot{\theta} = a_{21}\omega_x + a_{22}\omega_y + a_{23}\omega_z \tag{3.20}$$

These equations are shown to yield perfect reconstruction of the pitch and roll rates as well as the body angular rate about $\hat{\mathbf{z}}_\mathcal{B}$ (see Figure 3.4).

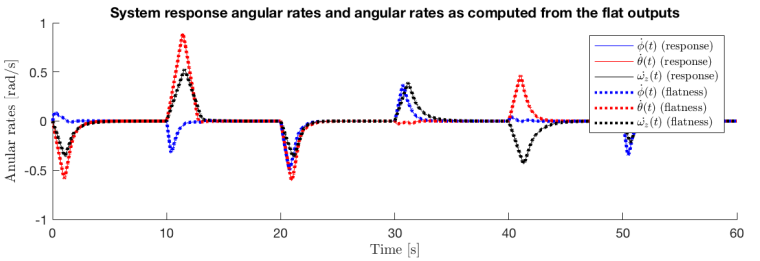

Figure 3.4 Closed PD-loop simulated angular rates and rates as computed by the flatness equations.

Angular accelerations We now seek the angular accelerations in the body frame, $\boldsymbol{\alpha}_\mathcal{B}$, before finally using the inverse dynamics to compute the control signal torques. This can be done similarly to the angular velocities, by first differentiating equation (3.11) twice with respect to time

$$m\mathbf{p}^{(4)} = \ddot{T}\hat{\mathbf{z}}_\mathcal{B} + 2\boldsymbol{\omega}_\mathcal{B} \times \dot{T}\hat{\mathbf{z}}_\mathcal{B} + \boldsymbol{\omega}_\mathcal{B} \times (\boldsymbol{\omega}_\mathcal{B} \times T\hat{\mathbf{z}}_\mathcal{B}) + \boldsymbol{\alpha}_\mathcal{B} \times T\hat{\mathbf{z}}_\mathcal{B} \tag{3.21}$$

Similarly to the case of angular velocities, projecting the above expression along $\mathbf{z}_\mathcal{B}$ and using the identity (3.4) results in

$$\ddot{T} = \hat{\mathbf{z}}_\mathcal{B} \cdot \ddot{T}\hat{\mathbf{z}}_\mathcal{B} = \hat{\mathbf{z}}_\mathcal{B} \cdot \underbrace{\left[m\mathbf{p}^{(4)} - \boldsymbol{\omega}_\mathcal{B} \times (\boldsymbol{\omega}_\mathcal{B} \times T\hat{\mathbf{z}}_\mathcal{B})\right]}_{\mathbf{v}} = \hat{\mathbf{z}}_\mathcal{B} \cdot \mathbf{v}. \tag{3.22}$$

Equation (3.21) can then be written in terms of the vector $\mathbf{v}$, as

$$\mathbf{v} = (\hat{\mathbf{z}}_\mathcal{B} \cdot \mathbf{v})\hat{\mathbf{z}}_\mathcal{B} + 2\left[\boldsymbol{\omega}_\mathcal{B} \times m(\hat{\mathbf{z}}_\mathcal{B} \cdot \mathbf{p}^{(3)})\hat{\mathbf{z}}_\mathcal{B}\right] + \boldsymbol{\alpha}_\mathcal{B} \times T\hat{\mathbf{z}}_\mathcal{B} \tag{3.23}$$

where we again used the expression for $\dot{T}$ derived in (3.13). Now, the x and y components can be computed similarly to the angular velocities by letting

$$\boldsymbol{\alpha}_{\mathcal{B}} \times \hat{\mathbf{z}}_{\mathcal{B}} = \frac{1}{T}\Big[\mathbf{v} - \hat{\mathbf{z}}_{\mathcal{B}} \cdot \mathbf{v} - 2\boldsymbol{\omega}_{\mathcal{B}} \times m(\hat{\mathbf{z}}_{\mathcal{B}} \cdot \mathbf{p}^{(3)})\hat{\mathbf{z}}_{\mathcal{B}}\Big] \tag{3.24}$$

where then

$$\alpha_x = -\hat{\mathbf{y}}_{\mathcal{B}} \cdot (\boldsymbol{\alpha}_{\mathcal{B}} \times \hat{\mathbf{z}}_{\mathcal{B}}) \tag{3.25}$$

$$\alpha_y = \hat{\mathbf{x}}_{\mathcal{B}} \cdot (\boldsymbol{\alpha}_{\mathcal{B}} \times \hat{\mathbf{z}}_{\mathcal{B}}) \tag{3.26}$$

In order to compute the z-component of the angular accelerations, we differentiate equation (3.17) with respect to time using the time derivative of the rotation as expressed in equation (3.5),

$$([\boldsymbol{\omega}_{\mathcal{B}}]_\times \mathbf{R}_{\mathcal{GB}})\boldsymbol{\omega}_{\mathcal{B}} + \mathbf{R}_{\mathcal{GB}}\boldsymbol{\alpha}_{\mathcal{B}} = \boldsymbol{\omega}_{\mathcal{B}} \times \dot{\phi}\hat{\mathbf{x}}_{\mathcal{B}} + \boldsymbol{\omega}_{\mathcal{C}} \times \dot{\phi}\hat{\mathbf{y}}_{\mathcal{C}} + \begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{B}} & \hat{\mathbf{y}}_{\mathcal{C}} & \hat{\mathbf{z}}_{\mathcal{G}}\end{bmatrix} \ddot{\boldsymbol{\eta}} \Leftrightarrow \mathbf{A}\boldsymbol{\alpha}_{\mathcal{B}} + \mathbf{b} = \ddot{\boldsymbol{\eta}} \tag{3.27}$$

where $\mathbf{A} = \begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{B}} & \hat{\mathbf{y}}_{\mathcal{C}} & \hat{\mathbf{z}}_{\mathcal{G}}\end{bmatrix}^{-1} \mathbf{R}_{\mathcal{GB}} \in \mathbb{R}^{3\times 3}$, just as in the case of the angular velocities, and the vector $\mathbf{b} \in \mathbb{R}^{3\times 1}$ is given by

$$\mathbf{b} = \begin{bmatrix}\hat{\mathbf{x}}_{\mathcal{B}} & \hat{\mathbf{y}}_{\mathcal{C}} & \hat{\mathbf{z}}_{\mathcal{G}}\end{bmatrix}^{-1} \left[([\boldsymbol{\omega}_{\mathcal{B}}]_\times \mathbf{R}_{\mathcal{GB}})\boldsymbol{\omega}_{\mathcal{B}} - \boldsymbol{\omega}_{\mathcal{B}} \times \dot{\phi}\hat{\mathbf{x}}_{\mathcal{B}} - \boldsymbol{\omega}_{\mathcal{C}} \times \dot{\phi}\hat{\mathbf{y}}_{\mathcal{C}}\right]. \tag{3.28}$$

As $\ddot{\psi}(t) = \dddot{\gamma}_4(t)$ is a known flat output, the z-component of the angular acceleration is then given by

$$\alpha_z = \frac{\ddot{\psi} - b_{31} - a_{31}\omega_x - a_{32}\omega_y}{a_{33}}. \tag{3.29}$$

At this point, we have full knowledge of the system states and their time derivatives from the flat outputs. Consequently, the system torques can be computed by means of inverting the angular dynamics

$$\boldsymbol{\tau}_r = \mathbf{I}_{\mathcal{B}}\boldsymbol{\alpha}_{\mathcal{B}} + [\boldsymbol{\omega}_{\mathcal{B}}]_\times \mathbf{I}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}} \tag{3.30}$$

showing that quaternion Newton-Euler system is indeed differentially flat by demonstrating (i)-(iii). In recreating the torques, the equations are tested by using the fourth numerical derivative of the positional response of the non-linear closed loop system. These naturally become very volatile when the angular states become saturated, resulting in discontinuous first derivatives. Despite this numerically induced error, the replication of torques is close to perfect despite the induced numerical errors (see Figure 3.5).

This makes the embedded motion planning an extremely powerful, as feed forward terms in torques and thrust can be computed directly from parametrisation of the four flat outputs. In addition, the theory enables a full system linearisation at any given point in time and implicitly prior verification of reachability in for instance the time varying LQR control scheme.

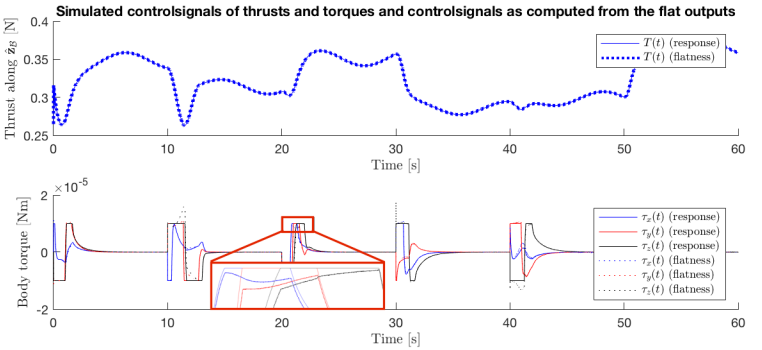

Figure 3.5 Closed PD-loop simulated control signals and control signals as computed by the flatness equations.

3.2 Generation of flat output trajectories

The travelling salesman problem (*TSP*) is an *NP*-hard *LP*-problem which revolves around finding the shortest path between a set of N points in the plane, and a method of solving this problem is a necessity if the *UAV* is to execute a set of tasks with minimal fuel consumption. The *TSP* has been the subject of much previous research, see e.g. [Padberg and Rinaldi, 1987] [Moscato and Norman, 1992], and there exist many solvers for the problem, commonly divided into exact and heuristic methods. The exact methods, such as the recursive branch-and-bound (*BnB*) described by [Balas and Toth, 1983] or the good branch-and-cut (*BnC*) methods guarantee an optimal solution with no upper bound on computational time. However, the heuristic algorithms improve the current solution on every iteration, and can therefore be terminated by some criterion yielding a close-to-optimal solution in upper bounded computational time. Recent results on heuristic *TSP* solvers and convergence criteria can be found in the work of [Noraini and Geraghty, 2011].

Problem formulation

The standard *LP*-problem formulation consists of finding the shortest closed path in the plane connecting a set of points, $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^{2\times 1} \mid i = [1, N] \in \mathbb{N}\}$, where each point is visited exactly once. The cost matrix, $\mathbf{C} \in \mathbb{R}^{N\times N}$, is defined by its entries

$$c_{i,j} = ||\mathbf{p}_i - \mathbf{p}_j||_2, \quad i, j \in [1, N] \tag{3.31}$$

and the path matrix $\mathbf{X} \in \mathbb{N}^{N \times N}$ is defined as

$$x_{i,j} = \begin{cases} 1 & \text{if the points } \mathbf{p}_i \text{ and } \mathbf{p}_j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \tag{3.32}$$

The problem is then to minimise the objective function

$$J(\mathbf{X}) = \sum_{i=1}^{N} \sum_{j=1}^{N} c_{i,j} x_{i,j}, \tag{3.33}$$

subject to constraints ensuring that each point is arrived at and left exactly once

$$\sum_{j \neq i, i=1}^{N} x_{i,j} = 1, \quad \sum_{j \neq i, j=1}^{N} x_{i,j} = 1. \tag{3.34}$$

Notably, this formulation of the N-point *TSP* can be extended to a higher dimension $P > 2$ by redefining $\mathbf{p}_i \in \mathbb{R}^P$, at the low additional cost of $(P - 2)N^2$ arithmetic operations per added dimension, as $\mathbf{C}$ is computed only once and then used as a lookup table.

The genetic algorithm

The genetic algorithm (*GA*) is based on a population of feasible solutions, a set of permutation laws, and a selection criteria. On each iteration, the population is shuffled by a set of permutation laws, and only the most fit members with regards to the selection criteria survive to the next generation. In this general case, a starting point is not needed, as all of the points are connected in a closed path by (3.34). The approach is similar to the work of [Noraini and Geraghty, 2011], but employs a different path representation, additional permutation laws and a new convergence criteria which is later extended to take point subset priorities into account.

For the N-point *TSP*, the path matrix $\mathbf{X}$ is reformulated as a path vector $\mathbf{s} = [s_1, \cdots, s_N]$ containing all integer numbers $0, ..., N-1$ with $s_0 = s_N$, thereby satisfying the feasibility conditions (3.34). In addition, the total population at iteration i is defined as $\mathcal{S}^i$ containing p_{max} path vectors, and $J(\mathbf{s})$ denotes the cost of a path vector. On each iteration, all solution vectors are extracted from $\mathcal{S}^i$ in random subsets, $\mathcal{G}$, containing four path vectors. In each subset, the most fit solution with regards to the selection criteria, $\hat{\mathbf{s}}$, is subjected to three permutation laws while the remaining paths are discarded. The best solution in each subset, along with its three permutations are then included in $\mathcal{S}^{i+1}$, the population of the next generation.

The only two conditions for the permutation laws is (i) that the definition of a path vector is not violated and (ii) that $\hat{\mathbf{s}}$ is permuted at random. Assuming that the integer numbers $a, b \in [0, N-1]$ are chosen at random

for each subset $\mathcal{G}$, we construct a total of three feasible laws. The first law, $A(\cdot)$, extracts a portion of the vector from index a to index b, reverses the segment, and inserts it at its original position. The second law $B(\cdot)$ swaps the element at index a with the one at element b. The third law $C(\cdot)$ extracts the element at index a and inserts it at index b, shifting all intermediary indices one step in the vector (see Algorithm 1)

Regardless of system requirements and computational capabilities, the monotonic cost function, J, will assume a general shape as shown in the work of [Zhang and Korf, 1996]. An initial rapid decrease of the cost is followed by a much slower convergence rate when approaching some limit point, with a clear distinction between the two regions. Consequently, we consider the best solution $\hat{\mathbf{s}}_i \subset \mathcal{S}^i$ at generation i to be close-to-optimal if

$$1 - \frac{J(\hat{\mathbf{s}}_i)}{J(\hat{\mathbf{s}}_{i_{prev}})} < \epsilon_d, \qquad i_{prev} = \lfloor \epsilon_h i \rfloor \tag{3.35}$$

where the constants $\epsilon_h \in (0, 1)$ and ϵ_d determine the solution accuracy. Using the general shape of the normalised cost function, the method holds for both small and large N, and by letting $\epsilon_h \approx 0.9, \epsilon_d \approx 0.1$, smaller problems ($N < 30$) are solved quickly, and visibly good solutions are computed for larger problems of $N > 100$.

Initialize: $\mathcal{S}^0$, $\mathbf{C}$, p_{max}, i_{max}, ϵ_d, ϵ_h
for i = 0,...,i_{max} **do**
  if $1 - J(\hat{\mathbf{s}}_i)/J(\hat{\mathbf{s}}_{i_{prev}}) < \epsilon_d$ **then**
    break
  end
  $\mathcal{S}^{i+1} = \emptyset$
  while $\mathcal{S}^i \neq \emptyset$ **do**
    randomize $\mathcal{G} \subset \mathcal{S}^i$
    $\hat{\mathbf{s}} = \min_{\mathbf{s}}(J(\mathbf{s}))\ \ \forall \mathbf{s} \subset \mathcal{G}$
    $\mathcal{S}^{i+1} = \mathcal{S}^{i+1} \cup \{\hat{\mathbf{s}}, A(\hat{\mathbf{s}}), B(\hat{\mathbf{s}}), C(\hat{\mathbf{s}})\}$
    $\mathcal{S}^i = \mathcal{S}^i \setminus \mathcal{G}$
  end
end

Algorithm 1: The proposed *GA* without priority assignment using the $A(\cdot)$, $B(\cdot)$ and $C(\cdot)$ permutations.

Priority by local permutations

A benefit of this particular *GA* formulation is that it accommodates priority assignment by local permutations through two simple modifications. Consider a total of K priority subsets of $\mathcal{P}_k \subset \mathcal{P}$, where a lower integer $k \in [1, K]$ indicates a higher priority. Here we presume to have a starting point $\mathbf{p}_s \subset \mathcal{P}$, as the direction and starting point of the path need to be known when considering the priority subsets. To be clear, this implies $N = 1 + \sum_{k=0}^{K} |\mathcal{P}_k|$ by (3.34) with $|\cdot|$ denoting set cardinality. Firstly, the initial path vector $\mathbf{s}_0$ is organised with the index of the starting point $\mathbf{p}_s$ first, followed by a sequence of indexes of all points in the priority subsets in decreasing order of priority before terminating at $\mathbf{p}_s$. Secondly, the permutation laws are set to act on a subset of points sharing priority, preserving the segments in the vector while shuffling the indices locally. Equivalent to solving many smaller *TSP*s in parallel, the formulation has the added benefit of decreasing complexity with the number of different priorities involved. In addition, rudimentary obstacle avoidance can be achieved by locating lines from point i to j that intersect the obstacle and letting $c_{i,j} = c_{j,i} = \infty$. This method can also be used to test if the problem is poorly posed, as the objective function then takes the value ∞ (see Fig. 3.6).

Complexity and performance

In the exact case, the computational complexity increases rapidly with the number of cities, N. For the simplest possible brute force method of checking all possible solutions, the worst-case complexity is obviously $O(N!)$. When adopting the implemented recursive *BnB* algorithm or the dynamical programming approach of Held and Karp described by [Goemans and Bertsimas, 1991], the worst-case time complexity is slightly better but still exponential as shown in the work of [Zhang and Korf, 1996]. The derived *GA* cannot be examined in the traditional time-complexity sense, but we may relate it to the worst-case complexity of the exact methods empirically by computing the 99% confidence interval $[i_-(N), i_+(N)]$ of the number of iterations required to converge, $i_{conv}(N)$, to a close to optimal solution by (3.35). By fitting polynomials, $p(N)$, of various degrees, a simple quadratic polynomial was found to be a satisfactory fit of the upper confidence bound (see Fig. 3.7). Indeed, the two-norm of the residual error is approximately $||\hat{i}_+(N) - p(N)||_2 \approx 60$ when fitting polynomials of orders $2, 3$, and 4. For this experiment, the population size is scaled linearly with N, leading to the total time complexity of $O(N^3)$. We stress that this is not the true worst-case time complexity, but still relatable to the exponential complexity of the two considered exact algorithms indicating that the *GA* scales better with N.

Evaluation of the *GA* implementation is done by comparison to the *BnB* solver, implemented as described by [Balas and Toth, 1983]. The performance

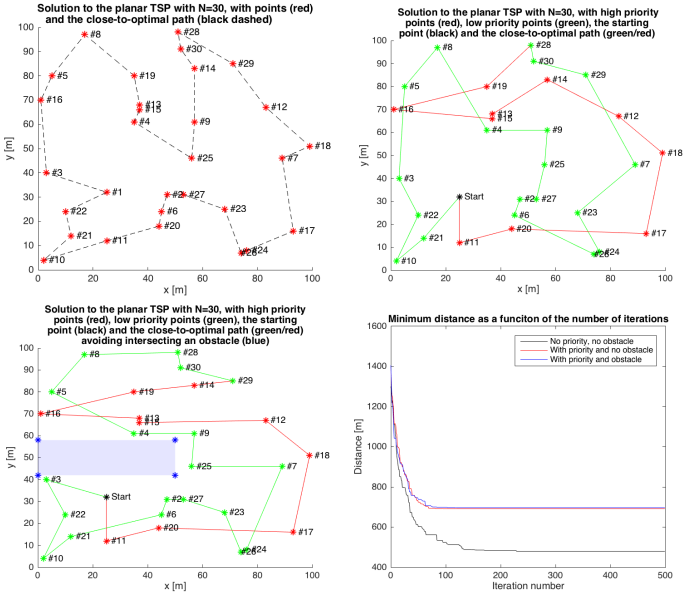

Figure 3.6 The solution the *TSP* with $N = 30$ scattered points in $x, y \in [0, 100]$. *Top left:* Close-to-optimal path for the standard problem (black dashed). *Top right:* Solution implementing point priority subsets, with the starting point (black), high priority points $\#11 - 20$ (red), priority points (green). *Bottom left:* Solution implementing point priority subsets and an obstacle (blue). *Bottom right:* The path cost as a function of iteration number with disabled convergence criterion for priority assignment (black), with priority assignment (red) and with both priority and obstacle avoidance (blue).

metrics are mean cost in meters, $\bar{J}$, and mean computational time in seconds, $\bar{t}$, examined when solving 50 random *TSP* in $\mathbb{R}^2$ with the coordinates uniformly distributed in $x, y \sim \mathcal{U}(1, 100)$ without obstacles and priorities (see Table 3.1).

The results on algorithmic complexity are clearly reflected in Table 3.1, where the implemented *BnB* algorithm quickly converges to an optimal solution for $N < 13$, slower for $N = 13$ and exceeding one minute for $N > 13$. Interestingly, the *GA* converges to the same optimal values and is faster on average for all problems $N > 7$. In addition, when $N < 40$ the computational time is on the order of 4 $[s]$, whereas the *BnB* fails to converge within 10 $[min]$. To relate the notion of computational times to larger values of N,

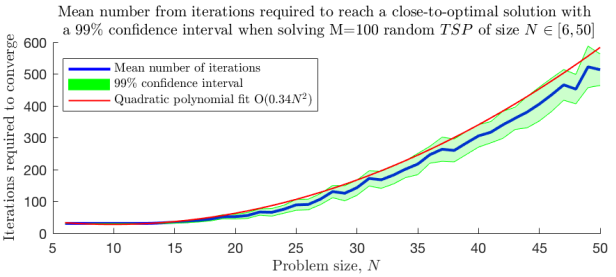

Figure 3.7 Mean value and 99% confidence interval when solving $M = 100$ N-point TSP with $N \in [6, 50]$ with a second order polynomial fitted to the upper bound of the confidence interval. In this experiment, the population size is defined as $2N + 60$ and the convergence parameters are set to $\epsilon_h = 0.9$ and $\epsilon_d = 0.1$ respectively.

Table 3.1 Mean computational time $\bar{t}$ [s] and mean cost $\bar{J}$ $[m]$ when solving 50 random N-point *TSP* using the *GA* and *BnB* solvers

N	$\bar{t}$ (GA)	$\bar{t}$ (BnB)	$\bar{J}$ (GA)	$\bar{J}$ (BnB)
7	0.0849	0.03	238	238
8	0.0913	0.11	266	266
9	0.0985	0.25	278	278
10	0.1106	0.65	292	292
11	0.1138	3.64	300	300
12	0.1324	4.17	305	305
13	0.1402	48.01	326	326
20	0.3155	-	401	-
30	1.4472	-	504	-
40	4.5259	-	563	-
50	10.9911	-	621	-

a simple test was done by computing the $N = 100$ *TSP*, converging to a close-to-optimal solution in < 2 $[min]$ with no path crossing, aligning nicely with fitted polynomial complexity theory above (see Fig. 3.7). Clearly, the algorithm can be extremely powerful in a quadcopter *UAV* application in $\mathbb{R}^3$. If the robot has a battery life of 10 minutes, and each job takes ~ 15 seconds, we would at most require the handling of $N = 40$ jobs and the *GA* could then be used to enable autonomous navigation between jobs during the entire battery lifetime.

3.3 Parametrization of flat outputs

A condition for using the powerful flatness equations is full knowledge of the flat outputs as defined in equation (3.6) and up to and including the fourth order derivatives. There are naturally a wide variety of parameterisations satisfying these conditions compatible with the output of the *TSP-GA*, and for the sake of usability we will consider Bezier and sinusoid functions, as well as polynomial splines complemented with constraints to avoid both static and dynamic obstacles.

Point-wise parametrisation with smoothing

In the simplest use case, points in flat output space generated by the *TSP-GA*, $\mathcal{P}$, is used directly with each point, $\mathbf{p}_k \in \mathcal{P}$, having an assigned time $T_k \in \mathbf{T}$ during which it is used as a reference point. This discrete parametrisation encompasses the sending of single positional set points, where then $T_k \to \infty$, but the resulting trajectory is discontinuous and obviously incompatible with the system dynamics. The first order derivative is an impulse and the flatness equations will fail as a consequence.

In order to make the discrete reference trajectory $\boldsymbol{\gamma}(t)$ compliant with system dynamics and enable use of the flatness equations, we let

$$\boldsymbol{\Gamma}(s) = \big[\mathbf{X}(s), \mathbf{Y}(s), \mathbf{Z}(s), \boldsymbol{\Psi}(s)\big]^T = \big[\mathcal{L}\{x(t)\}_s, \mathcal{L}\{y(t)\}_s, \mathcal{L}\{z(t)\}_s, \mathcal{L}\{\psi(t)\}_s\big]^T \tag{3.36}$$

denote the Laplace domain equivalent of the discrete reference trajectory. Applying a lowpass filter of order n with unit static gain

$$G(s) = \frac{a^n}{(s+a)^n} \tag{3.37}$$

to each element in $\boldsymbol{\Gamma}(s)$ results in a smoothened curve, $\hat{\boldsymbol{\Gamma}}(s) = G(s)\boldsymbol{\Gamma}(s)$, which has two notable properties. The first observation is that for a one dimensional unit step $\theta(t)$, $\mathcal{L}\{\alpha\theta(t)\} = \alpha\mathcal{L}\{\theta(t)\}$, implying that the positional part of the path $\hat{\boldsymbol{\Gamma}}(s)$ will always be linear in $\mathbb{R}^3$. The second property is that any flat output derivative of order $\leq n$ is finite and can be determined by

$$\mathcal{L}\left\{\frac{\partial x(t)}{\partial t}\right\}_s = s\mathbf{X}(s) - x(0). \tag{3.38}$$

Consequently, we form a smoothing single-input multiple-output *SIMO* filter $\mathbf{F}(s)$ and assume that at the time of commanding a unit step, $x(0) = x'(0) = x''(0) = x^{(3)}(0) = 0$. Discretisation with Tustin's approximation at a time

step of h [s], yielding the pulse transfer function

$$\mathbf{F}(s) = \frac{a^n}{(s+a)^n} \begin{bmatrix} s^4 \\ s^3 \\ s^2 \\ s \\ 1 \end{bmatrix} \Rightarrow \mathbf{H}(z) = \frac{1}{A(z)} \begin{bmatrix} B_4(z) \\ B_3(z) \\ B_2(z) \\ B_1(z) \\ B_0(z) \end{bmatrix} \tag{3.39}$$

Implementing the difference equation with $n = 5, a = 7.5, h = 0.02$ was shown to preserve numerical stability and generate discrete time references

$$\begin{bmatrix} x_s^{(4)}(hk) & x_s^{(3)}(hk) & \ddot{x}_s(hk) & \dot{x}_s(hk) & x_s(hk) \end{bmatrix}^T = \mathbf{H}(z)x(hk) \tag{3.40}$$

such that the magnitude three dimensional acceleration

$$||\mathbf{a}(hk)||_2 = \sqrt{\ddot{x}_s^2(hk) + \ddot{y}_s^2(hk) + \ddot{z}_s^2(hk)} < 10 \ \ [m/s^2] \tag{3.41}$$

for reference changes within the unit cube, comparable to the physical system constraints in maximum possible thrust. This method is applied to each flat output dimension at a low computational cost and allows the flatness equations to be applied safely to the discrete point sequence (see 3.8).

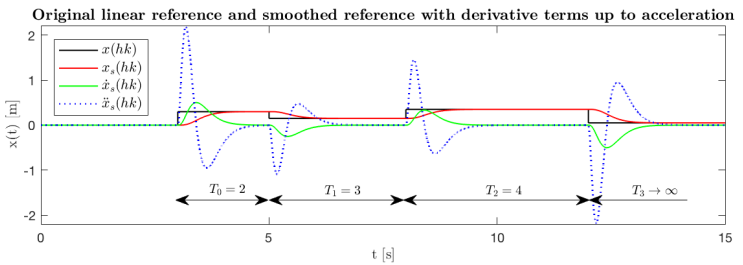

Figure 3.8 Response of the discrete time *SIMO* filter with $\{n = 5, a = 7.5, h = 0.02\}$, here applied to a typical one dimensional discontinuous trajectory including the first two smoothed derivative terms.

Cubic bezier parameterisation

The quadratic Bezier curve with arbitrary dimension is defined by a starting point, an intermediate control point and a terminal and terminal point $b_0, b_1, b_2 \in \mathbb{R}$, with the corresponding quadratic curve

$$B_{b_0,b_1,b_2}(t) = (1-t)[(1-t)b_0 + tb_1] + t[(1-t)b_1 + tb_2], \quad t \in [0,1]. \tag{3.42}$$

However, the more standard method of parameterisations such as the output of Blender is the cubic Bezier curve [Blender Online Community, 2016], which then includes two control points,

$$B_{b_0,b_1,b_2,b_3}(t) = (1-t)B_{b_0,b_1,b_2}(t) + tB_{b_1,b_2,b_3}(t), \quad t \in [0,1]. \tag{3.43}$$

Clearly, in this format the k^{th} spline is defined by a set of points $\mathcal{B}_k = \{b_{k,0}, b_{k,1}, b_{k,2}, b_{k,3}\} \in \mathbb{R}^4$ and a time interval $[0, T_k] \in \mathbb{R}^+$ during which the spline should be followed. Note that with the Bezier curve, the fourth derivative is zero at all times, indicating that we can't make full use of the flatness implementation as the fourth derivative used in computing reference torques will be zero at all times. An analogy can be made to restricting the use of the flatness by only providing a non-zero acceleration, which results in traditional positional feed-forward terms and no additional knowledge on the angular rate, angular acceleration or torque feed forward terms.

Sinusodial parameterisation

The sinusoidal parametrisation on the other hand can assume non-zero values in the fourth derivative. Here we consider curves with a linear amplitude in t, where the k^{th} spline takes the form

$$(A_k + B_k t)\sin(\omega_k(t_{0,k} + t) + \varphi_k) + C_k, \quad t \in [0, T_k] \tag{3.44}$$

for some constants $\mathcal{S}_k = \{A_k, B_k, C_k, t_{0,k}, \omega_k, \varphi_k\} \in \mathbb{R}^6$ defined on a time span $T_k \in \mathbb{R}^+$ (see 3.9).

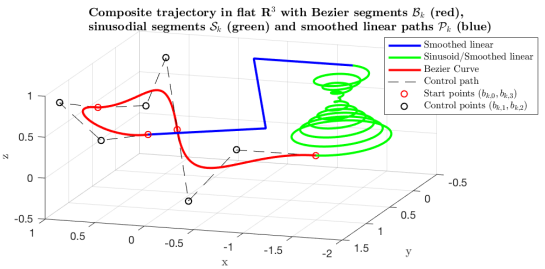

Figure 3.9 Example of a flat output trajectory containing cubic Bezier curves with the corresponding control points, sinusoid parameterisations and smoothed linear segments.

Constrained polynomial parameterisation

For the polynomial splines, we consider the method presented by [Richter et al., 2013] where a one dimensional trajectory segment composed of n

polynomials $P_1(t), ..., P_n(t)$,

$$P_k(t) = \sum_{i=0}^{N} p_{k,i} t^i, \qquad t \in [0, T_k] \tag{3.45}$$

with a maximum degree of $\deg(P_k) = N$, and a corresponding coefficient vector $\mathbf{p}_{(k)} = [p_{k,0}, ..., p_{k,N}]^T$. The problem is to minimise a cost function for such every polynomial spline

$$J(T_k) = \int_0^{T_k} c_0 P_k(t)^2 + c_1 P_k'(t)^2 + ... + c_N P_k^{(N)}(t)^2 dt = \mathbf{p}_{(k)}^T \mathbf{Q}_{(k)} \mathbf{p}_k \tag{3.46}$$

preserving continuity and enforcing boundary conditions. The complete constrained QP-formulation, including all n polynomials is then

$$\text{Minimize} \Big(\sum_{k=1}^{n} J(T_k) \Big) \qquad \text{subject to} \qquad \mathbf{A}\mathbf{p} - \mathbf{b} = 0 \tag{3.47}$$

where

$$\tag{3.48}$$

$$\sum_{k=1}^{n} J(T_k) = \begin{bmatrix} \mathbf{p}_{(1)} & \cdots & \mathbf{p}_{(n)} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_{(1)} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \mathbf{Q}_{(n)} \end{bmatrix} \begin{bmatrix} \mathbf{p}_{(1)} \\ \vdots \\ \mathbf{p}_{(n)} \end{bmatrix} = \mathbf{p}^T \mathbf{Q} \mathbf{p}. \tag{3.49}$$

Boundary conditions can be enforced for each spline by finding a matrix

$$\mathbf{A}_{(k)} \mathbf{p}_{(k)} = \mathbf{b}_{(k)} \tag{3.50}$$

for every know derivative at the time $t = 0$ (collected in $\mathbf{A}_0$) and time $t = T_k$ (collected in $\mathbf{A}_T$). With n_c boundary conditions for the k^{th} spline, then

$$\mathbf{A}_{(k)} = \begin{bmatrix} \mathbf{A}_{0,k} \\ \mathbf{A}_{T,k} \end{bmatrix} \in \mathbb{R}^{n_c \times N+1} \quad \text{and} \quad \mathbf{b}_{(k)} = \begin{bmatrix} \mathbf{b}_{0,k} \\ \mathbf{b}_{T,k} \end{bmatrix} \in \mathbb{R}^{n_c \times 1} \tag{3.51}$$

For the remaining, free boundary endpoints where no fixed derivative is specified, the splines on each side of a boundary point are set equal by enforcing

$$\mathbf{A}_{T,k} \mathbf{p}_k - \mathbf{A}_{0,k+1} \mathbf{p}_{k+1} = 0. \tag{3.52}$$

Dynamic obstacle avoidance

A powerful addition to the motion planning is to include knowledge about dynamical objects in the environment to update the trajectory in real-time without re-generating the trajectory. Consider a non-rotating obstacle, $\mathcal{S} \in \mathbb{R}^3$, restricted to a subset of space on $t \in [0, T_{tot}]$, where $T_{tot} = \sum_{k=1}^{n} T_k$, the

total time interval on which the generated trajectory is defined. En example of such a non-rotating object is a pendulum swinging in a two dimensional plane.

Now, if the obstacle stands still, any feasible positional trajectory $\mathbf{p}(t)$ may clearly come close but never intersect $\mathcal{S}$. If instead $\mathcal{S}$ follows an unknown trajectory $\mathbf{f}(t) \in \mathbb{R}^3$ with $\mathbf{f}(0) = 0$, any trajectory which is feasible at the time of generation may collide with the obstacle. However, if know a set of N points, $\mathcal{C} = \{\mathbf{c}_0, ..., \mathbf{c}_N\} \in \mathbb{R}^3$, on the reference trajectory which come close to a known obstacle at the time of generation, then we may estimate the trajectory of the object and shift the already generated trajectory accordingly. Let

$$d(t) = \min_{\mathbf{c} \in \mathcal{C}}(||\mathbf{p}_r(t) - \mathbf{c}||_2) \tag{3.53}$$

If we define a weight function with $w'(0) = w'(1) = w(1) = 0$, $w(0) = 1$, such that

$$w(x) = \begin{cases} 2x^3 - 3x^2 + 1 & \text{if } x \leq 1 \\ 0 & \text{if } x > 1 \text{ or } \mathbf{p}_r(t) \in \text{int}(\mathcal{S}_i) \end{cases} \tag{3.54}$$

The updated positional trajectory, $\hat{\mathbf{p}}_r(t)$, is simply

$$\hat{\mathbf{p}}_r(t) = \Big[1 - w\Big(\frac{d(t)}{d_{max}}\Big)\Big]\mathbf{p}_r(t) + w\Big(\frac{d(t)}{d_{max}}\Big)\mathbf{f}(t). \tag{3.55}$$

This method is simple, highly situational and performs very poorly if we have many obstacles, if these are very volatile and if the obstacles move far away from the N spheres with a radius d_{max} centred around the points in $\mathcal{C}$. However, it can be useful if demonstrating flight through a thrown hoop or a swinging pendulum. To demonstrate this, a simulation was done for a trajectory consisting of six splines in $\mathbb{R}^3$, passing through a hollow obstacle moving along an initially unknown, but later estimated trajectory $\mathbf{f}(t)$ (see Fig. 3.10).

3.4 Summary

In this section, the theory behind the implemented embedded sequence generator was presented. Firstly, the property of differential flatness was shown for the quaternion Newton-Euler dynamics, suggesting a parameterisation of position and yaw in order to generate feed forward terms in body rates and torques. Secondly, a genetic algorithm was proposed to mutate a feasible path into a close-to-optimal solution detailing the shortest path between a set of points considering obstacles and varying priority across the points. Thirdly, several methods of parametrising the trajectories in flat output space

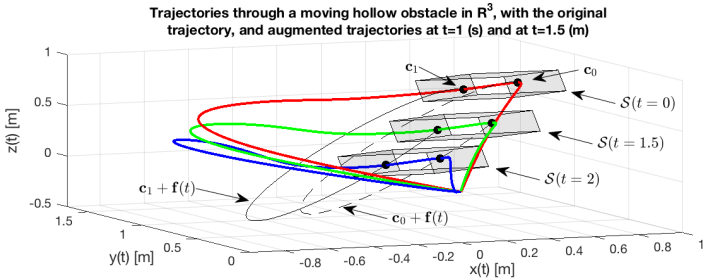

Figure 3.10 Polynomial minimum snap trajectories in $\mathbb{R}^3$ passing through a hollow, non-rotating obstacle $\mathcal{S}$ moving along a periodical trajectory $\mathbf{c}+\mathbf{f}(t)$ at the time $t=0$ (red), $t=1$ (blue) and $t=1.5$ (green).

were presented and implemented in the embedded system, supporting preloading for completely autonomous flight. Combined, the proposed method of motion-planning satisfies the specifications

(i) Compliance with the rigid-body dynamics.

(ii) Being economical in terms of power usage.

(iii) Avoiding paths which intersect known static obstacles.

(iv) Avoiding dynamic obstacles.

(v) Loading and on-line evaluation of precomputed trajectories represented with minimal information.

(vi) Compliance with current academic- and industry standards with .

(vii) Synchronising trajectory evaluation across multiple *UAV*'s.

(viii) Scheduling events.

with (i) and (v) discussed in the first section on differential flatness. The specifications (ii) and (iii) were discussed in the second section on the *TSP-GA*, the specifications (vi) was handled through various means of trajectory parametrisation and the reader is referred to [Greiff, 2017] for the implementation of the communication and details on the methods of synchronising trajectories and scheduling events pertaining to specifications (vii) and (viii).

4

Rotor control

In this section, we consider the direct open loop rotor control which was implemented in the embedded *UAV* firmware, discussing possible improvements. The brushless DC motors are by convention controlled through pulse width modulation (*PWM*), where duty cycles are set to control the rotor speeds and implicitly thrusts. This can be done by simply applying a series of maps or functions identified in the system identification, generating *PWM* necessary to achieve the thrust and torques in the rigid body control (see **Section A.1**). However, we also outline a possible improvement where rotors are run with current feedback in a separate micro-controller using on-line parameter identification to increase system robustness. In addition to making the power distribution more robust in the face of time varying parameters, the considered change would also migrate some of the computational effort from the current central processing unit, enabling higher sampling rates across the control system for increased performance.

4.1 Open loop control

The simplest conceivable method of rotor control is to use the affine maps from *PWM* duty cycle to rotor thrust, $\mathbf{M}_{d\to T} \in \mathbb{R}^{4\times 4}$ and rotor speeds squared to rotor thrust, $\mathbf{M}_{\Omega^2\to T} \in \mathbb{R}^{4\times 4}$, defined in **Section A.1**. In addition, knowing the map from rotor speeds squared to total thrust and body torques, $\mathbf{M}_{\Omega^2\to\boldsymbol{\tau}} \in \mathbb{R}^{4\times 4}$, as given in (2.11), the desired *PWM* duty cycles $\mathbf{d}_r(t) = \begin{bmatrix} d_1(t) & d_2(t) & d_3(t) & d_4(t) \end{bmatrix}^T \in [0,1]^4$ are easily computed from references in total thrust, $T_r(t)$ $[N]$, and desired rigid body torques, $\boldsymbol{\tau}_r(t)$ $[Nm]$,

$$\mathbf{d}_r(t) = \mathbf{M}_{d\to T}^{-1}\Big(\mathbf{M}_{\Omega^2\to T}\Big(\mathbf{M}_{\Omega^2\to\boldsymbol{\tau}}^{-1}\Big(\begin{bmatrix} T_r(t) & \boldsymbol{\tau}_r(t) \end{bmatrix}^T\Big)\Big)\Big). \tag{4.1}$$

While functional and currently used in the real-time implementation, the described method of rotor power distribution assumes time invariance in the

affine maps, which is not the case with parameters such as the PWM to thrust ratio, k, which depends greatly on the battery charge. Operating under the assumption of time invariance, increased robustness of the controllers for attitude and translation is required in order to support aggressive manoeuvres throughout battery lifetime.

4.2 Closed loop rotor control

As an alternate approach for future hardware revisions, an additional microcontroller could be added to the Crazyflie for the sole purpose of high rate rotor control. Such an implementation could also include two H-bridges, allowing the rotors to run in reverse for even more aggressive manoeuvres, which is currently impossible due to the definition of rotor signs in $\mathbf{M}_{\Omega^2 \to \boldsymbol{\tau}} \in \mathbb{R}^{4\times 4}$. Having presented the current method of rotor control, we will in this section investigate the idea of closed loop rotor control in theory detailing promising controllers and parameter estimators in discrete time.

For this discussion, we recall the *SISO* rotor from **Section 2.3**, with a control signal voltage $U_i(t)$ $[V]$, and state current $I_i(t)$ $[A]$ and angular velocity $\Omega_i(t)$ $[rad]$, disregarding the rotor position. In the Laplace domain, we let $U_i(s) = \mathcal{L}\{U_i(t)\}_s$ with similar notations for $I_i(s)$ and $\Omega_i(s)$. By the relationship of the *SISO* motor system from $U_i(s)$ to $\Omega_i(s)$,

$$\begin{bmatrix} G^{(i)}_{U\to\Omega}(s) \\ G^{(i)}_{U\to I}(s) \end{bmatrix} = \begin{bmatrix} \frac{K_t}{LJ^{\pm}} \\ \frac{1}{L}s + \frac{b}{LJ^{\pm}} \end{bmatrix} \frac{1}{s^2 + \left(\frac{b}{J^{\pm}} + \frac{R}{L}\right)s + \left(\frac{bR + K_t K_e}{LJ^{\pm}}\right)} \tag{4.2}$$

nterestingly, the two transfer functions share the same characteristic polynomial, and contains the factor $K_t/LJ^{\pm}$ in the numerator. If we assume knowledge of time invariant parameters of resistance R $[Ohm]$ and inductance L $[H]$, we may capture the variance in the model as changes in inertia $J\pm$, viscous friction b and constant K_t by estimating the parameters in the transfer function from $U_i(s)$ to $I_i(s)$ alone, allowing the complete identification of the coefficients in the voltage to rotor speed dynamics. As such, we will in this section consider controllers and methods of parameter and state estimation for each *SISO* system independently.

PID control

As a first attempt, the *PID* controller is implemented in parallel form due to its simplicity, described in detail in **Appendix B.1**. It uses forward difference discretisation for the I-part and backward differences for the D-part. All controller terms are included, anti-windup is used with control signal saturations u_{min} and u_{max}, and set-point weighting is included through two scalar parameters α and β. The *PID* control, while not being restricted to

linear systems, might require gain scheduling depending on how much the rotor dynamics change in time. For the purposes of this example, the controller was tuned to satisfy the specifications based on the identified rotor model (see Table A.2), resulting in the parameters in Table 4.1.

Table 4.1 Discrete time *PID* parameters used in the simulations.

Parameter	K	T_i	T_d	N	γ	β	h	u_{min}	u_{max}
Value	0.0015	0.1	0.025	100	0.4	0.6	0.002	500	4500

MRAC control

An alternative to the *PID* controller is to use model reference adaptive control (*MRAC*) in which each *SISO* rotor system is made to follow a reference model $G_m(s)$ by adaption using two scalar control gains, Γ_u and Γ_y. In order to synthesise the controller using the Lyapunov rule, the dynamics must be sufficiently well modelled and the function $G^{(i)}_{U\to\Omega}(s)$ needs to be strictly positive real (*SPR*) for the Kalman-Yokovic-Popov lemma to apply [Iwasaki and Hara, 2005]. However, the condition

$$\lim_{\omega\to\infty} \omega^2 \Re[G^{(i)}_{U\to\Omega}(i\omega)] > 0 \tag{4.3}$$

results in a contradiction as $n\omega^2(d_2 - \omega^2) < 0$ when $|\omega| > \sqrt{d_2} = 7.8$ with the identified parameters (see Table A.2). We conclude that the rotor system considered is not *SPR*, disqualifying it from controller synthesis using the Lyapunov rule. Instead, the controller is derived using the *MIT*-rule (see derivation in **Appendix B.2**), where the reference model is defined as a continuous time second order Hurwitz system

$$G_m(s) = \frac{B_m(s)}{A_m(s)} = \frac{\alpha\omega_m^2}{s^2 + 2\xi_m\omega_m s + \omega_m^2} = \frac{\alpha\omega_m^2}{s^2 + 2\xi_m\omega_m s + a_0^m}, \tag{4.4}$$

initially with $\omega_m = 10$ and $\xi_m = 0.707$ conforming to industry standard damping and the speed of the rotor system. In addition, the parameter α is included such that the static gain of the reference model matches the $G_m(0) \approx \alpha \approx n/d_2$. Similarly to the *PID* controller, the *MRAC* was tuned to satisfy the specifications when run on the identified rotor model (see Table A.2), resulting in the parameters in Table 4.2.

Error metric

In order to test performance, we define the error as integrated mean squared error (*MSE*) of two time varying vectors $\mathbf{r}(t), \mathbf{x}(t) \in \mathbb{R}^{N\times 1}$ defined on $t \in$

Table 4.2 Continuous time *MRAC* parameters used in the simulations.

Parameter	Γ_u	Γ_y	ω_m	ξ_m	α
Value	$1.8 \cdot 10^{-8}$	$1.6 \cdot 10^{-8}$	16.4	0.707	1780

$[t_i, t_f]$ using the l^2-norm $||\cdot||_2$,

$$E_p(\mathbf{r}(t), \mathbf{x}(t)) = \int_{t_i}^{t_f} (||\mathbf{r}(t) - \mathbf{x}(t)||_2)^{1/p} dt. \tag{4.5}$$

For the purposes of evaluate the rotor control and parameter estimation, we let $p = 1$ for future reference. All comparison will be done with respect to the available options, such that if we have a total of N tested controller responses $\mathbf{x}_i(t), \quad i \in [1, N]$ which are supposed to follow a reference $\mathbf{r}(t)$, then the performance of controller k will be measured by

$$E_p^k(\mathbf{r}, \mathbf{x}_k) = \frac{E_p(\mathbf{r}, \mathbf{x}_k)}{\min\limits_{i \in [1,N]} \Big(E_p(\mathbf{r}, \mathbf{x}_i) \Big)}. \tag{4.6}$$

In this way, the controller minimising the *MSE* to a power of p will yield $E_p^k(\mathbf{r}, \mathbf{x}_k) = 1$ while all other controllers will indicate how many magnitudes worse the reference was followed with respect to the integrated *MSE*.

Simulation study and comparison

For the purpose of simulating and comparing rotor control, we assume the thrust tor rotor speed squared ratio $k(t) \equiv 2.2 \cdot 10^{-8}$. In addition, knowing that the Crazyflie weighs $m \approx 0.027$ $[kg]$, the angular speed required to hover can be computed as

$$\Omega_h(t) \approx \sqrt{\frac{m \cdot g}{4 \cdot k(t)}} \equiv \sqrt{\frac{0.027 \cdot 9.81}{4 \cdot 2.2 \cdot 10^{-8}}} \approx 1750 \quad [rad/s] \tag{4.7}$$

In evaluating the rotor control, we consider controllers that operate well in the interval $\mathbf{\Omega} \in [1000, 2500]^4$ $[rad/s]$ and can start from a resting state, at $\Omega(t_0) = U(t_0) = i(t_0) = 0$. In addition, we assume that the angular rate measurements are corrupted by zero mean white gaussian noise $v_k \sim \mathcal{N}(0, 15)$ $[rad/s]$ and perform the simulation tests at a sample rate of 500 $[Hz]$ (see Figure 4.1). The first simulation (1) computes the error between the rotor speed, $\mathbf{\Omega}(t)$, and reference, $\mathbf{\Omega}_r(t)$ staring from rest at $\mathbf{\Omega}(t_0) = \mathbf{0}$ subject to a step with an amplitude of $\Omega_{max} = ||\mathbf{\Omega}(t)||_\infty = 2500$ $[rad/s]$. In both the *MRAC* and the *PID*, the rotor very quickly reaches the reference, but the *MRAC* is slightly slow on account of the adaptive gains starting

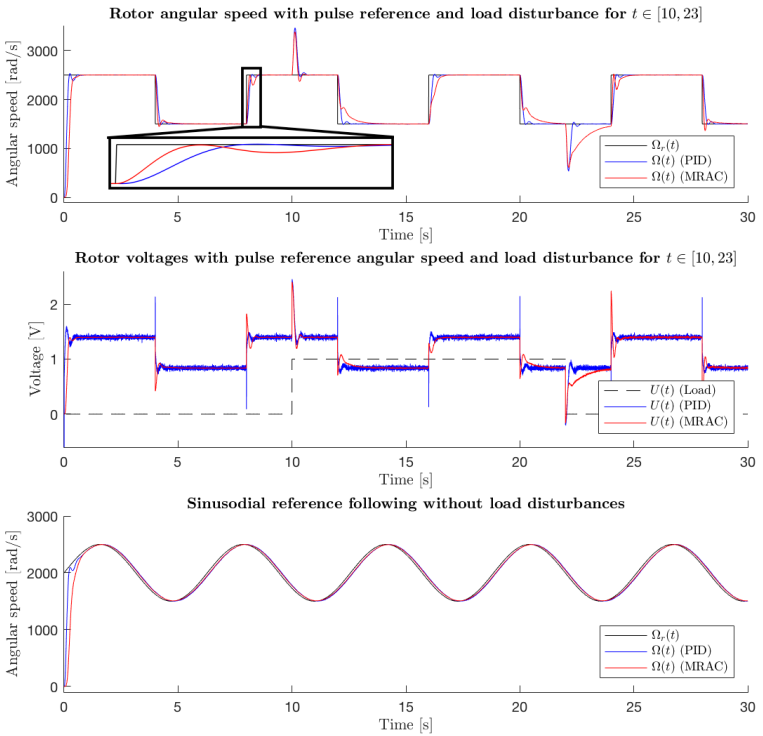

Figure 4.1 *Top*: Simulated performance of the two rotor controllers *PID* (blue) and *MRAC* (red) when running a an aggressive pulse reference (black). Here, $t \in [0, 4]$ corresponds to test (1), test (2) is run with the constant load disturbance at $t \in [10, 23]$ and test (4) is run without load disturbance and starting the rotor at $\omega(0) = 1750$. *Middle*: The control signals corresponding to the response in the top plot, showing the load disturbance (black, dashed). *Bottom*: Following a smooth sinusoidal trajectory corresponding to test (3).

at $\theta_u(t_0) = \theta_y(t_0) = 0$ (see Table 4.3). The second test (2) compares the controllers' disturbance rejecting properties by giving a very aggressive pulse reference rotor speed, alternating between 1750 ± 750 at a period of 8 $[s]$, then applying a load disturbance starting at $t = 10$ and ending at $t = 23$ $[s]$. Here we let $\Omega(t_0) = \Omega_{max}$ as we are not interested in capturing the initial rise time in the error norm computation. The purposes of this test is to see how the closed loop system rotors reacts when, for instance, entangling a strand of hair in a rotor or loosing part of a propeller. Both of which imply a sudden shift in the input signal required to retain the same reference rotor speed. Demonstrably, the *PID* controller, as expected, handles the disturbances very well, while the *MRAC* controller struggles to handle the constant offset. Both converge, to the reference, but convergence is much faster in the *PID* controller. In the third simulation (3) the controllers are set to follow a smooth sinusoidal reference trajectory, alternating between Ω_{min} and Ω_{max} at a rate of 1 [rad/s]. Here we assume that the rotor starts at $\mathbf{\Omega}(t_0) = 1750$ [rad/s] to avoid influencing the error norm with the initial rise time difference. The controllers are comparable, performing very well and only introducing a slight phase lag between the intended trajectory and the actual response. When in (4) performing a similar experiment with a pulse reference rotor speed alternating between 1700 ± 800 at a period of 8 $[s]$, we again see that the *PID* outperforms the MRAC, but only by a slim margin (see Table 4.3).

Table 4.3 Relative error norm in controller response during simulation tests.

Test	$E_{p=1}^{PID}(\mathbf{\Omega}_r, \mathbf{\Omega})$	$E_{p=1}^{MRAC}(\mathbf{\Omega}_r, \mathbf{\Omega})$
(1) rise	1	2.24
(2) load	1	2.78
(3) smooth	1.13	1
(4) aggressive	1	1.36

In conclusion, both the *MRAC* and the *PID* can be used, but when compared against each other in the relevant specifications the *PID* outperformed the *MRAC* slightly. In addition, the *PID* controller is easier to implement in real time and should therefore be preferred above the *MRAC*. The only reason for choose the *MRAC* is to include the non-linear rotor dynamics in the model used in the outer control loop, in which case we could simply approximate the closed loop *SISO* rotor by the reference model $G_m(s)$ and simply assume that any un-modelled non-linearities arising from the coupling of the rotors to the quadcopter dynamics are handled by the adaptive gains.

4.3 Rotor adaptation and estimation

Due to simplifications made in the quadcopter dynamics to attain computational feasibility in the state estimators and controllers, approximations such as the effects of velocities along the $\hat{\mathbf{z}}_{\mathcal{B}}$ direction will affect the rotor dynamics are disregarded completely. To enable robust on-line parameter estimation of the characteristic polynomial of $G_{I\to\Omega}$ in order to determine $G_{U\to\Omega}$, two variations of the least squares algorithms are considered. As time variation is expected, it is essential to include a forgetting factor, λ, in the algorithm formulation. For this discussion, we let

$$H^{(i)}_{U\to I}(z) = \mathcal{Z}\{G^{(i)}_{U\to I}(s)\} = \frac{B(z)}{A(z)} \quad \text{with} \quad B(z) = \sum_{i=0}^{n_b} b_i z^i, \;\; A(z) = \sum_{i=0}^{n_a} a_i z^i, \tag{4.8}$$

be the zero-order hold *ZOH* discrete time equivalent of the continuous time system, such that $\deg(B(z)) = 1$ and the polynomial $A(z)$ is of a degree $\deg(A(z)) = 2$ and monic. Furthermore, we let denote the current and amplitude at time $t = hk$ by U_k and I_k respectively. This allows us to express the system as

$$I_k = \boldsymbol{\varphi}_k^T \boldsymbol{\theta} + e_k \tag{4.9}$$

where $\mathbf{e}_k$ is white gaussian noise, if defining the regressor vector

$$\boldsymbol{\varphi}_k = \begin{bmatrix} -I_{k-1} & -I_{k-2} & U_{k-1} & U_{k-2} \end{bmatrix}^T \in \mathbb{R}^{4\times 1}, \tag{4.10}$$

and the corresponding parameter vector and parameter estimate at time k

$$\boldsymbol{\theta} = \begin{bmatrix} a_1 & a_0 & b_1 & b_0 \end{bmatrix}^T \in \mathbb{R}^{4\times 1}, \;\; \hat{\boldsymbol{\theta}}_k = \begin{bmatrix} \hat{a}_{1,k} & \hat{a}_{0,k} & \hat{b}_{1,k} & \hat{b}_{0,k} \end{bmatrix}^T \in \mathbb{R}^{4\times 1}. \tag{4.11}$$

Using a forgetting factor λ and an initial covariance matrix $\mathbf{P}_0 = \delta \mathbf{I}_{4\times 4}$ for some $\delta \in \mathbb{R}^+$, given a new current measurement I_k, the standard discrete time *RLS* algorithm is defined by the equations

$$\begin{aligned} \boldsymbol{\epsilon}_k &= I_k - \boldsymbol{\varphi}_k^T \hat{\boldsymbol{\theta}}_{k-1} \\ \mathbf{P}_k &= \frac{1}{\lambda}\Big(\mathbf{P}_{k-1} - \frac{\mathbf{P}_{k-1}\boldsymbol{\varphi}_k \boldsymbol{\varphi}_k^T \mathbf{P}_{k-1}}{1 + \boldsymbol{\varphi}_k^T \mathbf{P}_{k-1} \boldsymbol{\varphi}_k}\Big) \\ \hat{\boldsymbol{\theta}}_k &= \hat{\boldsymbol{\theta}}_{k-1} + \boldsymbol{\epsilon}_k \mathbf{P}_k \hat{\boldsymbol{\theta}}_{k-1} \end{aligned} \tag{4.12}$$

It is, however, well known that introducing the forgetting factor causes the norm of the covariance matrix $\mathbf{P}_k$ to increase exponentially with time if the signal is not sufficiently exciting, referred to as estimator wind-up. In the quadcopter dynamics, such an event may occur if a stable hovering position is kept for some time in which case the system dynamics hopefully remain

constant and any estimated parameter (with the exception of k) would likely be fix. To combat estimator wind-up, we could consider using a covariance reset at certain intervals in time, but a slightly more refined approach would be to use a regularized constant-trace *RLS* algorithm which enforces a constant covariance matrix trace, $\text{tr}(\mathbf{P}_k)$. The algorithm is widely used and can be written

$$\begin{aligned}
\boldsymbol{\epsilon}_k &= I_k - \boldsymbol{\varphi}_k^T \hat{\boldsymbol{\theta}}_{k-1} \\
\mathbf{K}_k &= \frac{\mathbf{P}_{k-1}\boldsymbol{\varphi}_k}{\boldsymbol{\lambda} + \boldsymbol{\varphi}_k^T \mathbf{P}_{k-1}\boldsymbol{\varphi}_k} \\
\bar{\mathbf{P}}_k &= \frac{1}{\lambda}\Big(\mathbf{P}_{k-1} - \frac{\mathbf{P}_{k-1}\boldsymbol{\varphi}_k\boldsymbol{\varphi}_k^T\mathbf{P}_{k-1}}{1 + \boldsymbol{\varphi}_k^T\mathbf{P}_{k-1}\boldsymbol{\varphi}_k}\Big) \\
\mathbf{P}_k &= c_1 \frac{\bar{\mathbf{P}}_k}{tr(\bar{\mathbf{P}}_k)} + c_2 \mathbf{I}_{4\times 4} \\
\hat{\boldsymbol{\theta}}_k &= \hat{\boldsymbol{\theta}}_{k-1} + \boldsymbol{\epsilon}_k \mathbf{K}_k \hat{\boldsymbol{\theta}}_{k-1}.
\end{aligned} \tag{4.13}$$

Having estimated the parameters of $H^{(i)}_{U\to I}(z)$, we may use the inverse discretisation to determine the time varying coefficients J and b of the transfer function $G^{(i)}_{U\to I}(s)$, corresponding to d_1, which may then be used in the previously derived *MRAC* controller and in estimating the rotor speed with model based estimators.

Simulation study and comparison

To demonstrate the implementation and the differences in the RLS algorithms, the two estimators (4.12) and (4.13) were applied to the voltage-current rotor dynamics (see Table A.2), is discretised at $h = 0.002$ $[s]$ using *ZOH*, resulting in the nominal parameter values $a_1 \approx -1.95$, $a_0 \approx 0.97$, $b_1 \approx 0.02$, $b_0 \approx -0.02$. It should be noted that the poles of this system are very close to the unit circle, as the characteristic polynomial has roots in $|z_{1,2}| \approx 0.98$. The pulse transfer function is initialised with the nominal parameter values, and the b_i parameters are varied in time using a sinusoidal perturbations with amplitudes of 0.01. The parameters were initialised with the values in Table 4.4, and the system was simulated with a persistently exciting sinusoidal voltage in order for the standard *RLS* algorithm to converge.

Two simulations were run, the first with an initial estimate of $\hat{\boldsymbol{\theta}}_0 = \mathbf{0}$ for $t \in [0, 5]$to compare initial parameter convergence, and the second simulation with $\hat{\boldsymbol{\theta}}_0 = \boldsymbol{\theta}_0$ for $t \in [0, 60]$ to compare transient parameter estimation. In both considered tests, the constant trace formulation far outperformed the standard *RLS* estimator in the integrated *MSE* error metric (see Table 4.5).

The parameter estimations show that the two-norm of the covariance matrix standard *RLS* algorithm varies greatly in time, and that it demon-

Table 4.4 Parameters used in simulating the RLS estimators.

Algorithm	δ	λ	c_1	c_2
Standard RLS	10^{-4}	0.97	$\sim$	$\sim$
Constant trace RLS	10^{-1}	0.95	10^{5}	10^{-2}

Table 4.5 Relative error in the parameter estimation during simulation tests.

Test	$E_{p=1}^{RLS}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$	$E_{p=1}^{ctRLS}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$
(1) initial convergence	14.79	1
(2) transient estimate	5.65	1

strates an exponential growth when introducing the forgetting factor (see Figure 4.2).

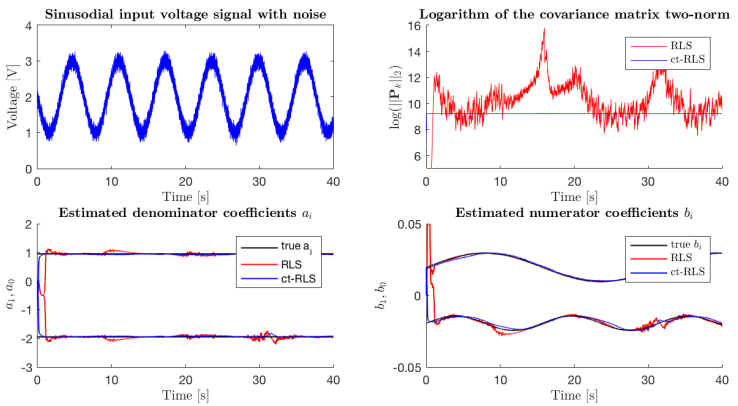

Figure 4.2 Example of parameter identification with *RLS* and *ct-RLS* algorithms showing the initial convergence behaviour when $\hat{\boldsymbol{\theta}}_0 = \mathbf{0}$ and the transient parameter estimation.

A tradeoff has to be made between fast estimation and stability. As can be seen in this example, $\lambda = 0.97$ enables good parameter estimation at the cost of coming close to diverging at $t \approx 16$ $[s]$ with the standard *RLS* algorithm. If we expect to hover in a stable position, the reference signals and system dynamics will be close to constant, potentially resulting in a rapidly growing covariance matrix and total instability unless we introduce some means of

covariance resetting in the *RLS* formulation. The *ct-RLS* algorithm on the other hand provides an upper bound on the l^2-norm of the covariance matrix and will not diverge in the event of a constant input signal. At worst, the parameters will remain constant and fail to converge to their true values.

4.4 Summary

In this section, we have presented a simple means of rotor control using the identified maps from desired rigid-body thrusts and torques to rotor thrusts and finally to *PWM* duty cycles of the brushless *DC* motors. The implemented method of control does not take time variance into account, and instead a rotor control system was considered with *PID* or *MRAC* regulation and *RLS* or *ct-RLS* parameter estimation, where the *PID* and *ct-RLS* proved superior for the considered rotor dynamics. The theory holds great promise and could potentially improve the robustness of the entire *UAV* control system, but currently remains in the theoretical realm. It will be implemented and tested in a real-time context in a separate M.Sc. project at *LTH* this spring.

5

Rigid-body control

Methods of rigid-body control are commonly developed in the Tait-Bryan parametrisation of rotation using one of three approaches to control. The first and simplest form of control is based on *PID*-control theory [Luukkonen, 2011], which has previously been used to great effect in real time applications as they may include feed forward terms computed by the property of differential flatness shown in **Section 3.1**. The second form of control relies on linear quadratic regulator theory (LQR) to stabilise the system [Castillo et al., 2005]. This may be done with a recursive and time-varying form as the system is differentially flat [Landry, 2015] and implement means of reachability verification by sums of squares optimisation as shown in [Tedrake, 2009]. The third and final category of controllers are predictive controllers, which are roughly equivalent to the LQR controllers when considering a time varying system over a finite control horizon or may be synthesised using $\mathcal{H}^{\infty}$-theory as shown in [Raffo et al., 2010].

In this chapter, we will first develop a method of saturating control signals in order to preserve controllability at all times, applicable to all prior mentioned methods of control. The considered system must be suitable for implementation in the embedded application to enable autonomous flight. Consequently, we do not consider the time varying LQR-type and predictive controllers due to computational constraints. We will introduce the control problem by defining methods commonly used in the Tait-Bryan parametrisation, compared in simulation with respect to

(i) Follow smooth references in the form of sinusoids

(ii) Follow aggressive references in the form of LP-filtered steps

(iii) Follow aggressive looping references assuming perfect state knowledge

(iv) Disturbance rejection with respect to varying mass

Having described the present methods of Tait-Bryan based control we will proceed with a real-time example. As rotation is parametrised as a rotation

matrix in the real-time application, the conventional Tait-Bryan controllers will not be implemented due to the phenomenon of dynamical unwinding (see **Section 2.4**). Instead, we present a computationally light method of geometric control acting directly on the rotation matrix, as originally theorised in [Lee et al., 2010]. The method has been subject to rigorous theoretical analysis on robustness and proven to work in real-time [Mellinger and Kumar, 2011]. In theory, a system implementing the geometric controller should be capable of satisfying the points (i)-(iii) even in a real-time context.

5.1 Saturations and controllability

In previous work, stabilising controllers implemented system dynamics linearised around a stable hovering state and were proven to work well [Castillo et al., 2005]. However, the implemented controller must be able to operate in the entire $\boldsymbol{\eta}$-space during aggressive flights, effectively excluding many conventional approaches. As an illustrative example of this, consider a commonly used continuous-time *PD*-controller with a Tait-Bryan parametrisation of rotation to map attitude and elevation errors to thrusts and torques [Luukkonen, 2011]. The equations may be summarised as

$$\begin{aligned} T &= (g + K_{D,z}(\dot{z}_r - \dot{z}) + K_{P,z}(z_r - z))\frac{m}{\cos(\phi)\cos(\theta)} \\ \tau_\phi &= (K_{D,\phi}(\dot{\phi}_r - \dot{\phi}) + K_{P,\phi}(\phi_r - \phi))I_{11} \\ \tau_\theta &= (K_{D,\theta}(\dot{\theta}_r - \dot{\theta}) + K_{P,\theta}(\theta_r - \theta))I_{22} \\ \tau_\psi &= (K_{D,\psi}(\dot{\psi}_r - \dot{\psi}) + K_{P,\psi}(\psi_r - \psi))I_{33} \end{aligned} \tag{5.1}$$

where $K_{i,j} \in \mathbb{R}^+$ is a control parameter, which, when correctly tuned, results in a stable hovering system if setting all references zero. However, if we were to let $\phi = \pi/2$, $T \to \infty$. Consequently, a saturation on the thrust $T \in [T_{min}, T_{max}]\ \forall t$ is required for the controller to remain feasible. However, if the upper bound is set too high in the controller, we might require a total thrust greater than what the motors are capable of generating. The difference in rotor speeds could then be zero, and we may find the system in a state where, by (2.9), $\tau_\phi = \tau_\theta = \tau_\psi = 0$ regardless of the reference torques. In this case the system will not be controllable until it has drifted away from the saturated state. By a similar argument, letting T_{min} be close to zero also diminishes controllability, and the problem can be solved by letting

$$0 \leq \epsilon 4 \cdot k\Omega_{min}^2 < 4 \cdot k\Omega_{min}^2 = T_{min} < T_{max} = (1-\epsilon)4 \cdot k\Omega_{max}^2 < 4 \cdot k\Omega_{max}^2, \tag{5.2}$$

where $[\Omega_{min}, \Omega_{max}]$ denotes the rotor speed saturation limits and $\epsilon \in (0, 1/2)$ determines how close the angular saturation is to the thrust saturation.

The final aspect that needs to be addressed is that the thrust should be limited differently depending on the system state. If $T_{max} >> mg$ and the attitude state changes quickly, the quad-rotor may reach a state where it is quickly accelerating towards the ground. This behaviour can be avoided by defining T_{max} differently depending on the sign of $\alpha = \arccos(\mathbf{R}_{\mathcal{BG}} \cdot \hat{\mathbf{z}}_{\mathcal{B}})$. We introduce the smooth function

$$\begin{cases} T_{max} = T^{+} & \text{if} \quad \alpha > 0 \\ T_{max} = T^{-} + (T^{+} - T^{-}) \sin^{2n}(\alpha) & \text{if} \quad \alpha < 0 \end{cases} \tag{5.3}$$

for some $n \in \mathbb{N}^{+}$, guaranteeing that the controller thrust remains within well defined bounds (see Figure 5.1). The example shows not only how the thrust saturations vary with n, but also hints to how conventional controllers fail when considering aggressive flight where a negative product $\mathbf{R}_{\mathcal{BG}} \cdot \hat{\mathbf{z}}_{\mathcal{B}}$ occurs, and a negative thrust is given for $\pi/2 < \alpha < \pi/2$ as a consequence. For future reference, we define the saturations as $\mathbf{S}_{\mathbf{\Omega}}(\mathbf{\Omega})$, where each element Ω_i is saturated to $[\Omega_{min}, \Omega_{max}]$. In addition, we let $\mathbf{S}_T(T)$, where the thrust is saturated such that $T \in [T_{min}, T_{max}]$ by equation (5.3), and let $\mathbf{S}_\tau(\boldsymbol{\tau})$ saturate the torque control signals such that $\tau_\phi, \tau_\theta, \tau_\psi \in [\tau_{min}, \tau_{max}]$ where the relationship in equation (5.2) is assumed with the cross configuration in (2.9).

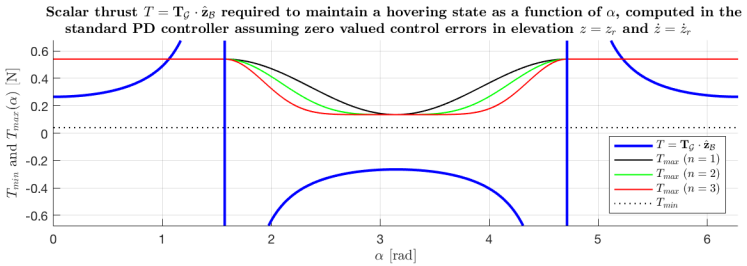

Figure 5.1 Computed thrust reference in the Tait-Bryan parametrised *PD* control system at various attitudes characterised by α. Looping the *UAV* without the saturations is impossible if using the simplified rotor-loop control, resulting in complex valued *PWM* signals, but made possible with the saturation function $\mathbf{S}_T(T)$, here shown for various orders $n = \{1, 2, 3\}$ with $\epsilon = 0.05$.

5.2 Tait-Bryan parametrised control

The most common way of devising controllers for *UAV* control is to consider the Tait-Bryan parametrisation of rotation. These methods of control, while subject to dynamical singularities induced by the gimbal lock, can be made to operate in the entire $\boldsymbol{\eta}$-space if adopting the approximation presented in **Section 2.1**. As the position and angular states cannot be controlled completely independently, the control system will be developed in two steps. The first is a stabilising inner attitude and elevation control, for which *PID* and *LQR* controllers will be considered. The second part is a tracking positional control in which positional errors are mapped to references in the Tait-Bryan angles. In this section, reference trajectories in control signals and states are sub-indexed with $\cdot_r$ and all simulations are made using the Euler-Lagrange rigid-body model with a Tait-Bryan parametrisation presented in **Section 2.1**.

Attitude and elevation control

As a first attempt at attitude control, a method similar to the *PD* control originally presented in [Luukkonen, 2011] is used to map control errors in Tait-Bryan angles to the corresponding torques and errors elevation to total thrust as given in (5.1). However, the thrust and torques are instead computed with the discrete time *PID*-regulators defined in **Appendix B.1**, referred to as $C_T(z)$ for the thrust and $C_\tau(z)$ for torques. In addition, feedforward terms from the flatness equations are included, and the absolute value of the rotational term $\cos(\phi)\cos(\theta)$ is taken in the thrust computation, such that

$$T = T_r + \frac{m}{|\cos(\phi)\cos(\theta)|} C_T(z)[z_r(t) - z(t)] \tag{5.4}$$

and

$$\begin{aligned} \tau_\phi &= \tau_{\phi,r} + I_{11} C_\tau(z)[\phi_r(t) - \phi(t)] \\ \tau_\theta &= \tau_{\theta,r} + I_{22} C_\tau(z)[\theta_r(t) - \theta(t)] \\ \tau_\phi &= \tau_{\psi,r} + I_{33} C_\tau(z)[\psi_r(t) - \psi(t)]. \end{aligned} \tag{5.5}$$

By applying the previously defined saturations $\mathbf{S}_T(\cdot), \mathbf{S}_\tau(\cdot)$ and $\mathbf{S}_\Omega(\cdot)$, operation in the complete $\boldsymbol{\eta}$-space is enabled at a low computational cost.

Linear quadratic regulator As an alternative to the *PID* controllers described above, we also consider the linear quadratic regulator (*LQR*) which has been used to great effect in previous work [Landry, 2015]. In the *LQR*, a quadratic cost function is minimised over an infinite horizon through a proportional matrix feedback, denoted $\mathbf{K}$, as derived in **Appendix B.3**. By this definition, errors in the states and control signals are punished by positive

definite matrices $\mathbf{Q}$ and $\mathbf{R}$ respectively, which can be modified to tune the controller. In the synthesis of our the linear quadratic regulator, the *UAV* dynamics are truncated to exclude the positional states in x and y, such that

$$\mathbf{x}(t) = \begin{bmatrix} z & \dot{z} & \boldsymbol{\eta}^T & \dot{\boldsymbol{\eta}}^T \end{bmatrix}^T \in \mathbb{R}^{8\times 1}, \qquad \mathbf{u}(t) = \begin{bmatrix} T & \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T \in \mathbb{R}^{4\times 1}. \tag{5.6}$$

We first define $\tilde{\mathbf{x}}(t) = \mathbf{x}_r(t) - \mathbf{x}(t)$ and $\tilde{\mathbf{u}}(t) = \mathbf{u}_r(t) - \mathbf{u}(t)$ as the deviation from a known reference trajectory in states, $\mathbf{x}_r(t)$, and control signals, $\mathbf{u}_r(t)$, generated from the differential flatness equations. The corresponding error dynamics of the truncated system are easily inferred from the linearized Tait-Bryan dynamics as given in **Section A.8**, where then

$$\dot{\tilde{\mathbf{x}}}(t) = \tilde{\mathbf{A}}_{\Delta\mathbf{x}}\tilde{\mathbf{x}}(t) + \tilde{\mathbf{B}}_{\Delta\mathbf{u}}\tilde{\mathbf{u}}(t). \tag{5.7}$$

By linearising about the stable hovering point and computing the gain, a time invariant regulator *TI-LQR* may be formed. Similarly to the *PD* controller of Lukkonen (5.1), this method of control becomes infeasible if operation far from the point of linearisation. A more computationally demanding option is to linearise the system about the reference trajectory and re-computing the *LQR* gain on each time step by solving the associated Riccati equation (see **Section B.3**), resulting in a time varying controller *TV-LQR*. In both the time varying and time invariant case, a condition for the Riccati equation to admit a solution is complete controllability of the system (5.7), which may be assumed to hold with high probability if using the approximation outlined in **Section 2.1**.

Integration and conditional anti-windup An inherent disadvantage with the standard *LQR*-formulation is that it, in essence, is a proportional state feedback controller and may give rise to stationary errors. To combat this issue, we let

$$\mathbf{C}_i = \begin{bmatrix} 1 & 0 & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbb{I}_{3\times 3} & \mathbf{0}_{3\times 3} \end{bmatrix} \in \mathbb{R}^{4\times 8} \tag{5.8}$$

and introduce four additional states as the integrated control errors,

$$\mathbf{x}_i(t) = \int_0^t \mathbf{C}_i[\mathbf{x}_r(\tau) - \mathbf{x}(\tau)]d\tau = \int_0^t \mathbf{C}_i\tilde{\mathbf{x}}(\tau)d\tau \in \mathbb{R}^{4\times 1}, \quad \mathbf{x}_i(0) = \mathbf{0}. \tag{5.9}$$

The derivative control errors are not integrated in order to avoid duplicate definitions of positional and angular control and preserve the full controllability of the model. In this form, the extended state-space error dynamics may then be expressed in terms of an extended state vector $\mathbf{x}_e(t) = \begin{bmatrix} \tilde{\mathbf{x}}(t) & \mathbf{x}_i(t) \end{bmatrix}^T \in \mathbb{R}^{12\times 1}$ with the matrices

$$\mathbf{A}_e = \begin{bmatrix} \tilde{\mathbf{A}}_{\Delta\mathbf{x}} & \mathbf{0}_{8\times 4} \\ \mathbf{C}_i & \mathbf{0}_{4\times 4} \end{bmatrix} \in \mathbb{R}^{12\times 12}, \quad \mathbf{B}_e = \begin{bmatrix} \tilde{\mathbf{B}}_{\Delta\mathbf{u}} \\ \mathbf{0}_{4\times 4} \end{bmatrix} \in \mathbb{R}^{12\times 4} \tag{5.10}$$

allowing the extended system to be written

$$\dot{\tilde{\mathbf{x}}}_e(t) = \tilde{\mathbf{A}}_e\tilde{\mathbf{x}}_e(t) + \tilde{\mathbf{B}}_e\tilde{\mathbf{u}}(t). \tag{5.11}$$

In order to allow large integral gains, a conditional anti-windup scheme (AW) is implemented to disable integral action when the total system thrust is in a saturated state. This is accomplished by simply letting $\mathbf{C}_i = \mathbf{0}$ when the thrust is in a saturated state $\mathbf{S}_T(T) - T = 0$ (see Figure 5.2). The considered formulation can be used to improve control and increase robustness to disturbances in applications less constrained by computational power, where it is compatible both with the LQR-type and predictive methods presented in [Landry, 2015] [Tedrake, 2009] [Raffo et al., 2010].

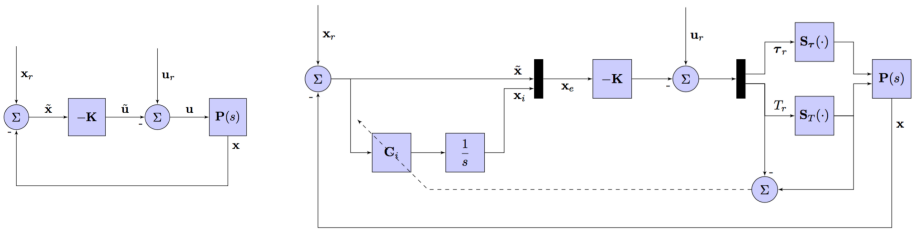

Figure 5.2 *Left:* The standard LQR controller operating on the error dynamics. *Right:* The LQR controller with integrating states and conditional anti-windup, controlling the rigid-body UAV process $\mathbf{P}(s)$.

Positional control With the previously defined inner controllers, computationally demanding outer model-predictive controllers can be run safely on the host computer at lower rates than the inner stabilising system. However, for the purpose of embedded autonomous control, we make use of standard PID formulation defined in **Section B.1** once again. A two degree of freedom controller for the xy-translation is considered, mapping positional control error in the global frame, $\mathbf{e}(t) = \mathbf{p}_r(t) - \mathbf{p}(t)$, to references pitch and roll. The general idea is to determine the angular references so as to make $\hat{\mathbf{z}}_B$ approach the vector $\begin{bmatrix} e_x(t) & e_y(t) \end{bmatrix}^T$ in the global xy-plane, done by mapping the control errors in the global frame to the body, such that

$$\begin{bmatrix} \phi(t) \\ \theta(t) \end{bmatrix} = \begin{bmatrix} \phi_r(t) \\ \theta_r(t) \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} C_x(z) & 0 & 0 \\ 0 & C_y(z) & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{R}_{\mathcal{GB}}\mathbf{e}(t), \tag{5.12}$$

which holds in the entire $\boldsymbol{\eta}$-space, allows for variable ψ-references and is compatible with any stabilising controller defined in the previous section.

Simulation study

In order to demonstrate the above theory, a set of simulations were run with the identified Crazyflie parameters. The rotors are saturated by $\Omega_i \in [0, 2500]$ $[rad/s]$, and, if applicable, the thrust saturations are computed with $\epsilon = 0.01$ and $n = 1$, saturating the torques to $\tau_\phi, \tau_\theta, \tau_\psi \in [-0.1, 0.1]$ $[N \cdot m]$. Furthermore, the inner controllers are assumed to run at a rate of 500 $[Hz]$ in order to achieve good stabilising control of the fast *UAV* dynamics.

For the *TV-LQR* controller, the diagonal positive-definite cost matrices were tuned according to

$$\mathbf{R} = \text{diag}([10^3,\ 10^7,\ 10^7,\ 10^7]) \in \mathbb{R}^{4\times4}$$
$$\mathbf{Q} = \text{diag}([10^4,\ 10^3,\ 10^2,\ 10^2,\ 10^2,\ 10,\ 10,\ 10]) \in \mathbb{R}^{8\times8}$$

and similarly, in the *TV-LQRiAW* controller synthesis, the cost matrices were determined as

$$\mathbf{R} = \text{diag}([10^3,\ 10^7,\ 10^7,\ 10^7]) \in \mathbb{R}^{4\times4}$$
$$\mathbf{Q} = \text{diag}([10^4, 10^3, 10^2, 10^2, 10^2, 10, 10, 10, 10^5, 10^3, 10^3, 10^3]) \in \mathbb{R}^{12\times12}.$$

In all simulations, the *PID* controllers were implemented in parallel form, described in detail in **Appendix B.1**. Using forward difference discretisation for the I-part and backward differences for the D-part, the controllers were tuned according to Table 5.1 with - indicating omitted terms in the formulation.

Table 5.1 Discrete time *PID* parameters used in the simulations.

Controller	K	Ti	T_d	N	γ	β	h	u_{min}	u_{max}
$C_T(z)$	10	-	5	100	1	1	0.002	-	-
$C_\tau(z)$	13	-	7	100	1	1	0.002	-0.1	0.1
$C_x(z)$	0.20	100	0.24	10	1	1	0.002	-10	10
$C_y(z)$	0.20	100	0.24	10	1	1	0.002	-10	10

Inner stabilising control with mass disturbance rejection Similar to the analysis of the rotor loop, comparison between the controllers is done in the relative integrated *MSE* metric, as defined in 4.2. Three tests are done to determine performance. The first test lets the closed loop system follow lowpass filtered unit steps in elevation and attitude with a single pole in 0.5 $[rad/s]$, such that $z_r(t) \in [0, 3]$ $[m]$ and $\phi_r(t), \theta_r(t), \psi_r(t) \in [-0.8, 0.8]$ $[rad]$.

The second simulation concerns smooth reference following, where sinusoidal references are followed with a frequency of $w = 1$ [rad/s] such that $z_r(t) \in [1, 3]$ $[m]$and $\phi_r(t), \theta_r(t), \psi_r(t) \in [-0.8, 0.8]$ $[rad]$ where the the phase of the references in attitude differ by $2\pi/3$.

In the third and most interesting test, references similar to that of the first test are followed and the system mass is suddenly increased from $m = 0.027$ to $m = 0.035$ $[kg]$ at t=15 $[s]$ before reverting to its original value of $m = 0.027$ $[kg]$ at t=35 $[s]$. In this case, the *MSE* error is evaluated for $t \in [10, 40]$ $[s]$ to capture the behaviour of a temporary increase in mass (see Table 5.2 and Table 5.3).

Table 5.2 Relative error norm in controller elevation response.

Test	$E_{p=1}^{PID}(z_r, z)$	$E_{p=1}^{LQR}(z_r, z)$	$E_{p=1}^{LQRiAW}(z_r, z)$
(1) step	2.01	1.43	1
(2) sinusoid	3.26	1.18	1
(3) mass load	2.32	2.73	1

Table 5.3 Relative error norm in controller attitude response.

Test	$E_{p=1}^{PID}(\boldsymbol{\eta}_r, \boldsymbol{\eta})$	$E_{p=1}^{LQR}(\boldsymbol{\eta}_r, \boldsymbol{\eta})$	$E_{p=1}^{LQRiAW}(\boldsymbol{\eta}_r, \boldsymbol{\eta})$
(1) step	24.30	1.25	1
(2) sinusoid	24.72	1.46	1
(3) mass load	10.43	1.1806	1

It is clear that the *LQR*-type controllers outperform the considered *PID*-type in the first two test-cases when operating relatively close to the stable hovering point, both for smooth and more aggressive references. However, this is assuming we have perfect knowledge of the system parameters in the *LQR* synthesis which is generally not the case in a physical ystem. It should be noted that the error metric is designed to clearly distinguish between the controller performances, and despite their differences, all tested controllers work sufficiently well to be considered for a real-time implementation (see Figure 5.3).

In the third test, we consider a shifting mass which may occur when the *UAV* picks up a small object. This disturbance is attenuated well in the *LQRiAW* due to the integration, and stationary error is clearly visible in the standard *LQR* controller as expected with proportional state feedback. Similarly, the *PD* elevation control is sensitive to mass changes and performs worse than the common *LQR* controllers

***Conditional anti-windup in* LQR** The considered conditional anti-windup in the time varying *LQR*-formulation is a powerful tool in attenuating disturbances, as it allows for greater punishment of the integral states through the **Q** matrix without risking large overshoots. To demonstrate the effect of

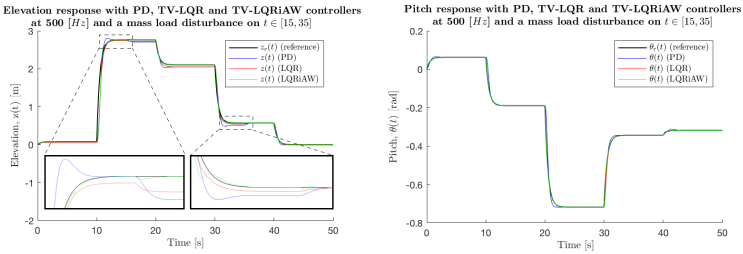

Figure 5.3 Comparison between the stabilising rigid-body controllers with simulated elevation, z, and roll, θ, when applying a mass load disturbance on $t \in [15, 45]$.

the implemented *AW* scheme, the model was set to follow lowpass filtered unit step references over $t \in [0, 30]$ [s] with a closed loop *LQRi*-regulator (see Figure 5.4). While in a non-saturated state, the two controllers yield the same system response, but the *LQRi* controller starts oscillating and threatens to diverge if the reference changes are large.

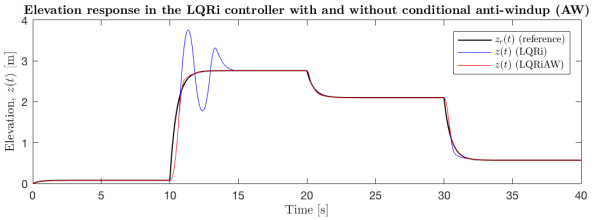

Figure 5.4 The effect of including conditional anti-windup in the *LQRi* controllers.

Looping manoeuvre with PID control As the *PID* is to be implemented in the embedded system due to computational constraints, the close loop system was set to perform a looping manoeuvre to demonstrate the effects of the saturations and stabilising control (see Figure 5.5). By increasing elevation temporarily and simultaneously ramping the reference in pitch $\theta_r(t)$, the *UAV* performs a loop over 1.5 [s] before stabilising in less than one second after completing the manoeuvre. Note here that the non-linear saturations allow the *PWM* signals to be split even when the thrust is in a saturated state, such that the reference in pitch reference may be followed at all times.

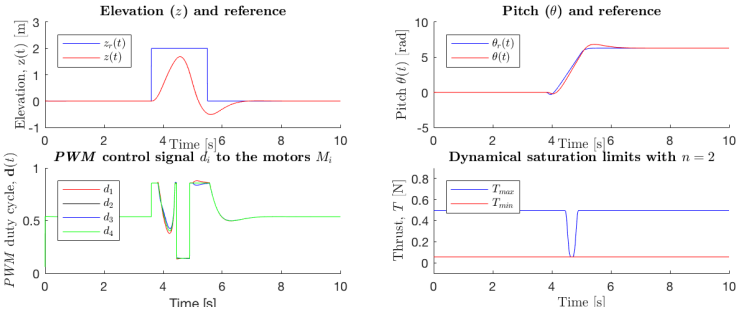

Figure 5.5 *Top left:* The reference elevation, z_r, and the elevation response z during the looping manoeuvre. *Top right:* The reference pitch, θ_r, and the pitch response θ. *Bottom left:* Generated *PWM* duty cycles for the motors during the loop. *Bottom right:* Saturation limits of in the thrust control signal as a function of time.

In addition, the saturation limits show the upper bound on the thrust decreases significantly when the quad-rotor is upside down, as designed in the definition of $\mathbf{S}_T(\cdot)$. Finally system stabilises at a pitch of 2π after completing the loop, showing that two infeasible regions with gimbal lock have been traversed successfully during the manoeuvre.

5.3 Geometric control

While there is some merit to using the simple *PID*-control with the Tait-Bryan angle representation, which was shown to be capable of performing complex manoeuvres such as looping, significant improvements can be made if abandoning the parametrisation entirely [Lee et al., 2010]. In the considered control system, full use is made of the flatness equations using force control with feedforward terms for translation in a computationally light controller. The positional control errors and velocity control errors are first defined as

$$\mathbf{e}(t) = \mathbf{p}_r(t) - \mathbf{p}(t), \qquad \text{and} \qquad \dot{\mathbf{e}}(t) = \dot{\mathbf{p}}_r(t) - \dot{\mathbf{p}}(t) \tag{5.13}$$

respectively. Including compensation for gravity and acceleration feedforward terms, the thrust along the body $\mathbf{z}_\mathcal{B}$ unit vector can be written

$$T = \mathbf{T}_\mathcal{G} \cdot \hat{\mathbf{z}}_\mathcal{B} = (\mathbf{K}_p\mathbf{e}(t) + \mathbf{K}_v\dot{\mathbf{e}}(t) + mg\hat{\mathbf{z}}_\mathcal{G} + m\ddot{\mathbf{p}}_r(t)) \cdot (\mathbf{R}^T_{\mathcal{BG}}\hat{\mathbf{z}}_\mathcal{G}). \tag{5.14}$$

In contrast to previously defined controllers, the geometric attitude control makes use of the reference rotation, $\mathbf{R}_r$, the reference body rates, $\boldsymbol{\omega}_r$, and

feed-forward torques, $\boldsymbol{\tau}_r$, from the differential flatness. With this information, we define the attitude error function

$$\Psi(\mathbf{R}_{\mathcal{BG}}, \mathbf{R}_r) = \frac{1}{2}\text{tr}[\mathbf{I} - \mathbf{R}_r^T \mathbf{R}_{\mathcal{BG}}] \tag{5.15}$$

omitting the need for parametrising the rotation in the degenerate quaternion or Tait-Bryan attitude spaces. This function is locally positive definite around $\mathbf{R}_{\mathcal{BG}} = \mathbf{R}_r$, as shown in [Bullo and Lewis, 2004]. Consequently, with the *vee* map $[\cdot]^\vee$ defined in **Section 2.1** as the inverse operation to the map $[\cdot]_\times$, it may be shown that $\Psi(\mathbf{R}_{\mathcal{BG}}, \mathbf{R}_r)$ is minimised with respect to $\mathbf{R}_{\mathcal{BG}}$ for

$$\mathbf{e}_R(t) = -\frac{1}{2}[\mathbf{R}_r^T \mathbf{R}_{\mathcal{BG}} - \mathbf{R}_{\mathcal{BG}}^T \mathbf{R}_r]^\vee = \mathbf{0} \Rightarrow \mathbf{R}_{\mathcal{BG}} = \mathbf{R}_r. \tag{5.16}$$

The idea is then to adjust define the rigid-body torques using $\mathbf{e}_R(t)$ in order to make the physical rotation, $\mathbf{R}_{\mathcal{BG}}$, approach the intended rotation, $\mathbf{R}_r$, as originally done in [Lee et al., 2010].

With the the body rate, $\boldsymbol{\omega}_{\mathcal{B}}$, is estimated or measured directly from the gyroscope, we may also form a body rate feedback law based on the rotation matrix time derivative identities presented in **Appendix A.11**. Clearly, the error between the reference and actual time derivative of the rotation, when compared in the same tangent spaces satisfy,

$$\dot{\mathbf{R}}_{\mathcal{BG}} - \dot{\mathbf{R}}_r(\mathbf{R}_r^T \mathbf{R}_{\mathcal{BG}}) = \mathbf{R}_{\mathcal{BG}}[\boldsymbol{\omega}_{\mathcal{B}}]_\times - \mathbf{R}_r[\boldsymbol{\omega}_r]_\times(\mathbf{R}_r^T \mathbf{R}_{\mathcal{BG}}) \tag{5.17}$$

Which may be written

$$\mathbf{R}_{\mathcal{BG}}([\boldsymbol{\omega}_{\mathcal{B}}]_\times - \mathbf{R}_{\mathcal{BG}}^T \mathbf{R}_r[\boldsymbol{\omega}_r]_\times \mathbf{R}_r^T \mathbf{R}_{\mathcal{BG}}) = \mathbf{R}_{\mathcal{BG}}([\boldsymbol{\omega}_{\mathcal{B}} - \mathbf{R}_{\mathcal{BG}}^T \mathbf{R}_r \boldsymbol{\omega}_r]_\times) \tag{5.18}$$

using Euler's rotational theorem (2.4) and the identities in **Appendix A.10**. We may then form a rotational time derivative control error

$$\mathbf{e}_\omega(t) = \mathbf{R}_{\mathcal{BG}}^T \mathbf{R}_r \boldsymbol{\omega}_r - \boldsymbol{\omega}_{\mathcal{B}} = \mathbf{0} \Rightarrow \dot{\mathbf{R}}_{\mathcal{BG}} = \dot{\mathbf{R}}_r. \tag{5.19}$$

A simple attitude controller may then be formed by

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_x \end{bmatrix} = \mathbf{K}_R \mathbf{e}_R(t) + \mathbf{K}_\omega \mathbf{e}_\omega(t) \tag{5.20}$$

giving rise to a seemingly robust control system used to great effect both in many practical applications [Mellinger and Kumar, 2011]. However, if complementing the controller with feedforward torques from the flatness equations (3.30) computed using the Newton-Euler equations (2.39), such that

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_x \end{bmatrix} = \mathbf{K}_R \mathbf{e}_R(t) + \mathbf{K}_\omega \mathbf{e}_\omega(t) + \boldsymbol{\tau}_r(t), \tag{5.21}$$

some interesting stability properties may be proven. Notably, the above controller gives rise to near global exponential stability with respect to the flat outputs [Lee et al., 2010]. If

$$\Psi(\mathbf{R}_{\mathcal{BG}}(t_0), \mathbf{R}_r(t_0)) < 2 \tag{5.22}$$

and

$$||\mathbf{e}_\omega(t_0)||_2 < \frac{2}{\lambda_{max}(\mathbf{I}_\mathcal{B})}\mathbf{K}_\omega(2 - \Psi(\mathbf{R}_{\mathcal{BG}}(t_0), \mathbf{R}_r(t_0))) \tag{5.23}$$

then, for some $a, b > 0$ is has been proven in [Lee et al., 2010] that

$$\Psi(\mathbf{R}_{\mathcal{BG}}(t), \mathbf{R}_r(t)) \leq \min\{2, ae^{-bt}\} \tag{5.24}$$

As such, the controller provides a guarantees on stability that has not yet been proved for the simpler *PID*-control system.

Experimental results

All presented systems, including *PID*, *TI-LQR*, *TI-LQRiAW* and geometric feedback control, were implemented separately in the embedded system for autonomous control. However, for the purposes of demonstrating the application, we will in this section only consider the geometric tracking $SE(3)$-control system. The controller proved very difficult to tune and proved far less robust than anticipated in the simulations, with performance varying greatly between quad-rotors. Three conceivable causes for this behaviour can be imagined, the first being noise induced by the motor vibrations, as the accelerometer is mounted in the same physical frame as the motors without damping. The second possible cause is the *UWB*-positioning system measurement variance, which is in the decimetre range as shown in later sections. The third potential problem is in the controller implementation itself, and more specifically the definition of rotational control errors, $\mathbf{e}_r(t)$. Surprisingly, the translational tuning $\mathbf{K}_p = 0.2 \cdot \mathbf{I}$ and $\mathbf{K}_d = 0.08 \cdot \mathbf{I}$ in (5.14) was found to be optimal both in simulation and the real time application, however, the rotational tuning parameters had to be lowered by a factor of 10, to $\mathbf{K}_R = 0.06 \cdot \mathbf{I}$ and $\mathbf{K}_\omega = 0.00025 \cdot \mathbf{I}$ in the attitude control (5.21), for the system to retain stability. This hints at a discrepancy between the physical system and the identified inertial parameters, and possibly an error in the firmware implementation of $\mathbf{e}_R(t)$.

The full control system was set up with *UWB* state estimation implementing the geometric *SE(3)* controller with the above tuning. The system was run autonomously with the embedded evaluation of pre-loaded trajectories using the flatness equations to compute the feed-forward terms. The first test shows the system response to step changes in elevation, alternating between $z = 1.2$ and $z = 1.8$ at a period of $t = 5$ [s] using the smoothing filter to enable evaluation of the *DF* equations (see Figure 5.6).

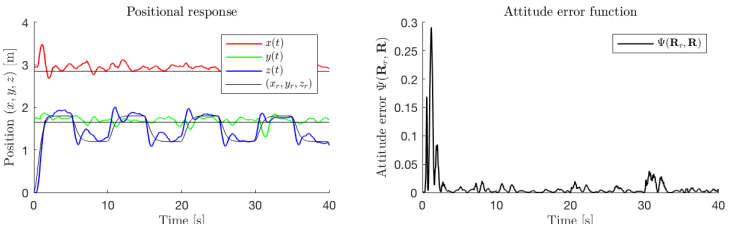

Figure 5.6 *Left*: Evaluated positional reference trajectory and estimated position as logged from the Crazyflie during autonomous flight with *UWB* localiaztion. *Right*: Attitude error function.

There is a slight overshoot in position due to the very aggressive tuning set to enable following of polynomial, sinusoid and Bezier curve trajectories. This can be remedied by moving the pole of the smoothing filter closer to origin in the embedded motion planning, here set to $\omega = -7.5$ in the experiments. This enables operation close to the dynamical constraints, while retaining stability in the system. The attitude error function $\Psi(\mathbf{R}_{\mathcal{BG}}, \mathbf{R}_r)$ increases slightly while ascending to a maximum elevation of $z = 1.8$, which is done in < 1 [s], and then remains close to zero as intended, being continuously minimised by the controller.

For less aggressive trajectories, such as a sinusoid parametrised movement in the xy-plane, the controller manages to follow the positional reference trajectory nicely considering the rather large estimator variance in the *UWB* system (see Figure 5.7). In this example, the reference trajectory was set to retain a height of $z_r(t) = 1$ $[m]$ and move in a perfect circle, defined by $(x_r(t), y_r(t)) = (\cos(2t), \sin(2t))$ which implies moving around the circle seven times in less than 20 seconds. In addition, the yaw reference was set to $\psi = -0.3$ $[rad]$ throughout the experiment, showing not only that reference trajectories may be followed in the entire flat output space, but also that the flatness implementation is correct, as the generated references in pitch and roll are followed and seemingly compliant with the system dynamics. When instead considering a unit step trajectory in the xy-plane, the aggressive smoothing once again results in positional overshoots, as each edge of the square is traversed in 2.5 seconds. While showcasing the translational response, this example also demonstrates that the infeasible trajectories generated by the *TSP-GA* may be loaded directly into the Crazyflie via the radio and followed autonomously.

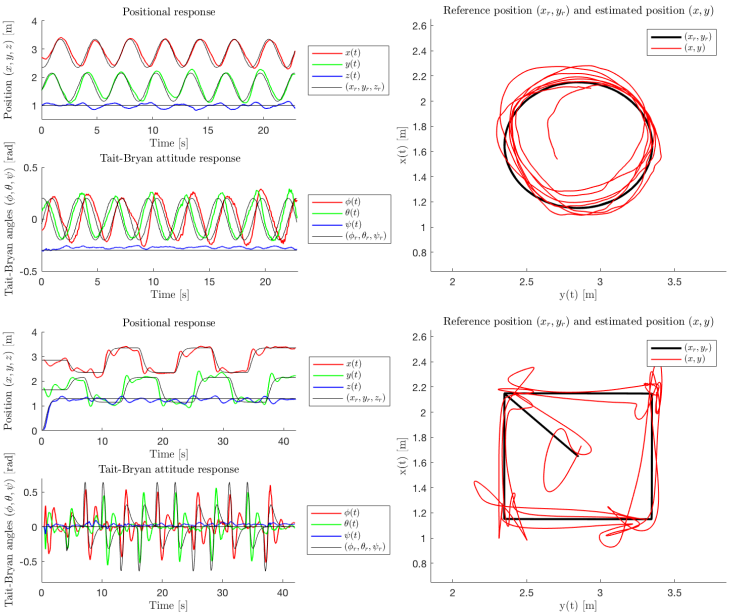

Figure 5.7 Following unit sinusoid (*Top*) and unit step (*Bottom*) reference trajectories in the xy-plane with *UWB* localiaztion. *Left*: References and estimated position and attitude. *Right* Positional trajectory in the xy-plane with *UWB* localiaztion.

The final example illustrates the motion planning and obstacle avoidance by generating a composite trajectory consisting of polynomial splines and unit steps. The trajectory is generated to avoid two tables and then used for autonomous navigation (see Figure 5.8). This demonstrates controller performance in cluttered environments, showing that high performance *UAV* applications are possible without expensive *MOCAP* systems.

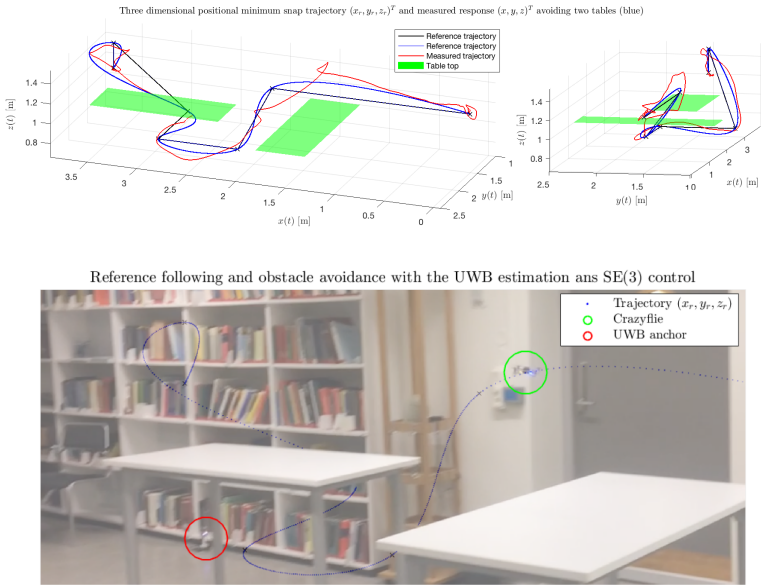

Figure 5.8 *Top:* Linear path (black) and generated reference trajectory (blue) as evaluated in the *UAV* firmware and the estimated position (red) avoiding two tables (green).

5.4 Summary

In this chapter, three distinct methods for rigid body control of the Crazyflie were derived with non-linear saturations to guarantee controllability of the *UAV*. A *PID*-control system was presented which proved capable of performing complex manoeuvres such as loops in simulation. The *LQR*-type controllers were presented, and an modification including the integration of positional errors was implemented with conditional anti-windup scheme to improve disturbance attenuation in the event of changing mass. Finally the geometric *SE(3)* controller was presented. Despite being being very difficult to tune and less robust than anticipated with respect to motor induced noise, the geometric controller performance was tested in a real-time application with *UWB*-localisation. The presented experiments demonstrati that the entire embedded real-time control system, with on-line trajectory evaluation, differential flatness equations and geometric tracking control is functional. It also shows that high performance autonomous control is possible with simple *UWB* estimation.

6

Inner state estimation

State estimation of the UAV can be accomplished in many ways, and a vast amount of factors need to be taken into consideration when formulating real-time compliant algorithms. The implementation should be a valid option for high performance research labs and private hobby implementations alike, and must therefore support a wide range of sensory equipment with varying degrees of precision. In this section we therefore consider (i) the *IMU* measurements, (ii) various motion capture systems (*MOCAP*), (iii) optical flow measurements, (iv) laser ranging and finally (v) *UWB* positioning. The goal is to support fusion of all possible combinations with information arriving at variable rates, and for this purpose, three types of state estimators are considered. As the *IMU* sensor is mounted on the *UAV*, it can be assumed to provide information at all times. Hence, in the minimal system of using only (i) the *IMU*, complementary filtering in the form of quaternion attitude heading and reference systems (*AHRS*) will be used. When additional sensory data is available, a scalar update extended Kalman filter (*EKF*) formulation will be developed, building on the preexisting work by Hamer [Mueller, 2016] [Mueller et al., 2015].

An advantage of the family of Kalman filters is their simple modification to support sensor fusion by including measurement equations in the update step. If done correctly, the drivers may push available information into the filter in any combination (i)-(v) without recompiling the firmware. Consequently, we will first present the filters and then proceed to discuss the sensors (ii)-(v) independently in terms of their fundamental limitations, outlier structure and measurement equations, detailing exactly how they enter the estimator update step. In addition, for (ii) the *MOCAP* systems, standalone methods of positional estimation will be given for stereo- and mono-vision cameras separately, enabling use of high performance systems such as Vicon or low performance cameras such as the Kinect. As for (iii) the optical flow and (iv) the laser ranging, the necessary drivers and developed hardware will be discussed in brevity. Finally for the *UWB*-system, two separate methods of ranging will be presented and robust ranging protocols will be described.

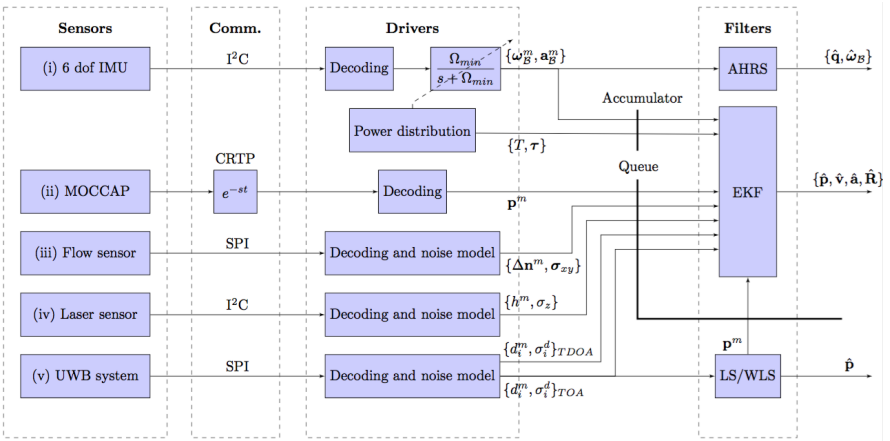

Figure 6.1 The inner state estimation with four blocks, the sensor block containing five separate sensors (i)-(v) for which information is communicated through one of three protocols, *SPI*, I^2C or the , Crazy Real-Time Protocol (*CRTP*) developed by Bitcraze. The sensory information is decoded in the driver block and fused with each other in the filter block to estimate the *UAV* states.

In addition, the Cramer-Rao lower bound (*CRLB*) [Sengijpta, 1995] is derived and used to relate the *UWB*-estimators to their theoretical limitations as a function of the anchor *UWB* placement. Finally, static estimation in the form of least-squares estimators (*LS*) will be derived for positional estimation in the *UWB* system, which may either be outputted directly or included in the dynamical filters to increase the amount of information included.

6.1 Model independent estimation

As the inertial measurement unit (*IMU*) is mounted to the *UAV*, all considered methods of state estimation will include gyroscope and accelerometer measurements as a starting point. The Crazyflie implements the *MPU-9250* package [InvenSense, 2014] which also provides magnetometer information, here discarded due to the many magnetic disturbances in indoor environments. It should be noted that the *IMU* is mounted in the same hardware frame as the motors without damping, and caution must be taken when handling the data as noise from the motors will be aliased when downsampled, potentially corrupting the spectral content at lower frequencies. The nominal range of operation for the motors is $\Omega_{min} \approx 500 < \Omega_i < 2500$

$[rad/s]$, with a stable hovering state occurring at $\Omega_{hover} = \Omega_i \approx 1700\ [rad/s]$ $\forall i \in \{1, 2, 3, 4\}$.

The lowest rate at which the *IMU*-sensor data is acquired is at $f_s = 500$ $[Hz]$. This corresponds to a Nyqvist frequency of $f_N = 250\ [Hz]$, resulting in the aliasing of all accelerometer and gyroscopic disturbances at angular rates of $\Omega > 2\pi f_N \approx 1570\ [rad/s]$ according to the Nyquist-Shannon sampling theorem [Unser, 2000]. In this mode of operation, the system performance is affected by the aliasing even when attempting to retain a stable hovering state as $\Omega_h > 2\pi f_N$. Consequently, a low-pass filter with a pole in $s = 500\pi$ $[rad/s]$ and unit static gain was implemented in the *IMU* driver and run at a rate of 1000 $[Hz]$, causing a negligible phase lag in the measurements while attenuating the motor noise at a low computational cost. With this precaution in place, we will proceed to discuss the model independent complementary *AHRS* filter developed by Madgwick for *IMU*-based attitude estimation [Madgwick et al., 2011].

Attitude heading and reference systems

A simple form of attitude estimation is the complementary *AHRS*-type filters, where gyroscopic measurements of angular rates in the body frame using the quaternion formalism in (2.36),

$$\boldsymbol{\omega}^q = \begin{bmatrix} 0 & \boldsymbol{\omega}_{\mathcal{B}} \end{bmatrix}^T = \begin{bmatrix} 0 & \omega_x & \omega_y & \omega_z \end{bmatrix}^T. \tag{6.1}$$

Similarly, the accelerometer measurements are defined as

$$\mathbf{a}^q = \begin{bmatrix} 0 & \mathbf{a}_{\mathcal{B}} \end{bmatrix}^T = \begin{bmatrix} 0 & a_x & a_y & a_z \end{bmatrix}^T. \tag{6.2}$$

The objective of the *AHRS* is to combine fuse this sensory data in order to determine the rotation quaternion from the body to global frame

$$\mathbf{q}_{\mathcal{BG}} = \begin{bmatrix} q_w & q_x & q_y & q_z \end{bmatrix}^T. \tag{6.3}$$

The typical approach is to align accelerometer measurement with a normalised gravitational field

$$\mathbf{g}^q = \begin{bmatrix} 0 & 0 & 0 & -1 \end{bmatrix}^T, \tag{6.4}$$

here defined in the global coordinate system to find noisy approximations of the rotation about the $\hat{\mathbf{x}}_{\mathcal{B}}$- and $\hat{\mathbf{y}}_{\mathcal{B}}$-axis, complemented by integrated gyroscopic measurements. In doing so, the drift in the angular estimates caused by the numerical integration of angular rates states is mitigated, but the rotation around the body z-axis will still be drift in time which is detrimental to autonomous control. However, as the yaw rate is known from the

gyroscopic measurements, the method is still of use in simpler joy-stick velocity control. Instead of the conventional Tait-Bryan angle parametrisation of rotation which is susceptible to gimbal lock, we will instead use the quaternion formulation with steepest descent optimisation as originally proposed by Madgwick et.al. [Madgwick et al., 2011].

Integration and optimization Using equation (2.37) from **Section 2.2**, we may express time derivative of the rotational quaternion in terms of the measured body rates as

$$\dot{\mathbf{q}}_{\mathcal{BG}} = \frac{1}{2}\mathbf{q}_{\mathcal{BG}} \otimes \boldsymbol{\omega}^q. \tag{6.5}$$

using the simple quaternion product (2.29). Discretising using forward Euler integration and with the forward shift operator, z, yielding the estimated quaternion of rotation from the gyroscope measurements as

$$\mathbf{q}_{\mathcal{BG}}^{(gyro)} = \mathbf{q}_{\mathcal{BG}} z^{-1} + \Delta t\Big(\frac{1}{2}\mathbf{q}_{\mathcal{BG}} z^{-1} \otimes \boldsymbol{\omega}^q\Big). \tag{6.6}$$

provided we know the sample rate and initial conditions of the system, which is $\mathbf{q}_{\mathcal{BG}} = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^T$ if the quadcopter starts at $\phi = \psi = \theta = 0$.

In order to get a rate of change in the quaternion from the accelerometer data, a simple optimisation problem to find the quaternion which best aligns the gravitational acceleration with the measured acceleration. This is done by using the quaternion rotation equation (2.36) in defining the function

$$\mathbf{F}(\mathbf{q}_{\mathcal{BG}}, \mathbf{g}^q, \mathbf{a}^q) = \mathbf{q}_{\mathcal{BG}} \otimes \mathbf{g}^q \otimes \mathbf{q}_{\mathcal{BG}}^* - \mathbf{a}^q \tag{6.7}$$

and solving the optimisation problem

$$\min_{\mathbf{q}_{\mathcal{BG}} \in \mathbb{H}} \Big(\mathbf{C}(\mathbf{q}_{\mathcal{BG}}, \mathbf{g}^q, \mathbf{a}^q)\Big) \qquad \text{where} \qquad \mathbf{C}(\cdot) = \frac{1}{2}\mathbf{F}(\cdot)^T\mathbf{F}(\cdot). \tag{6.8}$$

In Madgwick's filter, the gradient descent method is used with a single iteration, effectively allowing the optimisation program to be written in terms of the forward shift operator, z, as

$$\mathbf{q}_{\mathcal{BG}}^{(acc)} \approx \mathbf{q}_{\mathcal{BG}} z^{-1} - \mu \frac{\nabla \mathbf{C}(\mathbf{q}_{\mathcal{BG}} z^{-1}, \mathbf{g}^q, \mathbf{a}^q)}{||\nabla \mathbf{C}(\mathbf{q}_{\mathcal{BG}} z^{-1}, \mathbf{g}^q, \mathbf{a}^q)||_2}. \tag{6.9}$$

The advantages of the method is that it can be defined in terms of the function $\mathbf{F}(\cdot)$, as

$$\nabla \mathbf{C}(\cdot) = \frac{1}{2}\nabla \mathbf{F}(\cdot)^T\mathbf{F}(\cdot) \tag{6.10}$$

where both $\nabla\mathbf{F}(\cdot)$ and $\mathbf{F}(\cdot)$ are well defined and cheaply computed.

Special caution should here be taken, as the original derivation of the filter assumes the body frame and sensory frame are defined with $\hat{\mathbf{z}}_{\mathcal{B}} = -\hat{\mathbf{z}}_{\mathcal{S}}$, resulting in different signs in the $\mathbf{g}^q$ quaternion, hence we get different definitions of the function and gradients than in the original filter derivation [Madgwick et al., 2011].

Two substantial objections can be made at this point. The first is that clearly works best if the translational acceleration is small. This is an inherent issue in any complementary filtering technique relying on aligning the accelerometer measurements with the gravitational field. If the intention is to fly aggressively, the influence of $\mathbf{q}_{\mathcal{BG}}^{(acc)}$ must be made small. The second objection is that the filter never converges to a desired attitude. However, this is generally is not a problem as convergence is not a necessity if the constant μ is chosen to make the steps of the optimisation is made very small. Intuitively, the integration of the gyroscopic data causes filter divergence and μ should simply be set large enough to counteract this divergence. As such, and due to the normalisation in the gradient method, any filter constant $\mu = \alpha \Delta t ||\dot{\mathbf{q}}_{\mathcal{BG}}||_2$ with $\alpha > 1$ will prevent the filter from diverging.

Sensor fusion and bias compensation Having formulated the quaternion optimisation problem, Madgwick sets the convergence rate of the gradient descent formulation, $\mu/\Delta t$, equal to the divergence rate of the gyroscope measurements, β. This divergence rate may be computed as the rate of the linear drift in attitude when *IMU* perfectly still and integrating the gyroscope data. The standard complementary filter equations are then used to combine the high-frequency characteristics of the integrated gyroscope data with the low-frequency characteristics of the attitude estimated through the gradient descent optimisation,

$$\hat{\mathbf{q}}_{\mathcal{BG}} = \gamma \mathbf{q}_{\mathcal{BG}}^{(acc)} + (1-\gamma)\mathbf{q}_{\mathcal{BG}}^{(\omega)}. \quad \gamma \in [0,1]. \tag{6.11}$$

Using the proposed filter fusion coefficient $\gamma = \beta \Delta t / \mu \ll 1$, the filter can be written on the compact form

$$\hat{\mathbf{q}}_{\mathcal{BG}} = \hat{\mathbf{q}}_{\mathcal{BG}} z^{-1} + \Delta t \Big(\frac{1}{2} (\mathbf{q}_{\mathcal{BG}} z^{-1} \otimes \boldsymbol{\omega}^q) - \beta \frac{\nabla \mathbf{C}(\mathbf{q}_{\mathcal{BG}} z^{-1}, \mathbf{g}^q, \mathbf{a}^q)}{||\nabla \mathbf{C}(\mathbf{q}_{\mathcal{BG}} z^{-1}, \mathbf{g}^q, \mathbf{a}^q)||_2} \Big) \tag{6.12}$$

In the work of Madgwick, it also was shown that the quaternion rate of change as estimated by the accelerometer data can be used to estimate the gyroscope bias. By inverting the formula for the quaternion rate of change (2.37), the gyroscope bias, $\boldsymbol{\omega}_{bias}$, can be expressed as the static component of the angular rates as estimated by the accelerometer

$$\begin{bmatrix} 0 \\ \boldsymbol{\omega}_{bias} \end{bmatrix} = 2 \int \mathbf{q}_{\mathcal{BG}}^{*} z^{-1} \otimes \frac{\nabla \mathbf{C}(\mathbf{q}_{\mathcal{BG}} z^{-1}, \mathbf{g}^q, \mathbf{a}^q)}{||\nabla \mathbf{C}(\mathbf{q}_{\mathcal{BG}} z^{-1}, \mathbf{g}^q, \mathbf{a}^q)||_2} dt \tag{6.13}$$

where then the bias compensated angular rates can be written

$$\boldsymbol{\omega}_{comp} = \boldsymbol{\omega}_{\mathcal{B}} - \zeta \boldsymbol{\omega}_{bias} \tag{6.14}$$

including a weight ζ, as in the original bias compensation. The weight is included to control the convergence rate of the gyroscope bias estimate, which should be tuned situationally or gain scheduled depending on the application. Formulated this way, the filter has the advantage of operating without singularities in the entire attitude-space, but also suffers from a drift in the estimated rotation about the $\mathbf{z}_{\mathcal{B}}$ axis when only using the 6 degree of freedom *IMU* in spite of the bias compensation, as no magnetometer data is included in the filter formulation.

In order to validate the above theory before implementing the filter in the Crazyflie firmware, a simple simulation was run when controlling the Newton-Euler quaternion model using the geometric SE(3) controller (see Figure 6.2).

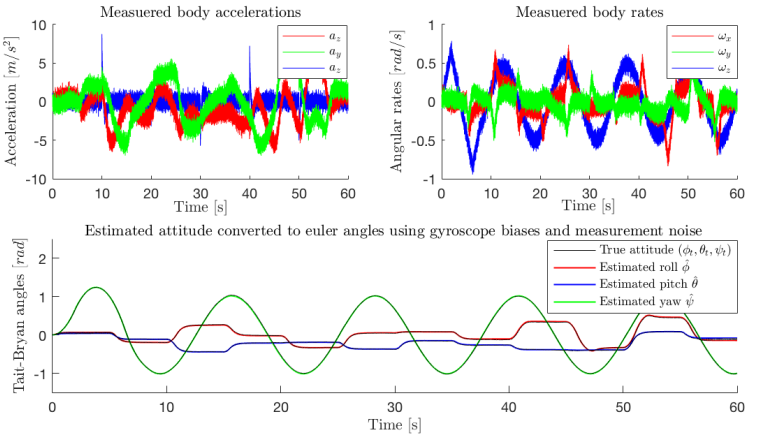

Figure 6.2 The closed loop quaternion-model set to follow a LP-filtered elevation references with a period of 10 [s], random LP-filtered references in roll and pitch $\phi_r(t), \theta_r(t) \in [-0.8, 0.8]$ $[rad]$ with a period of 5 $[s]$, sinusoidal yaw reference $\psi_r(t) = \sin(0.5t)$. The resulting IMU measurements are corrupted by biased noise corresponding to 0.1 $[degrees/s]$ of acceleration (top left) and accelerations (top right) are then filtered using Madgwick's method arriving at estimates very close to the true values.

While following random references in elevation and attitude, body rates and body accelerations were corrupted by gaussian noise and linear drifts.

From these measurements, the attitude was estimated, parametrised in Tait-Bryan ZYX angles and compared to the true angular response, yielding slight errors but near perfect estimation in the simulation time, validating the implementation against previous theory derived originally by Madgwick [Madgwick et al., 2011]. The presented bias-compensated six degree of freedom filter was subsequently implemented in the Crazyflie firmware.

6.2 Model based state estimation

Much prior work has been done on *UAV* state estimation in navigation systems, and the standard approach is to implement variations of the extended Kalman filter (*EKF*) [Huxel and Bishop, 2009]. This is done for good reason, as the employed approximations, while restrictive in terms of robustness, makes *EKF* computationally efficient compared to alternatives such as the standard unscented Kalman filters (*UKF*) [Wan and Van Der Merwe, 2000] and Bayesian particle filters (*GPF*) [Arulampalam et al., 2002] [Hol et al., 2006] [Douc and Cappé, 2005]. Prior to this thesis, an *EKF* had been implemented for sensor fusion of time-of-arrival (*TOA*) measurements from the *UWB LPS* [Mueller, 2016] [Mueller et al., 2015]. As a consequence, we will present the current estimator equations together with modifications to improve performance, discuss numerical stability and potential issues with observability.

Assumptions and definitions

Consider the discrete time system

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \in \mathbb{R}^{N\times 1} \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k) \in \mathbb{R}^{M\times 1} \end{aligned} \tag{6.15}$$

with gaussian noise $\mathbf{w}_k$ and $\mathbf{v}_k$ with the covariance properties

$$\mathbb{E}\left\{\begin{bmatrix}\mathbf{w}_k \\ \mathbf{v}_k\end{bmatrix}\right\} = \mathbf{0} \qquad \mathbb{E}\left\{\begin{bmatrix}\mathbf{w}_k \\ \mathbf{v}_k\end{bmatrix}\begin{bmatrix}\mathbf{w}_k \\ \mathbf{v}_k\end{bmatrix}^T\right\} = \begin{bmatrix}\mathbf{Q} & 0 \\ 0 & \mathbf{R}\end{bmatrix} \tag{6.16}$$

for some positive definite $\mathbf{Q} \in \mathbb{R}^{N\times N}$, $\mathbf{R} \in \mathbb{R}^{M\times M}$, thereby assuming uncorrelated state- and measurement noise. This approximation may be crude and should be revisited in future research, as the gyroscope and accelerometer are mounted in the same frame and likely subject to the similar noise from the motors. In the *EKF*, just as in the standard Kalman filter, the state vector probability density function, $p(\mathbf{x}_k|\mathcal{S}_k)$ conditioned by the sequence of past measurements and control signals $\mathcal{S}_k = \{\mathbf{y}_0, \cdots, \mathbf{y}_k, \mathbf{u}_0, \cdots, \mathbf{u}_{k-1}\}$, is propagated through time in two steps. As the system (C.16) is assumed to

be non-linear, the *EKF* uses a first order multivariate Taylor-approximation to predict a future state, $\mathbf{x}_k^f$, which is then corrected through filtering using a Kalman gain resulting in the sub-optimal estimate $\hat{\mathbf{x}}_k$. The reader is referred to **Appendix C.3** complete derivation of the standard *EKF*, but for future reference, we define the system Jacobians

$$\mathbf{F}_k = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{0})}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k} \qquad \mathbf{H}_k = \frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{0})}{\partial \mathbf{x}}\Big|_{\mathbf{x}_k^f, \mathbf{u}_k} \tag{6.17}$$

where $\mathbf{F}_k$ can be expressed analytically at all times in the quaternion dynamics, and $\mathbf{H}_k$ depends on the available measurements. In addition, if assuming that noise may be non-additive, we let

$$\mathbf{W}_k = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w}_k)}{\partial \mathbf{w}}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \mathbf{w}_k} \qquad \mathbf{V}_k = \frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{v}_k)}{\partial \mathbf{v}}\Big|_{\mathbf{x}_k^f, \mathbf{u}_k, \mathbf{v}_k}. \tag{6.18}$$

Observability

In all Kalman filters, the property of observability is of vital importance for estimator stability. Any unobservable state will per definition not be seen through the measurement equation $\mathbf{h}(\mathbf{x}_k, \mathbf{v}_k)$, cannot be corrected and will therefore never converge to a meaningful solution. The *UAV* dynamics are highly non-linear, greatly complicating the notion of observability, but some insight can be gained from analysing the linearised system constituted by the error dynamics of the linearised system $\{\mathbf{F}_k, \mathbf{H}_k\}$. Observability for a specific linearisation in the *EKF* can then be determined by computing the observability Gramian

$$\mathbf{W}_o(0, \infty) = \int_0^\infty (e^{\mathbf{F}_k \tau})^T \mathbf{H}_k^T \mathbf{H}_k e^{\mathbf{F}_k \tau} d\tau \in \mathbb{R}^{n \times n}. \tag{6.19}$$

which by the Cayley-Hamilton theorem [Vilfan, 1973] is equivalent to full row-rank of the observability matrix

$$\mathcal{O} = \begin{bmatrix} \mathbf{H}_k \mathbf{F}_k^0 \\ \vdots \\ \mathbf{H}_k \mathbf{F}_k^n \end{bmatrix} \in \mathbb{R}^{nm \times n-1}. \tag{6.20}$$

To illustrate the effects of observability on state estimation in the *UAV* dynamics in the context of the *EKF*, consider the simple one dimensional double integrator given by, $\ddot{x} = u$, with a non-linear measurement equation $h(x, \dot{x})$. In state-space form, the discrete-time system at a time-step of Δt is written

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \mathbf{x} + \begin{bmatrix} \Delta t^2/2 \\ \Delta t \end{bmatrix} \mathbf{u} \quad \text{with} \quad \mathbf{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}, \mathbf{u} = \begin{bmatrix} 0 \\ u \end{bmatrix}, \tag{6.21}$$

where due to the system being linear and time invariant, $\mathbf{F}_k = \mathbf{A} \ \forall k$.

If the measurement equation contains information on the position, $h(x, \dot{x}) = \alpha x \Rightarrow \mathbf{H}_k = \begin{bmatrix} \alpha & 0 \end{bmatrix} \Rightarrow \text{rank}(\mathcal{O}) = 2$. In the second case, we assume it only contains information on the velocity, and $h(x, \dot{x}) = \alpha \dot{x} \Rightarrow \mathbf{H}_k = \begin{bmatrix} 0 & \alpha \end{bmatrix} \Rightarrow \text{rank}(\mathcal{O}) = 1 \neq 2$ making the positional state unobservable. In the third case, we have composite measurements, such as $h(x, \dot{x}) = \alpha x \dot{x}$, where for instance $x = \beta \neq 0, \dot{x} \equiv 0 \Rightarrow \mathbf{H}_k \equiv \begin{bmatrix} 0 & \alpha\beta \end{bmatrix} \Rightarrow \text{rank}(\mathcal{O}) = 1 \neq 2$ resulting in a locally unobservable position, x.

As the translational dynamics of the quadcopter is a triple integrator in essence, we can only guarantee stability if the measurement equation contains information on the zeroth order states, that is translation and attitude. If it only provides second order derivative information (as from the *IMU* accelerometer) the filter will quickly diverge in a quatratic fashion for the positional states. However, if complementing this with first order derivative information (as from optical flow), we will se a slower divergence as a linear drift in the positional estimates. Global convergence can only be attained if including zeroth order information (as from the *UWB LPS* or *MOCAP* systems). In the final case of a composite measurement equation, we may temporarily loose information depending on its structure, potentially rendering the system locally unobservable even with zeroth order measurements.

Kalman gain approximation

In many *IMU* driven navigation systems, the most costly aspect of the *EKF* is the inversion and many matrix operations of the gain computation and subsequent covariance update. A crude approximation of the true Kalman gain can sometimes be used to great effect if the system is highly observable, where the measurements are included in a scalar fashion [Huxel and Bishop, 2009]. Here, we let $\epsilon^i = z_k^i - h^i(\mathbf{x}_k^f, \mathbf{0})$ denote the error between measurement and prediction in the measurement equation of index i where $i \in 1, \cdots, M$. Furthermore, let $\mathbf{H}_k^i$ denote the corresponding row in the measurement Jacobian, let $\mathbf{K}_k^i$ be a Kalman gain column vector and $\mathbf{R}^{ii}$ be the i^{th} diagonal element of $\mathbf{R}$. This allows for measurements to enter but naturally comes at the cost of the Kalman gain decreasing on each iteration i due to the injection of measurement noise in the covariance matrix. As such, the first measurements in the sequence $\{1, ..., M\}$ will affect the posterior covariance matrix greatly, while the later contributions will have little effect on the estimate. This can be remedied somewhat by including the more reliable measurements first in the sequence $\{1, ..., M\}$ [Huxel and Bishop, 2009].

Joseph form update and numerical stability

The *EKF* is not an optimal filter as opposed to the standard Kalman filter, and the first order approximation could potentially be a crude approximation

if the dynamics are highly non-linear. In addition, the estimate and covariance matrices do not accurately capture the conditional probability density function $p(\mathbf{x}_k|\mathcal{S}_k)$ and few guarantees can be made regarding its convergence properties. A precaution to decrease the risk of estimator divergence due to numerical errors is to use the Joseph correction, computing the covariance matrix according to (C.29) in order to preserve symmetry. We may also enforce symmetry and check boundedness of $\mathbf{P}_k$ on each correction, resetting the filter if necessary (see Algorithm 2).

Receive $\mathbf{u}_{k-1}, \mathbf{y}_k$;
Prediction step
$\mathbf{x}_k^f = \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0})$;
$\mathbf{P}_k^f = \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{W}_{k-1}\mathbf{Q}\mathbf{W}_{k-1}^T$;
Correction step
for *i = 1,...,M* **do**
  $\epsilon^i = y_k^i - h^i(\mathbf{x}_k^f, \mathbf{0})$;
  $\mathbf{K}_k^i = \mathbf{P}_k^f(\mathbf{H}_k^i)^T/[\mathbf{H}_k^i\mathbf{P}_k^f(\mathbf{H}_k^i)^T + \mathbf{V}_k^i\mathbf{R}^{ii}(\mathbf{V}_k^i)^T]$;
  $\hat{\mathbf{x}}_k = \mathbf{x}_k^f + \mathbf{K}_k^i\epsilon^i$;
  $\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k^i\mathbf{H}_k^i)\mathbf{P}_k^f(\mathbf{I} - \mathbf{K}_k^i\mathbf{H}_k^i)^T + \mathbf{K}_k^i\mathbf{V}_k^i\mathbf{R}^{ii}(\mathbf{V}_k^i)^T(\mathbf{K}_k^i)^T$;
  $\mathbf{x}_k^f = \hat{\mathbf{x}}_k,\ \mathbf{P}_k^f = \mathbf{P}_k$;
end
$\hat{\mathbf{x}}_k = \mathbf{x}_k^f,\ \mathbf{P}_k = \mathbf{P}_k^f$;
$\mathbf{P}_k = \frac{1}{2}(\mathbf{P}_k + \mathbf{P}_k^T)$;

Algorithm 2: The scalar update Joseph-form *EKF* with enforced symmetry

6.3 MOCAP positioning

In the considered motion capture positioning, the goal is to locate a single *UAV* using any one of two camera types. In all considered applications, a taken image, $\mathcal{I} \in \mathbb{R}^{N_x \times N_y}$, is defined by a total of N_x and N_y pixels. Each pixel, $\mathcal{I}_{i,j}$, may contain data on either greyscale brightness, as with common web cameras, or depth $d \in \mathbb{R}^+$ $[m]$ as with disparity image functionality in the more advanced Kinect cameras [ASUS, 2016]. Throughout this section, we presume to know the camera angle of aperture, denoted θ_{px}, θ_{py}. With this limited information, we will proceed to develop methods of state estimation for both single- and multiple camera systems, making use of depth information when available and detailing how the information enters the *EKF*.

Measurement equations and limitations

In order to distinguish an object in the frame, a set of N background images, $\mathcal{I}_i^b$, are taken and the mean image is computed $\bar{\mathcal{I}}^b = \sum_{i=1}^{N} \mathcal{I}_i^b / N$. The resulting image is subtracted from any subsequently taken image, resulting in a difference image $\mathcal{I}^{diff} = \mathcal{I} - \bar{\mathcal{I}}^b$. The difference image is then lowpass filtered by convolution with a simpler gaussian blur kernel

$$\mathcal{F} = \frac{1}{16} \rceil \begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix} \lfloor \tag{6.22}$$

such that the convolved image becomes

$$\tilde{\mathcal{I}}^{diff} = \mathcal{F} * \mathcal{I}^{diff} = \sum_{n=0}^{2} \sum_{m=0}^{2} \mathcal{F}_{n,m} \mathcal{I}_{i-n,j-m}^{diff} \tag{6.23}$$

Next, all the M pixel indices (i, j) corresponding to pixels in the convolved image satisfying $\tilde{\mathcal{I}}_{i,j}^{diff} > \epsilon$ for some numerical limit $\epsilon \in \mathbb{R}^+$ are used to find a pixel "centre of pixel mass", by

$$(\bar{n}_x, \bar{n}_y) = \left\lceil \frac{1}{M} \sum_{n=1}^{M} (i, j) \right\rfloor \tag{6.24}$$

where then, by necessity, $\bar{n}_x \in [1, N_y]$ and $\bar{n}_y \in [1, N_y]$.

Next, we presume to know the camera position of the i^{th} camera in the room, $\mathbf{c}_i \in \mathbb{R}^3$, in the global frame, $\mathcal{G}$. Furthermore, the calibrated normal of the direction in which the camera is facing, $\mathbf{n}_i^{cal} \in \mathbb{R}^3$, with a corresponding rotation $\mathbf{R}_i^{cal} \in \mathbb{R}^{3\times3}$ such that $\mathbf{n}_i^{cal} = \mathbf{R}_i^{cal}[0, 0, 1]^T$ are both presumed to be known. The direction in which the UAV supposedly is can be determined by defining

$$\alpha_x = \theta_x \Big(\frac{n_x}{N_x} - \frac{1}{2} \Big), \qquad \alpha_y = \theta_y \Big(\frac{n_y}{N_y} - \frac{1}{2} \Big) \tag{6.25}$$

letting

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_x) & -\sin(\alpha_x) \\ 0 & \sin(\alpha_x) & \cos(\alpha_x) \end{bmatrix}, \qquad \mathbf{R}_y = \begin{bmatrix} \cos(\alpha_y) & 0 & -\sin(\alpha_y) \\ 0 & 1 & \\ \sin(\alpha_y) & 0 & \cos(\alpha_y) \end{bmatrix} \tag{6.26}$$

and finally forming

$$\mathbf{n}_i = \mathbf{R}_i^{cal} \mathbf{R}_x \mathbf{R}_y [0, 0, 1]^T. \tag{6.27}$$

With this theory, we have means of determining the direction of the detected UAV from a fixed camera position in the global frame. However, the method assumes that the UAV is the only moving object in the scene.

Cameras with depth vision In the case of the Kinect cameras, which provide disparity images with depths, we only need a single camera to provide the position of the *UAV* in the global frame. The general expression of taking the mean from N cameras is given by

$$\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^T = \frac{1}{N} \sum_{i=1}^{N} \mathbf{c}_i + d_i \mathbf{n}_i. \tag{6.28}$$

Entering the information in the *EKF* as rows in the vector-valued measurement function, $\mathbf{h}(\mathbf{x}, \mathbf{0})$ is trivial. We simply let $\mathbf{h}^{row}(\mathbf{x}, \mathbf{0}) = x\hat{x}$, $\mathbf{h}^{row+1}(\mathbf{x}, \mathbf{0}) = y\hat{y}$, $\mathbf{h}^{row+2}(\mathbf{x}, \mathbf{0}) = z\hat{z}$ with $\hat{y}$ denoting the position of the state y in the state vector $\mathbf{x}$. In all cases, the jacobians simply become $\mathbf{H}^{row}(\mathbf{x}) = \hat{x}$, $\mathbf{H}^{row+1}(\mathbf{x}) = \hat{y}$, $\mathbf{H}^{row+2}(\mathbf{x}) = \hat{z}$.

Cameras without depth vision Consider a set of lines $\mathbf{l}_i = \mathbf{c}_i + t\mathbf{n}_i, \ \ t \in \mathbb{R}$ given by the camera positions $\mathbf{c}_i$ and normal directions $\mathbf{n}_i$. Generally, these lines are unlikely to intersect, and instead, we must find the point closest to all of these lines in some optimal sense. In the standard LS formulation, the closest distance squared between an arbitrary point $\mathbf{p} \in \mathbb{R}^3$ and a line $\mathbf{l}_i$ may then be written,

$$||\mathbf{p} - \mathbf{l}_i||_2^2 = (\mathbf{c}_i - \mathbf{p})^T (\mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T)(\mathbf{c}_i - \mathbf{p}) \tag{6.29}$$

as shown in **Appendix C.4** using the projector $\mathbf{P}_i = \mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T$. The total cost of an arbitrary point $\mathbf{p}$ for a total of N lines can then be defined as

$$J(\mathbf{p}) = \sum_{i=1}^{N} ||\mathbf{p} - \mathbf{l}_i||_2^2 = \sum_{i=1}^{N} (\mathbf{c}_i - \mathbf{p})^T (\mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T)(\mathbf{c}_i - \mathbf{p}) \tag{6.30}$$

which, when differentiated with respect to $\mathbf{p}$ gives an extremal point at

$$\frac{\partial J(\mathbf{p})}{\partial \mathbf{p}} = -2(\mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T)(\mathbf{c}_i - \mathbf{p}) = 0 \tag{6.31}$$

which is indeed a minimum, as

$$\frac{\partial^2 J(\mathbf{p})}{\partial \mathbf{p}^2} = 2(\mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T) = 2\mathbf{P} \tag{6.32}$$

is positive semidefinite on account of $\mathbf{P}$ being idempotent. With this result, we simply form the linear system

$$\mathbf{A} = \sum_{i=1}^{N} (\mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T), \qquad \mathbf{b} = (\mathbf{I} - \mathbf{n}_i \mathbf{n}_i^T)\mathbf{c}_i \tag{6.33}$$

from which the *LS* estimate of the position can be written

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} ||\mathbf{A}\mathbf{p} - \hat{\mathbf{b}}||_2 = \mathbf{A}^{\dagger}\hat{\mathbf{b}} \tag{6.34}$$

as solved by the standard Normal equations where $\mathbf{A}^{\dagger} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ denotes the lefthand Moore-Penrose pseudo-inverse [Trefethen and Bau III, 1997]. This *LS* formulation enables depth estimation from two or more cameras, improving with the number of cameras included. If implementing two cameras, care must be taken as any situation with $\mathbf{l}_1$ and $\mathbf{l}_2$ being near parallel will result in a poorly conditioned system. From the *LS* estimate, the measurement equations and corresponding measurement Jacobians are formed just as in the case of depth vision cameras above (see Figure 6.3)

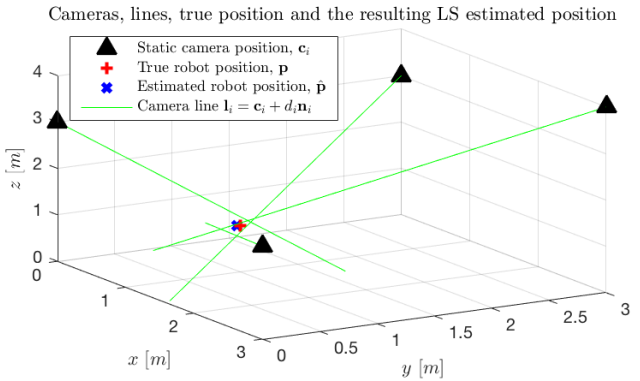

Figure 6.3 True and estimated position using the *LS* formulation in a four camera system and perturbing the true normals slightly.

Real-time example

To demonstrate the real-time implementation with *MOCAP* estimation using depth sensing cameras, the Crazyflie was set to swing in a string of length r $[m]$, estimating the position using the *EKF* with a single Kinect 1 camera [ASUS, 2016]. To verify the implementation, we consider the dynamics of a three dimensional damped spherical pendulum derived using the Euler-Lagrange equations and Hamilton's principle in a spherical coordinate system [Young et al., 2007]. When complemented with energy dissipation due to air resistance, b, the governing equations may be written

$$\begin{cases} \ddot{\theta} = \left(\dot{\theta}^2 \cos(\phi) + \dfrac{g}{r}\right) \sin(\phi) - b\dfrac{r|\dot{\phi}|\dot{\phi}}{m} \sin(\phi) \\ \ddot{\phi} = -2\dfrac{\dot{\theta}\dot{\phi}}{\tan(\phi)} - b\dfrac{r \sin(\dot{\phi})|\dot{\theta}|\dot{\theta}}{m} \end{cases}, \qquad \begin{cases} x = r \sin(\theta)\cos(\phi) \\ y = r \sin(\theta)\sin(\phi) \\ z = r \cos(\phi) \end{cases} \tag{6.35}$$

By simulation of the spherical pendulum model, we may visually compare the estimated position of the *UAV* with the governing dynamics of the pendulum, determined by initial conditions and parameters of mass, $m = 0.027$ $[kg]$, pendulum length, $r = 2$ $[m]$, drag coefficient $b = 0.001$ $[rad^2/kg]$ and gravitational acceleration $g = 9.81$ $[m/s^2]$(see Figure 6.4). As expected, the mass follows an elliptical trajectory when approaching the stationary point in $(r, \theta, \psi) = (-2, 0, 0)$. Close to this point, we may approximately model movement along the x- and y-axes as an under-damped harmonic oscillator

$$\ddot{x} + 2\xi\omega_0\dot{x} + \omega_0^2 x = 0 \tag{6.36}$$

for which it is easily verified that the general solution may be written

$$x(t) = A_0 e^{-\xi\omega_0 t} \cos(\omega_0\sqrt{(1-\xi^2)}t + \phi) \tag{6.37}$$

when $\xi^2 < 1$ [Glad and Ljung, 2000]. By generating an estimated trajectory of the swinging mass, the coefficients $\{A_0, \xi, \omega_0, \phi\}$ may be fitted with a non-linear regression, showing that the estimator accurately follows a trajectory compliant with the spherical pendulum dynamics (see Figure 6.4).

The chosen example illustrates a problem of positioning using a single Kinect camera, occurring when $|x(t)|$ is large. Here the *UAV* comes close to the boundary of the flyable volume, which is approximately 1 $[m^3]$ due to the small size of the quad-rotor and limitations of the camera. At these points, we are more likely to loose track of the *UAV* resulting in many outlier measurements during which the *EKF* updates without correction, resulting in a deviations from the fitted trajectory. Furthermore, this example shows not only that the Kinect implementation is functional, but more importantly that the *EKF* implementation works. Consequently, any *MOCAP* system

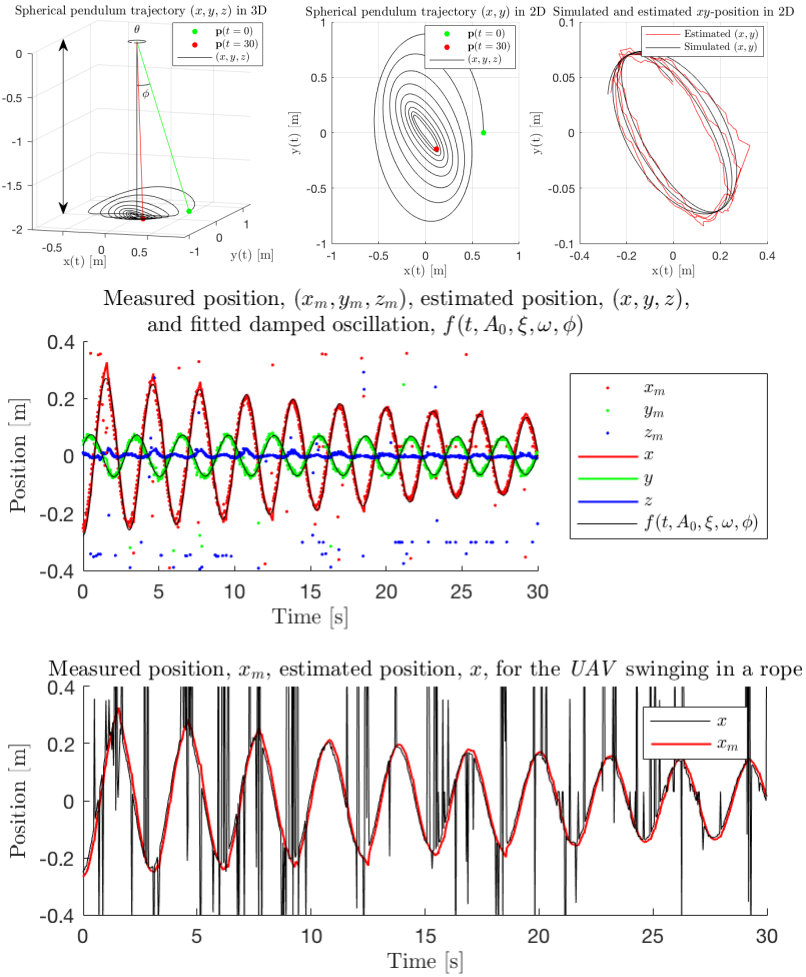

Figure 6.4 *Top left* Simulation of the spherical pendulum in $\mathbb{R}^3$. *Top center* Projection of the trajectory in the xy-plane *Top right* Estimated position using the Kinect 1 and posterior simulated response. *Center:* Measured position (x_m, y_m, z_m) as computed from $(\bar{n}_x, \bar{n}_y)$, estimated position (x, y, z) and fitted damped harmonic oscillation (black). *Bottom:* Estimated position, x, and measured position, x_m, including outliers.

outputting an estimated position of the *UAV* may be used instead of the Kinect, including the web-camera approach with *LS* estimation and more advanced systems such as the *VICON*. In fact, the *VICON* system has already been run with the *EKF* at *ETH* [Mueller et al., 2016].

6.4 UWB positioning

There exist many ways of passively estimating positions in an *UWB* system which can be employed depending on the hardware at hand. One of the simpler approaches is to use anchors designed to detect the angle of arrival (*AOA*) at which the wavefront hits their antenna [Pahlavan et al., 2002]. An alternative method is the received signal strength approach (*RSS*), which has proven especially effective for multi-path echoes in non-line-of-sight (*NLOS*) conditions [Akgul, 2010] [Gentile et al., 2013]. Here, a logarithmic power-distance relationship is used to determine estimate position by analysing the signal amplitude in the receivers. The third an final method is commonly referred to as the time-of-flight (*TOF*) approach. In this case, data is encoded in packets and transmitted using an *UWB*-radio, detailing a precise time when the packet is sent. By comparing the time stamps at which the data was transmitted and received, the time lag in the communication may be computed giving a very accurate ranging measurements if the clocks of the system are synchronised [Ledergerber et al., 2015]. The *TOF* methods can be used for passive localisation using the time difference of arrival method *TDOA* and for more robust active estimation using the time of arrival method*(TOA)*, both of which will be developed in this section.

Hardware

In the case of the *DW1000* chip, the only supported modes of ranging is the time of flight type methods [Decawave, 2014c], as the the transmission (*TX*) and receiving (*RX*) timestamps as computed internally in the chip. Therefore, we only consider localization by *TOA* and *TDOA* positioning in this section.

In an *UWB LPS* consisting of a network of *DW1000* chips, information is exchanged in accordance with the *IEEE* 802.15.4a standard [Group, 2004].By this standard, each encoded packet must consist of four distinct parts which will be described briefly for the sake of context and to enable a discussion on channel impulse response (*CIR*) in the noise modelling. The first two parts of the packet are the preamble sequence and the start of frame delimiter (*SFD*), both transmitted by emitting single pulses at a known time interval. This allows the coding of ternary sequences, whereby a positive impulse is denoted " + " , a negative pulse as " − " and an omitted pulse symbolises "0". The preamble is implemented to enable identification of the frequency band on

which the data is sent, and may be defined by as much as 4096 symbols. The much shorter *SFD*-sequence is species the end of the preamble sequence and is used to determine the precise time-stamps at which the packet arrives. The third part is the physical layer header (*PHR*), defining the rate and length at which the fourth part of the packet, the data field, is coded. While the binary coding of information is very interesting in itself, this section will primarily concern the preamble and how it is used to compute the time of flight.

Prior to this thesis, the *DW1000* chip had been implemented in an expansion board on the Crazyflie and in a set of radio anchors by Bitcraze AB, refer to [*Bitcraze UWB LPS*] for more details on the electronics and mechanics of the positioning system.

Measurement equations and limitations

Consider an unknown position, $\mathbf{p} \in \mathbb{R}^3$ $[m]$, of a chip (tag) capable of sending timestamped packets to a set of i other chips (anchors) at known fixed positions $\mathbf{p}_i \in \mathbb{R}^3$. Many problems become apparent when attempting to implement such a system. Firstly, the clocks in the robot and the anchors will not be perfectly synchronised, resulting in grave estimation errors. A temporary offset of nano seconds in the clocks will translate to an offset in meters in the distance as the packets are transferred at the speed of light, c. This problem is unavoidable, and a concrete example is the properties of the room temperature crystal oscillator (*RTXO*) in the *DWM* chips, which results in clock frequency variations on the order of ± 0.5 $[ppm]$ in the first minute after startup [Decawave, 2014b]. To cope with this issue, some method of clock synchronisation or estimate of clock difference, Θ_i, between the anchor i and the robot needs to be considered. Another problem is the stochastic measurement noise, which varies depending on the robots location relative the anchors due to the non-isotropic radiation patterns of the *DWM* chip [Decawave, 2014d]. However, for the sake of simplicity, we consider complete isotropy in the antenna and independent and identically distributed measurement noise $w_i \sim \mathcal{N}(0, {}^t\sigma_i^2)$. To get an idea of system performance, a standard deviation of ${}^t\sigma_i \approx 0.36 \cdot 10^{-10}$ $[s]$ was measured in a line-of-sight *TOA* measurement with the *DWM* chip at a range of 1.5 $[m]$. This corresponds roughly to the worst case ± 0.1 $[m]$ ranging accuracy guaranteed by the manufacturers of the chip [Decawave, 2014d]. Throughout this section, results will be presented with reference to a nominal and known anchor positions in $\mathbb{R}^3$ constituting a flyable space of 45 $[m^3]$ corresponding to a large room (see Table 6.1).

Time of arrival In the *TOA* method, we consider the measurement equation with respect to the time of flight t_i with respect to anchor i. Any time of flight measurement then yields a distance ranging through the relationship $d_i = ct_i$ $[m]$ with c $[m/s]$ denoting the speed of light. The measurement

Table 6.1 Nominal known anchor positions.

Anchor (i)	1	2	3	4	5	6
x_i $[m]$	-1.5	-1.5	-1.5	1.5	1.5	1.5
y_i $[m]$	-3	0	3	3	0	-3
z_i $[m]$	3	0	3	0	3	0
$c^t\sigma_i$ $[m]$	0.08	0.08	0.08	0.08	0.08	0.08

equation may be formulated as

$$\hat{t}_i = \frac{||\mathbf{p} - \mathbf{p}_i||_2}{c} - \Theta_i + w_i, \tag{6.38}$$

where it is evident that problem of variable clock rates will have a great effect on the measured time difference, $\hat{t}_i$. The possible values of solutions $\mathbf{p}$ of each measurement resides on a sphere centred around the $\mathbf{p}_i$ in $\mathbb{R}^3$. In this case we will either have to consider some means of multilateration to compute the intersecting points, or include the measurements directly in the dynamical estimator, as previously done in the work of Hamer [Mueller et al., 2015].

In the real-time implementation, communication between the robot and anchors is bidirectional for increased robustness with respect to constant clock offsets and linear clock drift. Details on the implemented symmetric double-sided two way ranging (*SDS-TWR*) protocol complete with an error analysis can be found in **Appendix C.2**. As the protocol is bidirectional, each anchor will be locked to a on pre-determined time intervals using time-division multiple access (*TDMA*) in a Round-Robin fashion [Chan, 2007]. Consequently, we note that a very a limited number of robots can be run in the same anchor system with this protocol in place. However, as the *SDS-TWR* protocol eliminates clock drift, the *TOA* information enter the *EKF* as $\mathbf{h}^{row}(\mathbf{x}, \mathbf{0}) = c\hat{t}_i \triangleq \hat{d}_i$, where then

$$\mathbf{H}^{row}(\mathbf{x}) = \frac{\partial \hat{d}_i}{\partial \mathbf{x}} = \frac{(x - x_i)\hat{x} + (y - y_i)\hat{y} + (z - z_i)\hat{z}}{||\mathbf{p} - \mathbf{p}||_2}, \tag{6.39}$$

where $\hat{y}$ denotes the position of the state y in the state vector $\mathbf{x}$.

Time difference of arrival In the *TDOA* case, we effectively compare two *TOA* measurements from a pair of anchors $\{i, j\}$, and the measurement equation becomes

$$\hat{t}_{i,j} = \hat{t}_i - \hat{t}_j = \frac{||\mathbf{p} - \mathbf{p}_i||_2 - ||\mathbf{p} - \mathbf{p}_j||_2}{c} - \Theta_i + \Theta_j + w_i - w_j. \tag{6.40}$$

Consider a set of anchors with perfectly synchronised clocks, which can be done in real-time by anchor-to-anchor communication using the *SDS-TWR*

protocol. Then $\Theta_i - \Theta_j = 0$, and there is no need for bidirectional communication with the robot. This allows unlimites swarms of *UAV*s to run in the same *LPS*, but naturally comes at the cost of a more complex measurement equation, which effectively can be seen as finding the intersection of a set of hyperbola in $\mathbb{R}^3$ (see Figure 6.5). Similarly to the *TOA* case, the measurement equation is written $\mathbf{h}^{row}(\mathbf{x}, \mathbf{0}) = c\hat{t}_{i,j} \triangleq \hat{d}_{i,j}$, forming the measurement Jacobian as in (6.41), with

$$\mathbf{H}^{row}(\mathbf{x}) = \frac{\partial \hat{d}_i}{\partial \mathbf{x}} - \frac{\partial \hat{d}_j}{\partial \mathbf{x}}. \tag{6.41}$$

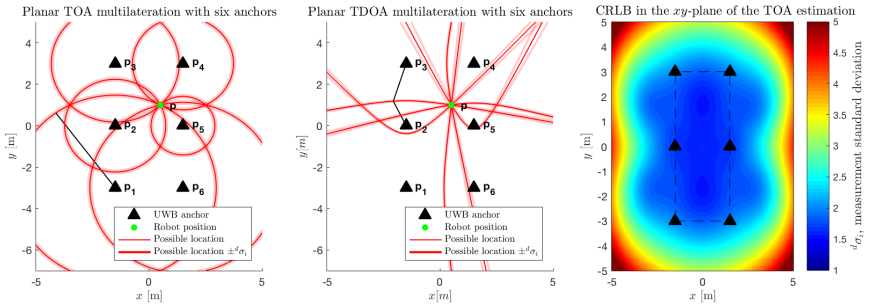

Figure 6.5 *Left.* Localisation of the Crazyflie (green) in the xy-plane with *TOA* measurements. *Center* Localisation of the Crazyflie (green) in the xy-plane with *TDOA* measurements. *Right:* The *CRLB* in the xy-plane for *TOA* assuming no clock drift at $z = 1.5$ $[m]$ in the nominal anchor positions where ${}^d\sigma_i = c\,{}^t\sigma_i \approx 0.08$ $[m]$.

Performance bounds The theoretical performance bounds of the considered *TOA* and *TDOA* methods have been investigated rigorously in previous work [Kaune, 2012] [Ledergerber et al., 2015] [Gentile et al., 2013]. The most common method is to evaluate the minimum variance of the positional estimate, $\hat{\mathbf{p}}$, in terms of the Cramer-Rao lower bound (*CRLB*), which is valid regardless of the estimator scheme and as long as the estimate is unbiased $\mathbb{E}[\hat{\mathbf{p}}] = \hat{\mathbf{p}}$. The *CRLB*, here derived in $\mathbb{R}^3$, can be written

$$\mathbb{E}[(\hat{\mathbf{p}} - \mathbf{p})(\hat{\mathbf{p}} - \mathbf{p})^T] \geq \mathbf{I}^{-1}(\mathbf{p}) \tag{6.42}$$

in which the variance of the estimate is bounded from below by the inverse of the Fisher Information Matrix (*FIM*), $\mathbf{I}(\mathbf{p})$ [Sengijpta, 1995] [Gentile et al.,

2013]. The *FIM* is given by

$$\mathbf{I}(\mathbf{p}) = \mathbb{E}\Big[\Big(\nabla_{\mathbf{p}} \ln\big(f(\hat{\mathbf{d}}|\mathbf{p})\big)\Big)^2\Big] \tag{6.43}$$

where $f(\hat{\mathbf{d}}|\mathbf{p})$ denotes the probability density function of the multivariate gaussian distribution of ranging measurements, conditioned by the true position of the robot $\mathbf{p}$ (see **Appendix C.1** for the complete derivation). Note that computing the bound for the *TOA* case only assumes knowledge of the anchor positions $\mathbf{p}_i$ and the standard deviations of the *TOA* measurements of each anchor ${}^t\sigma_i$. It can therefore easily be modified to study the effect of non-isotropic radiation patterns and anchor failures, which is left open for future research. Having determined this bound in the nominal anchor configuration, we have method of analysing the error induced by the *UWB* anchor position, which in our chosen configuration of anchors according to Table 6.1 yields reasonable positional estimate bounds within the convex hull of the anchors (see Figure 6.5). Here the *CRLB* is plotted in the xy-plane at $z = 1.5$, showing that the configuration will allow flight even outside of the convex hull of the anchors at a cost of a degraded estimate, with similar results $\forall z \in [0, 3]$ $[m]$. In the chosen configuration, the *CRLB* indicates that our best case positional estimate will yield standard deviations $\gtrsim 1.8 \cdot {}^d\sigma_i = 1.8c \cdot {}^t\sigma_i \approx 15$ $[cm]$, which will be our reference when evaluating the estimator performance.

Noise model

In order to discuss ways of modelling disturbances we must first consider the *SFD* sequence, $s[n]$, is used in the computation of time of flight measurement. Conforming with the *IEEE 802.15.4a* standard, $s[n]$ is chosen as one of two ternary Ipatov-type sequences containing $N \in \{8, 64\}$ symbols, where for the $N = 8$ case,

$$s[n] = \{0, +, 0, -, +, 0, 0, -\}. \tag{6.44}$$

This sequence has the property of being near perfectly periodically autocorrelative, which is of vital importance when trying to locate packets in an incoming signal $x[n]$ containing one or more sequences $s[n]$ corrupted by noise. Autocorrelation of the *SFD* sequence, $r_{ss}[l]$, can be evaluated in several metrics, such as the magnitude of Golay merit factor,

$$F_G = N^2\Big(2\sum_{l=1}^{N-1} r_{ss}^2[l]\Big)^{-1}, \tag{6.45}$$

a measure of how small the squared autocorrelation $r_{ss}^2[l] \;\; \forall l \neq 0$ is relative $r_{ss}^2[0]$. If the merit factor F_G is large, it is clear that applying a a simple cross-correlative convolution on $x[n]$ will yield peaks when the *SFD* sequence is encountered, have small side-lobes for $l \in [1 - N, N - 1] \backslash \{0\}$, and be close to

zero at all other times, as the correlation between $s[n]$ and a white random sequence is zero. For the $N = 8$ symbol ternary sequence defined in (6.44), $F_G = 21.3$, comparable to binary sequences which rarely exceed $F_G = 6$ for any choice of N [Jedwab et al., 2013].

The cross-correlation between the received signal and the *SFD* sequence at a time lag of l is then

$$r_{sx}[l] = (s * x)[l] = \sum_{n=-\infty}^{\infty} s^*[n]x[n-l] \tag{6.46}$$

yielding well defined peaks at delays, τ, corresponding to when a packet is received, of which the first forms the time of flight measurement $\hat{t}_i$ in the *TOA*/*TDOA* cases.

The time evolution of the cross correlation is commonly referred to as the channel impulse response (*CIR*), $C(t, \tau) \in \mathbb{C}$, [Walree, 2011] [Tse and Viswanath, 2005]. The *CIR* is computed in the *DW1000* chip and can be read from an accumulator memory where it is generated at rates between $294 - 5263$ $[Hz]$ depending on the preamble length, channel bandwidth and chosen *SFD* sequence [Decawave, 2014a] (see Figure 6.6). We have no way of knowing exactly how it is done due to intellectual property restrictions, but the working assumption is that a cross correlation similar to that of (6.46) is implemented.

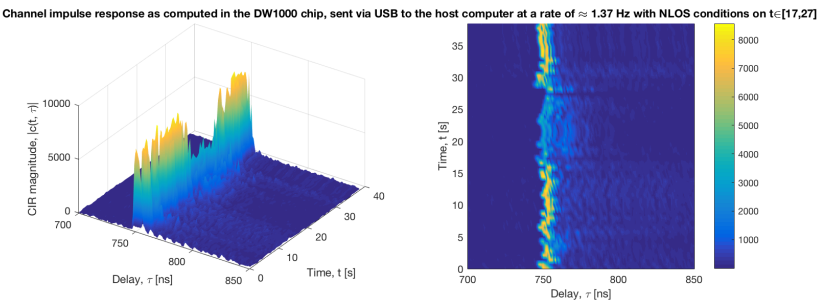

Figure 6.6 Measured channel impulse response when moving the out of the line of sight during $t \in [17, 27]$ $[s]$.

To find the peaks of the *CIR*, commonly referred to as multi-path components, the *DW1000* chip implements a Leading Edge (*LE*) algorithm due to it's simplicity. This algorithm typically defines two rectangular windowed

means

$$w_s[l] = \frac{1}{N_s} \sum_{i=0}^{N_s-1} |r_{sx}[l-i]|, \quad w_l[l] = \frac{1}{N_l} \sum_{i=0}^{N_l-1} |r_{sx}[l-i]|, \tag{6.47}$$

where the small and large windows $w_s[n]$ and $w_l[n]$ contain N_s and N_l measurement points respectively. The windowed means are run on the the *CIR* amplitude are compared in time, detecting a peak at the first instance when the conditions

$$w_s[l] - \epsilon_l w_l[l] > 0 \quad \wedge w_s[l] > \epsilon_t \tag{6.48}$$

are met for some constant $\epsilon_l < 1$ and ϵ_t relating to the noise floor (see Figure 6.7).

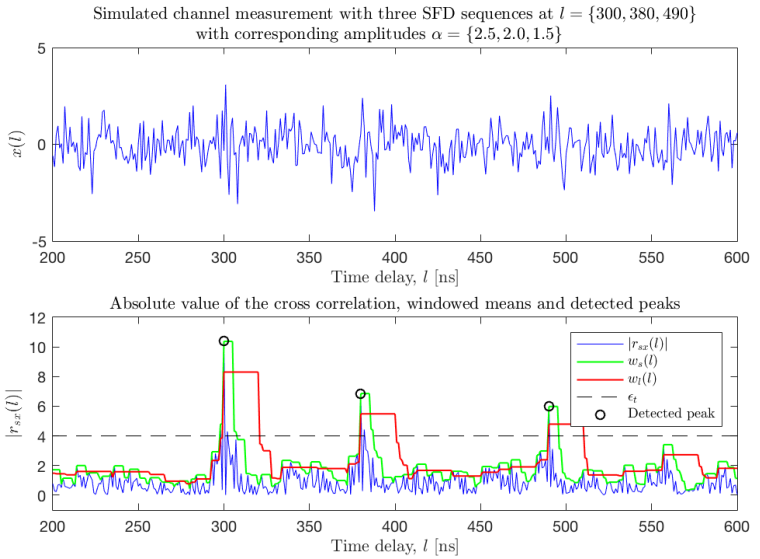

Figure 6.7 The leading edge algorithm applied to a the cross covariance (bottom) of a simulated channel measurement (top) including three *SFD* headers of varying amplitude and large amounts of white gaussian noise. In this demonstrative example, $\epsilon_l = 0.8$ and $\epsilon_t = 4$

With this background we proceed to model the measurement noise in a multi-path environment. If the $s[n]$ component in $x[n]$ is attenuated by a factor of $\alpha < 1$ from the nominal value $r_{sx}(0)$ due to reflections and distance

travelled, this will be visible in the amplitude of the cross-correlation, as for $y[n] = \alpha x[n]$, clearly

$$r_{sy} = \sum_{n=-\infty}^{\infty} s^*[n]\alpha y[n+l] = \alpha \sum_{n=-\infty}^{\infty} s^*[n]x[n+l] = \alpha r_{sx}[l]. \tag{6.49}$$

This is depicted seen in Figure 6.7, and the result allows the determination of accuracy in the ranging and the modelling of measurement noise in the dynamical filter. For this purpose we crudely assume that the *CIR* amplitude decreases proportionally to the distance travelled by the packet squared, $C(t, \hat{t}_i) \propto (c\hat{t}_i)^{-2}$. If so, the metric $\hat{d}_i^2 |C(t, \hat{t}_i)| \in \mathbb{R}$ will be be roughly constant regardless of the ranging time of flight in an echo free environment. However, every time the signal is reflected, some energy will be lost to the environment and we may therefore let the noise in the time of flight measurement of the *EKF* be written

$$\mathbf{h}^{row}(\mathbf{0}, \mathbf{v}) = \left(\alpha \hat{d}_i^2 |C(t, \hat{t}_i)|\right)^{-1} v_{tof}(t) \tag{6.50}$$

where the gain $\alpha = 10^{-4}$ is included to normalize the *CIR* amplitude and $v_{tof}(t) \sim \mathcal{N}(0, {}^d\sigma_i^2)$, determined to be ${}^d\sigma_i \approx 0.2$ [m] experimentally.

Interestingly, by applying a second *LE* algorithm to the *CIR*, significant multi-path components can be tracked in time, and their phase and amplitude could be used for simultaneous localisation and mapping *SLAM* [Kuang et al., 2013], potentially eliminating the need for multiple anchors. However, due to the low resolution of the *CIR* with each time delay corresponding to ≈ 1 $[ns]$, we can only detect multi-path components at distance intervals of $\approx 0.33m$, which is much more coarse than typical testbeds used in *UWB SLAM* research. In addition, computational constraints imposed by the *STM32* micro-controller makes meaningful multi-path component tracking impossible in the embedded real-time application. Better modelling of noise based on *CIR* and multi-path component tracking is left an open invitation to future research.

Real-time example

Due to a lack of ground truth data, the performance of the *CIR* compensated noise model in the *EKF* was only compared to the original implementation of Mike Hamer with respect to the estimated position relative the reference position. By circling a small wall of tables in a known elliptic orbit using the *SE(3)* control system, the anchors are intermittently be blocked by the walls, and multi-path components will be registered instead of the direct path, degrading the estimator performance (see Figure 6.8).

The experiment hints at a slight improvement in performance using the *CIR*-compensation, but nothing can be said with certainty except that the

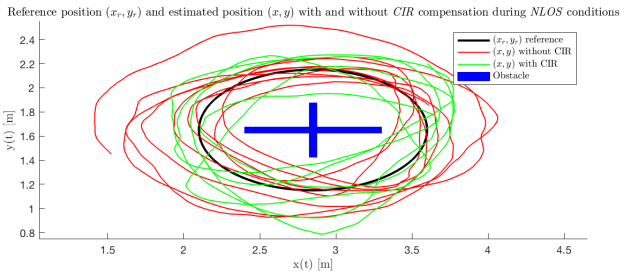

Figure 6.8 Positional estimation of the Crazyflie when following an elliptic orbit around in non-line-of-sight conditions with and without *CIR* multi-path compensation, the *UWB* measurements in *TOA*-form with gyroscopic and accelerometer measurements in the embedded scalar *EKF*.

method does not seem to improve the estimation significantly to warrant the additional computational resources required. Many reasons combine to explain this result. Firstly, there is redundancy in the number of anchors used on the nominal configuration. When determining the position, no more than three anchor-to-robot measurements are required, and we have as many reliable measurements at all times in the tested configuration. Secondly, the implemented relationship between the distance measurement and the *CIR* amplitude is a crude approximation of reality, and could be improved using results from *RSS* research [Srinivasa and Haenggi, 2009]. Thirdly, the *EKF* operates under the assumption of gaussian state distributions which is a poor approximation both with and without *CIR* compensation. Despite this, both the *CIR*-compensated and regular *EKF* are robust to a point where they may be used with confidence, as demonstrated in **Section 5**. In conclusion, no significant improvement was seen when using the *CIR*-compensation in the *TOA* ranging, but the use of multi-path components should be investigated further in order to increase positional accuracy in non-line-of-sight conditions.

6.5 Optical flow and laser positioning

While systems such as Vicon, Optitrack and Qualisys yield *mm*-precision accuracy at rates exceeding 100 $[Hz]$, the previously discussed *UWB LPS* with *SDS-TWR* protocol and *CIR*-compensation was shown to yield measurements with *dm*-precision at a fraction of the cost, but only supporting small numbers of quad-rotors with the time division multiple access (*TDMA*) technique. When instead using *TDOA* ranging, the *LPS* supports an arbitrary number of quad-rotors by eliminating the two-way communication.

However, the measurement accuracy suffers greatly with flight becoming near impossible outside of the convex hull of the anchors [Mueller et al., 2015].

The aforementioned methods have two common drawbacks, they require (i) external anchors and cameras to be set up and (ii) line of sight with the quadcopter, restricting flight to a certain visible volume in space. While self evident in the camera system, this was shown conclusively in the *UWB* case by studying the *CRLB* outside of the convex hull of the anchors. Consequently, in a practical application where the *UAV* is to perform well in a larger room such as a supermarket, alternate methods have to be considered. In this section, we develop the necessary framework for optical flow and laser estimation, with all sensory equipment fixed to the *UAV*. This allows for complete autonomy at a low monetary cost with no need for external systems, removing the upper bound on flyable volume and and enabling operation in non-line-of-sight conditions.

The term optical flow refers to a flow of two-dimensional images, in which certain features, such as patterns or pixel intensities, are tracked in time. This can be done in many ways, and a substantial body of research has dealt with creating robust methods of determining how images are translated [Horn and Schunck, 1981] and the methods' performance in terms of robustness and accuracy [Barron et al., 1994]. Our work is not concerned with how the pixel counts $(n_x, n_y) \in \mathbb{N}_0$ are derived, but rather how this information can be included in a Bayesian filter and fused with additional sensory information at variable rates.

Sensor preliminaries

Many interesting means of sensing optical flow exist, and we have been granted access to an alpha-prototype sensor which is small and looks to be a very affordable option. A non-disclosure agreement is currently in place regarding the sensor and the company producing it. As a consequence, the derived theory will therefore be presented in very general terms to avoid any possibility of infringement.

In any optical flow implementation, we presume to know the aperture angle of the camera$(\theta_{px}, \theta_{py})$ $[rad]$ and the number of pixels (N_x, N_y) $[pixels]$ of the image which is tracked in time. In addition, the accumulated pixel count $(\Delta n_x, \Delta n_y)$ $[pixels/s]$ is measured at small time steps Δt [s] as a measure of image translation in time (see Figure 6.9). We also assume knowledge of the rotation matrix estimate $\hat{\mathbf{R}}$ and elevation $\hat{z}$ $[m]$ from the state estimator. Finally, the rotational vector of the body frame $\hat{\boldsymbol{\omega}}_{\mathcal{B}} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$, is known directly from the gyroscope measurements.

For future reference, we consider three different coordinate systems in developing the theory, the first being the inertial system $O(\hat{\mathbf{x}}_{\mathcal{I}}, \hat{\mathbf{y}}_{\mathcal{I}}, \hat{\mathbf{z}}_{\mathcal{I}})$, the second being the body system $O(\hat{\mathbf{x}}_{\mathcal{B}}, \hat{\mathbf{y}}_{\mathcal{B}}, \hat{\mathbf{z}}_{\mathcal{B}})$ and finally the system of the

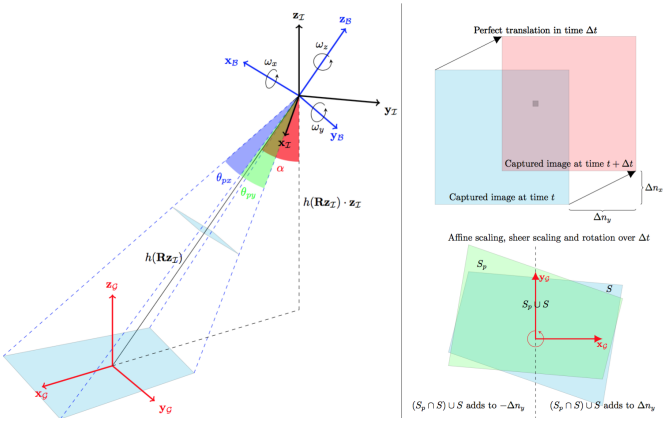

Figure 6.9 *Left*: Coordinate systems and relationships between angles used to define the measurement equations for the flow and *TOF* sensors. *Right*: Translated images over a time step of Δt $[s]$.

photographed image $O(\hat{\mathbf{x}}_\mathcal{P}, \hat{\mathbf{y}}_\mathcal{P}, \hat{\mathbf{z}}_\mathcal{P})$, where

$$\hat{\mathbf{x}}_\mathcal{P} = \frac{\hat{\mathbf{x}}_\mathcal{B} - (\hat{\mathbf{x}}_\mathcal{B} \cdot \hat{\mathbf{z}}_\mathcal{I})\hat{\mathbf{z}}_\mathcal{I}}{||\hat{\mathbf{x}}_\mathcal{B} - (\hat{\mathbf{x}}_\mathcal{B} \cdot \hat{\mathbf{z}}_\mathcal{I})\hat{\mathbf{z}}_\mathcal{I}||_2}, \quad \hat{\mathbf{z}}_\mathcal{P} || \hat{\mathbf{z}}_\mathcal{I}, \quad \hat{\mathbf{y}}_\mathcal{P} = \hat{\mathbf{z}}_\mathcal{P} \times \hat{\mathbf{x}}_\mathcal{P}. \tag{6.51}$$

The laser ranging is accomplished by means of the time-of-flight sensor, the *vl53l0x* [STMicroelectronics, 2014]. The vertical-cavity surface-emitting laser is the smallest on the global market at the time of writing, with a cost of $2.3 per unit. The sensor provides centimetre precision measurements at a range of $h < 1.2 - 2$ $[m]$ depending on the sample rate and mode of operation and the properties of the reflective surface. The angle of aperture of the collector is $\theta_{pz} = 25°$ and the distribution of the estimation error was experimentally shown to be well approximated by a normal distribution with a standard deviation which is a known function of the ranging distance [STMicroelectronics, 2014].

Hardware design

In order to run the *vl53l0x* sensor with the Crazyflie, a small printed circuit board (*PCB*) was created (see Figure 6.10). Designed so as not to interfere with the *SPI* communication of the *DW1000* chip, the expansion board enables the parallel operation of *UWB* and laser ranging. A driver was written to enable the long ranging mode of the laser ranging sensor at a rate of 100 Hz with outlier rejection. As faulty measurements in the laser ranging typically result in $h > 8$ $[m]$, only measurements $0 < h < 2$ $[m]$ were considered feasible.

In similar fashion, a *PCB* was developed to mount both the flow sensor and the *vl053x* simultaneously (see Figure 6.10). Due to conflicts in communication, this more extensive board cannot presently be run together with the *UWB*, but changes to the *SPI* communication could facilitate parallel use with the the UWB, laser and flow sensors. A driver was written to (i) sample the accumulated pixel counts at 100 $[Hz]$, (ii) rotate the accumulated pixel counts into the body frame, (iii) run a fast anti-aliasing filter on the measurements and (iv) perform outlier rejection. Bad measurements typically result in exact zeros or very large pixel counts, and similarly to the laser ranging, feasible measurements were considered to fulfill $0 < (\Delta n_x, \Delta n_y) < 100$ $[pixels]$.

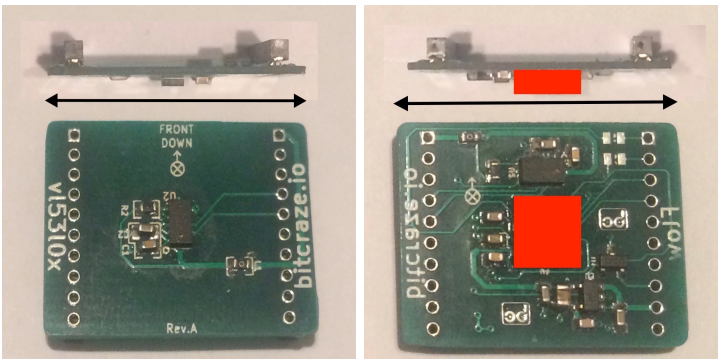

Figure 6.10 *Left*: *PCB* for laser ranging in the negative $\hat{\mathbf{z}}_{\mathcal{B}}$-direction using the *vl053x* sensor (black, center) with a 30 $[mm]$ arrow. *Right*: *PCB* for laser ranging (black, top) and optical flow (covered red, center) with a 30 $[mm]$ arrow.

Laser ranging measurement equations

To estimate the elevation, z, from the laser ranging measurement we first let $\alpha = \arccos[(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}}]$ (see Figure 6.9). If the laser sensor field of view $\theta_{pz} \to 0$, the resulting measurement becomes

$$z = h\mathbf{z}_{\mathcal{B}} \cdot \mathbf{z}_{\mathcal{I}} = h(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}} \Leftrightarrow h = \frac{z}{(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}}} = \frac{z}{\cos(\alpha)}. \tag{6.52}$$

However, if the field-of-view is significantly larger than zero, such as $\theta_{pz} = 25°$ given in the specifications of the *vl53l0x* sensor, it is clear that $h = z$ if $\alpha < \theta_{pz}/2$. When the $\hat{\mathbf{z}}_{\mathcal{B}}$ vector is no longer contained in the emitted laser cone, the measured distance h will correspond to the line on the measurement

cone closest to the axis $\hat{\mathbf{z}}_{\mathcal{B}}$,

$$\mathbf{h}^{row}(\mathbf{x}, \mathbf{0}) = h = \begin{cases} z & \text{if} \quad |\alpha| < \theta_{pz}/2 \\ \dfrac{z}{\cos(|\alpha| - \theta_{pz}/2)} & \text{if} \quad |\alpha| \geq \theta_{pz}/2 \end{cases}. \tag{6.53}$$

Determining the measurement Jacobian is trivial if disregarding changes in $\mathbf{R}$. For example, in a Tait-Bryan representation, $(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}} = \cos(\phi)\cos(\theta)$, showing that we can only correct the covariance with respect to pitch and roll, while the yaw remains unobservable. As rotations corresponding to pitch and roll are well determined from gyroscope and accelerometer information, the changes in $\mathbf{R}$ are omitted in the measurement Jacobian.

Optical flow measurement equations

In deriving the measurement equations of the flow sensor, we let (θ_x, θ_y) denote the angular offset about the $\hat{\mathbf{x}}_{\mathcal{B}}$- and $\hat{\mathbf{y}}_{\mathcal{B}}$-axis respectively, where then

$$\theta_x = \frac{\theta_{px}}{N_y} n_y, \qquad \theta_y = \frac{\theta_{py}}{N_x} n_x \tag{6.54}$$

As a first approximation, we let $\hat{\boldsymbol{\omega}}_{\mathcal{B}} = \mathbf{0}$ and $\hat{\mathbf{R}} = \mathbf{I}$, implying that the ground velocity can be written in terms of the accumulated pixel count as

$$\dot{x} = h\dot{\theta}_y \simeq h\frac{\Delta\theta_y}{\Delta t} = h\frac{\theta_{py}}{\Delta t N_x}\Delta n_x. \tag{6.55}$$

However, if we may presume to have some previous estimate of the system state and $\boldsymbol{\omega}_{\mathcal{B}} \neq \mathbf{0}$, it is clear that the rotations around the $\hat{\mathbf{x}}_{\mathcal{B}}$- and $\hat{\mathbf{y}}_{\mathcal{B}}$-axis will contribute to the optical flow. To minimise this effect during aggressive flight, we simply add a term compensating for the body rate

$$\dot{x} = h\frac{\theta_{py}}{\Delta t N_x}\Delta p_x - h\omega_{\mathcal{B},y}. \tag{6.56}$$

Finally, if in addition we let $\mathbf{R} \neq \mathbf{I}$, it is clear that the image taken may be subject to not only translation, but also scaling, which, combined with a rotation about $\hat{\mathbf{z}}_{\mathcal{B}}$ will contribute to the accumulated pixel count. Due to the symmetry of the image, this effect dependends greatly on the rotation about the $\hat{\mathbf{z}}_{\mathcal{B}}$-axis, and is therefore only relevant when $\omega_{\mathcal{B},z} \gg 0$. In this case, the true speed can be written

$$\dot{x} = h\frac{\theta_{py}}{\Delta t N_x}\Delta p_x - h\omega_{\mathcal{B},y} - P_x. \tag{6.57}$$

where P_x denotes the contribution to the pixel count caused by the rotation about the $\hat{\mathbf{z}}_{\mathcal{B}}$-axis and asymmetries in the image due to $\hat{\mathbf{R}} \neq \mathbf{I}$. An expression

for P_x can be derived considering the scaled image, S, in the $\hat{\mathbf{x}}_{\mathcal{P}}\hat{\mathbf{y}}_{\mathcal{P}}$-plane and its reflection in the $\hat{\mathbf{x}}_{\mathcal{P}}$- and $\hat{\mathbf{y}}_{\mathcal{P}}$-axis, S_p, from which it is easily verified that the surface $(S_p \cap S) \cup S$ contributes to the pixel count, and that this contribution depends on the rotation $\boldsymbol{\omega}_{\mathcal{B}}\hat{\mathbf{z}}_{\mathcal{B}}$ (see Figure 6.9). For simplicity, P_x is be assumed to be small, but if we consider flight with large values of ω_z, the term will have to be defined properly.

Letting $\mathbf{R} \neq \mathbf{I}$, $\boldsymbol{\omega}_{\mathcal{B}} \neq 0$ and considering small rotations $\omega_z \ll 1$ in the optical flow estimation, the non-linear measurement equation at the rows of indices m and $m+1$ may be written

$$\begin{aligned} \mathbf{h}^{row}(\mathbf{x}, \mathbf{0}) = \Delta n_x &= \frac{N_y \Delta t}{\theta_{py}} \Big(\frac{\dot{x}(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}}}{z} + \omega_y \Big) \\ \mathbf{h}^{row+1}(\mathbf{x}, \mathbf{0}) = \Delta n_y &= \frac{N_x \Delta t}{\theta_{px}} \Big(\frac{\dot{y}(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}}}{z} - \omega_x \Big) \end{aligned} \tag{6.58}$$

where the corresponding measurement Jacobian for Δn_x is

$$\mathbf{H}^{row}(\mathbf{x}) = \frac{\partial \Delta n_x}{\partial \mathbf{x}} = \frac{N_y \Delta t}{\theta_{py}} \Big[((\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}}) \Big(\frac{1}{z} \hat{\dot{x}} - \frac{\dot{x}}{z^2} \hat{z} \Big) + \hat{\omega}_y \Big] \tag{6.59}$$

where $\hat{\omega}_y$ denotes the position of the state ω_y in the state vector $\mathbf{x}$. A similar expression for the measurement Jacobian row corresponding to the pixel count Δn_y. As in the laser ranging, the term $(\mathbf{R}\mathbf{z}_{\mathcal{I}}) \cdot \mathbf{z}_{\mathcal{I}}$ gives no information on the rotation about the z-axis and can be disregarded to simplify the measurement Jacobian formulation.

Adaptive filtering for removal of positional drift

An inherent problem for optical flow estimation comes from the estimation of unobservable states, which will not converge to any meaningful solution as shown in **Section 6.2**. In the case of optical flow, this applies to the positional states where the internal global estimation of translation (x, y) will diverge from the true global position (x_t, y_t) given time such that $(\Delta x, \Delta y) = (x - x_t, y - y_t) \neq \mathbf{0}$. However, if this drift is small enough such that during a battery lifetime of T $[s]$, there clearly exists a circular bound

$$\sup_{t \in [0,T]} ||(\Delta x(t), \Delta y(t))||_2 < C \tag{6.60}$$

for some constant C. As such, at the end of every flight, we may control the quadcopter to a point (x_r, y_r) and be certain that it resides within a circle with radius C around the intended point. If we then introduce some known feature on the floor, the raw image taken by the optical flow sensor may be analysed to find the feature, determine $(\Delta x, \Delta y)$ and synchronise the *EKF* estimator such that $(x, y) \triangleq (x_r, y_r) + (\Delta x, \Delta y)$. Note that this

allows for complete independence from any external electrical system. The *UWB* anchors or *MOCAP* cameras could for instance be replaced by a small printable note of paper, placed at a known position on the ground.

For the purpose of synchronising the position, we consider a target feature consisting of concentric circles, which has been used succesfully in vision-based autonomous landing [Verbandt et al., 2014]. The target feature is defined by a set of M circles with radii $R = \{r_1 < \cdots < r_M\}$ $[m]$ and the image taken by the flow sensor is defined as $\mathcal{I}$. A pixel in this image is denoted $\mathcal{I}_{i,j}$, $i, j \in [1, N]^2$, assuming a square image with $N = N_x = N_y$. Similarly, we assume that the camera angle of aperture is the same in the x and y directions, with $\theta_{px} = \theta_{px}$. The idea is then to define a kernel, $\mathcal{K} \in \mathbb{N}^{N_k \times N_k}$, which represents the concentric circle feature in question, scaled to fit the taken image, such that resolution of the kernel decreases with the altitude. This can be accomplished by adjusting the kernel size with

$$N_k = \left\lceil \frac{N}{\theta_{px}} \arcsin\left(\frac{r_M}{z}\right) \right\rceil, \tag{6.61}$$

assuming the the angle of aperture is small and that any image is taken from a stable hovering state. Using the gaussian kernel, $\mathcal{F}$, defined in (6.22) we may form a lowpass filtered two-dimensional cross correlation

$$\mathcal{F} * \mathcal{K} * \mathcal{I} = \sum_{a=-\infty}^{\infty} \sum_{b=-\infty}^{\infty} (\mathcal{F} * \mathcal{K})_{a,b} \mathcal{I}_{i-a,j-b} \tag{6.62}$$

where, in the case of the concentric circles,

$$(\hat{i}, \hat{j}) = \underset{(i,j)\in[1,N]^2}{\arg\max} \, (\mathcal{F} * \mathcal{K} * \mathcal{I})_{i,j} \tag{6.63}$$

yields a good estimate of the centre of the concentric circles in the taken image. Computation of the shift from $(\Delta x, \Delta y)$ from the pixels $(\hat{i}, \hat{j})$ is then trivial when knowing the estimated elevation.

Measurement noise models

In the case of the laser ranging, it was experimentally verified that the measurement noise, $v_h(t) \sim \mathcal{N}(0, \sigma_h^2)$, was zero-mean and normally distributed. The variance of the noise depends on luminosity conditions (specifically in the infrared spectrum), the reflective properties of the surface and the ranging distance [STMicroelectronics, 2014]. In contrast, measurement noise of the optical flow measurements, $v_{\Delta x}(t) \sim \mathcal{N}(0, \sigma_{\Delta x}^2)$ and $v_{\Delta y}(t) \sim \mathcal{N}(0, \sigma_{\Delta y}^2)$ are more difficult to measure and model. Some information can be inferred from the flow sensor, which provides a greyscale factor $Q_g \in [0, Q_g^{max}]$ and an image quality factor $Q_i \in [0, Q_i^{max}]$, where a low greyscale factor and

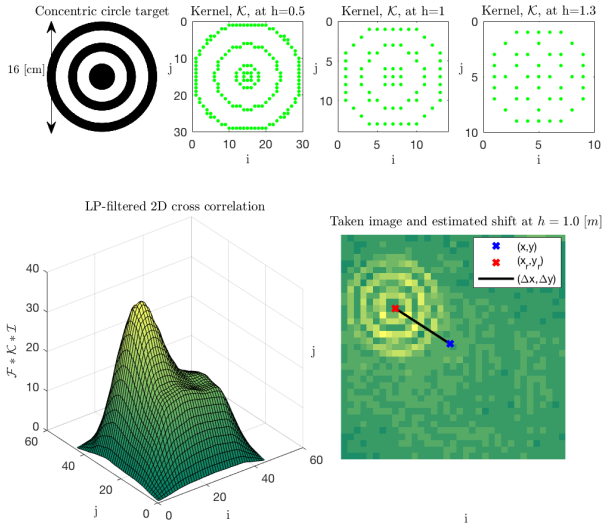

Figure 6.11 *Top*: Concentric circle target and filter kernels $\mathcal{K}$ at various elevations, h. *Bottom left* LP-filtered cross covariance of a taken image and a filter kernel, $\mathcal{F}*\mathcal{K}*\mathcal{I}$. *Bottom right* Taken image with estimated centre of the concentric circle feature from an elevation of $h = 1.0$ with a corresponding positional shift as computed in the *UAV* firmware yielding an error of single pixels, corresponding to an accuracy of ± 4 [cm].

high image quality indicate a lower standard deviation of the measurement noise. In addition, the flow sensor provides information on the shutter value $Q_s \in [0, Q_s^{max}]$ indicating the luminosity for the environment, which is of relevance to the laser ranging measurement.

Using the data from the flow sensor, we model the laser ranging measurement error as

$$\mathbf{h}^{row}(\mathbf{0}, \mathbf{v}) = \Big(1 + \alpha_h \frac{1 + Q_s + h - Q_i}{Q_s^{max} + h^{max} + Q_i^{max}}\Big) v_h(t) = \mathbf{V}_k^{row} v_h(t) \qquad (6.64)$$

where $\mathbf{V}_k^{row}$ is the diagonal element of the measurement noise Jacobian for the corresponding row, as the equation is linear in $v_h(t)$. Based on the sensor data sheet and manual tuning $\sigma_h = 0.003$ [m] and $\alpha_h \approx 3$ [STMicroelectronics, 2014], such that the maximum uncertainty at poor lighting, bad reflectance and high altitude becomes $10\sigma_h$. The m^{th} diagonal element of the measurement noise covariance matrix, $\mathbf{R}$, is then σ_h^2

In the optical flow we contend with only using the image factor

$$\begin{bmatrix} \mathbf{h}^{row}(\mathbf{0}, \mathbf{v}) \\ \mathbf{h}^{row+1}(\mathbf{0}, \mathbf{v}) \end{bmatrix} = \left(1 + \alpha_{\Delta n} \frac{Q_g}{Q_g^{max}}\right) \begin{bmatrix} v_{\Delta x}(t) \\ v_{\Delta y}(t) \end{bmatrix} \tag{6.65}$$

where the diagonal element of the measurement noise Jacobian is easily identifiable, the noise characteristics were experimentally determined to be $\sigma^2_{\Delta x} = \sigma^2_{\Delta y} \approx 0.3$ $[pixels/s]$ and $\alpha_{\Delta n} \approx 5/3$.

Real-time study

In order to demonstrate the implementation, the Crazyflie was flown over the course of 50 $[s]$ with PD control, optical flow, laser ranging and *IMU* fusion in the scalar update *EKF*. The beginning of the flight was controlled manually in order to show performance when flying carelessly and it's effects on the estimator covariance. The *UAV* was then set to fly autonomously with constant speed in the negative $\mathbf{x}_{\mathcal{B}}$-direction during 6 $[s]$ (blue) and then set to hover autonomously for 20 $[s]$ (green) before landing (see Figure 6.12).

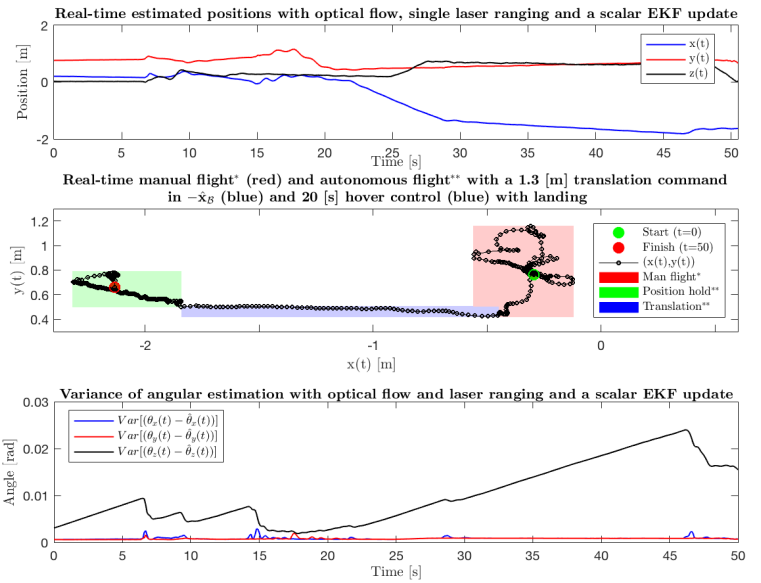

Figure 6.12 *Top*: Positional response during the experiment. *Center*: Travelled trajectory in the *xy*-plane. *Bottom* The estimator attitude variance during the experiment.

The scalar update *EKF* with is clearly capable of sustaining flight only using optical flow and laser measurements and the *IMU* sensor. Indeed, the altitude estimation is en par with z-estimation in the *UWB LPS* while in a stable hovering state. Both yield constant offsets in the decimetre range due to the noise caused by the motors, but the standard deviation of the elevation estimate estimate is typically $\sqrt{\mathbb{V}[\hat{z}]} \approx 5.2$ [cm] in the *UWB* case and $\sqrt{\mathbb{V}[\hat{z}]} \approx 1.6$ [cm] while hovering using the laser and optical flow sensor.

The biggest difference is in the positional xy-estimates, which are typically more accurate than then z-estimate in the *UWB* estimation. However, in the optical flow, the estimates exhibit a linear drifting behaviour on account of being unobservable in the measurement equations. As a consequence, the estimates will not converge to a meaningful solution, and drift in a linear fashion if attempting to stand perfectly still. This behaviour is expected from the discussion on observability, and shows that the measurements enter the *EKF* as intended (see **Section 6.2**). The drift effect could possibly be remedied by increasing the rate of the filter or by including additional flow sensors [Santamaria-Navarro et al., 2015]. However, a bigger concern is the rotation about the z-axis (yaw) becoming very uncertain in comparison to the rotation about the x- and y-axis (see Figure 6.12). This uncertainty grows significantly when the translational speeds, and implicitly angular speeds, are constant. In this state, the yaw becomes locally unobservable in the estimator.

The rotation matrix and specifically the yaw, ψ, is of vital importance in mapping the global position. For example, a constant offset of 10° during a 10 $[m]$ flight in the xy-plane results in positional errors up to 1.6$[m]$. On the firmware side, improvements of the z-axis rotation estimate can be made by including states for bias estimation in the accelerometer and gyroscope measurements [Sola, 2012] [Santamaria-Navarro et al., 2015] which should be investigated in future revision of the *EKF*. However, in general, applications with autonomous flight and flow sensing require means resetting estimates based on additional sensory information, in order to remove the drift in the positional states. With the current implementation, a practical solution is to place a small target on the ground and use the adaptive kernel filtering to estimate it's position relative the target. This target could be made of concentric circles to be more easily identifiable, allowing the resetting of translational x and y estimate. Alternatively, it could be made L-shaped, in which case the true yaw could be inferred by using a rotated kernel in the cross correlation.

While much of necessary algorithms have been implemented and verified in the practical implementation, the concept of drift removal by adaptive filtering needs to be investigated further, as it has the potential of enabling flight in an unbounded volume without external motion capture systems and with greater precision than the *UWB* estimates in the *TWR*-ranging mode.

7

Conclusions and summary

Over the course of this thesis, great progress has been made on developing a *UAV* platform for advanced research at *LTH* while simultaneously contributing useful code to Bitcraze AB. The main developments will here be summarised, indicating contributions to existing theory and work with a single dagger†, and brand new developments with a double dagger‡. Note that this thesis only deals with the theory and in particular the inner control system, referring the interested reader to [Greiff, 2017] for (i) the complete report with a discussion on outer $\mathcal{L}_1$ theory, time varying *LQR* and nonlinear *MPC* and to (ii) the current working implementation in the Robot Operating System (*ROS*). Furthermore, see [Greiff, 2016] for (iii) the motion capture with Kinect cameras as a implemented in a standalone *ROS* stack and (iv) [Hoenig et al., 2015] for the current working *ROS* driver, complete with contributions to facilitate communication of trajectories in specialised packets. The items (i)-(iv) were all developed from scratch or contributed to throughout the thesis, but are omitted due to a lack of space.

In the second chapter, the rotor dynamics were explored in the unconventional cross-configuration employed by Bitcraze†. In addition, the rigid-body dynamics were derived using the Euler-Lagrange equations with a ZYX Tait-Bryan parametrisation of rotation†. This approach was shown to suffer on account of the Gimbal lock phenomenon, and a remedy for the issue was presented guaranteeing well conditioned dynamics and controllability with a high probability in the complete attitude space‡. As an alternative, the equations of motion were derived using Newton-Euler equations with a quaternion representation of the rotation. The resulting model was shown to be less computationally complex than the Tait-Bryan alternative, omitting the need for evaluating trigonometric terms at the cost of diminished controllability†. The quaternion system was presented in its state-space form†, in an analytically derived linearised system‡and in a discrete time form taking the constant gravitational term into account†.

In the third chapter, algorithms were developed for motion planning by first deriving a heuristic solver for the travelling salesman problem to find

a close-to-optimal path through a set of points ordered by subsets with varying priority[‡]. Next, the notion of differential flatness was derived and implemented for the quaternion representation of rotation[†]. A method of low-pass smoothing was implemented for creating the necessary flat outputs from a sequence of points in flat output space[‡], and a method of quadratic-programming was explored to create minimum snap polynomial splines compliant with system dynamics[†]. In addition, methods of inflation and projection were developed to find the shortest linear distance around certain classes of obstacles[‡]and inequality constraints were explored to contain the splines generated by the quadratic program into safe a priori known sets[‡]. Functional implementations of the flatness and sequence generator were done in *C* and made part of the Crazyflie firmware[‡].

In the fourth chapter, a method of controlling the rotor loops independently of each other was explored with proportional-integral-derivative (*PID*) and model reference adaptive *MRAC* controllers, which both proved effective[†]. In addition, two means of parameter estimation were explored within the recursive least squares framework, providing a simple robust and computationally efficient method of determining the time varying parameters in the transfer function from voltage to rotor speed from estimating parameters in the transfer function from voltage to current[‡].

In the fifth chapter, the inner control system of the Crazyflie was developed to enable autonomous, high performance flight without relying on any external computer. This included investigating non-linear saturations to guarantee controllability even during complex manoeuvres[‡], the comparison of *LQR* and *PID* based approaches for attitude control[†], as well as the derivation of a 2-dof *PID* controller for tracking control[‡]. A novel method of integrating states in the *LQR* scheme was developed to handle changes in mass[‡], implementing conditional anti-windup. Finally, a promising geometrical SE(3) controller previously derived by [Lee et al., 2010] was implemented and verified in a Simulink environment[†]before being implemented in the Crazyflie firmware and demonstrated in several real-time experiments[‡].

In the sixth and final chapter, Madgwick's *AHRS* filter was presented for *IMU* based state estimation and implemented with bias compensation in the Crazyflie firmware[†]. In addition, the scalar update *EKF* was presented[†], evaluated in simulation and implemented in the firmware. Theory was developed to enable *MOCAP* estimation in the scalar update *EKF*, with systems including both depth sensing and non-depth sensing cameras[‡]. The pre-existing open-source code was augmented to support multi-path compensation in the *UWB LPS* case by using the *CIR*[‡]. Finally, theory necessary to run the *UAV* autonomously using optical flow[‡]and laser ranging[‡]was presented. The equations were implemented in the firmware, including a method of resetting the unobservable states with known features[‡]. The codes were developed to fuse all of the above methods for state estimation in the *UAV* firmware[‡].

Bibliography

Åström, K. J. and R. M. Murray (2010). *Feedback systems: an introduction for scientists and engineers*. (visited on 18-12-2016). Princeton university press.

Akgul, F. O. (2010). *Modeling the Behavior of Multipath Components Pertinent to Indoor Geolocation*. (visited on 18-10-2016). Worcester Polytechnic Institute.

Arulampalam, M. S., S. Maskell, N. Gordon, and T. Clapp (2002). "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking". *Signal Processing, IEEE Transactions on* **50**:2. (visited on 02-06-2016), pp. 174–188.

ASUS (2016). *Kinect Xtion PRO*. (visited on 17-06-2016). URL: `https://www.asus.com/3D-Sensor/Xtion_PRO/specifications/`.

Augugliaro, F., E. Zarfati, A. Mirjan, and R. D'Andrea (2015). "Knot-tying with Flying Machines for Aerial Construction". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. (visited on 23-07-2016). IEEE, Piscataway, NJ, pp. 5917–5922.

Baez, J. C. (2005). "On quaternions and octonions: their geometry, arithmetic, and symmetry by john h. conway and derek a. smith". *Bull. Amer. Math. Soc* **42**. (visited on 23-10-2016), pp. 229–243.

Balas, E. and P. Toth (1983). *Branch and bound methods for the traveling salesman problem*. Tech. rep. (visited on 02-10-2016). Management Science Research Report No. MSRR 488.

Bangura, M., M. Melega, R. Naldi, and R. Mahony (2016). "Aerodynamics of rotor blades for quadrotors". *arXiv preprint arXiv:1601.00733*. (visited on 15-07-2016).

Barron, J. L., D. J. Fleet, and S. S. Beauchemin (1994). "Performance of optical flow techniques". *International journal of computer vision* **12**:1. (visited on 14-12-2016), pp. 43–77.

Bitcraze. *Bitcraze uwb lps*. (visited on 16-08-2016). URL: `https://wiki.bitcraze.io/doc:lps:index`.

Blender Online Community (2016). *Blender - a 3D modelling and rendering package*. Blender Foundation. Blender Institute, Amsterdam. URL: `http://www.blender.org`.

Bullo, F. and A. D. Lewis (2004). *Geometric control of mechanical systems: modeling, analysis, and design for simple mechanical control systems*. Vol. 49. Springer Science & Business Media.

Castillo, P., R. Lozano, and A. Dzul (2004). "Stabilization of a mini-rotorcraft having four rotors." In: *IROS*. (visited on 23-10-2016), pp. 2693–2698.

Castillo, P., R. Lozano, and A. Dzul (2005). "Stabilization of a mini rotorcraft with four rotors". *IEEE control systems magazine* **25**:6. (visited on 08-07-2016), pp. 45–55.

Chan, T. S. (2007). "Time-division multiple access". *Handbook of Computer Networks: LANs, MANs, WANs, the Internet, and Global, Cellular, and Wireless Networks, Volume 2*, pp. 769–778.

Chen, T. and B. A. Francis (2012). *Optimal sampled-data control systems*. (visited on 06-17-2016). Springer Science & Business Media.

Chovancová, A., T. Fico, L. Chovanec, and P. Hubinsk (2014). "Mathematical modelling and parameter identification of quadrotor (a survey)". *Procedia Engineering* **96**. (visited on 08-07-2016), pp. 172–181.

Decawave (2014a). *Application note: aps006*. (visited on 22-10-2016). URL: `http://thetoolchain.com/mirror/dw1000/aps006_channel_effects_on_range_accuracy.pdf`.

Decawave (2014b). *Application note: aps011*. (visited on 22-10-2016). URL: `http://www.decawave.com/sites/default/files/resources/aps011_sources_of_error_in_twr.pdf`.

Decawave (2014c). *Application note: aps013*. (visited on 22-10-2016). URL: `http://thetoolchain.com/mirror/dw1000/aps013_dw1000_and_two_way_ranging.pdf`.

Decawave (2014d). *Dwm1000 datasheet*. (visited on 22-10-2016). URL: `http://www.decawave.com/sites/default/files/resources/dwm1000-datasheet-v1.3.pdf`.

Douc, R. and O. Cappé (2005). "Comparison of resampling schemes for particle filtering". In: *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*. (visited on 08-07-2016). IEEE, pp. 64–69.

Emami, S. (2013). *UWB Communication Systems: Conventional and 60 GHz*. Springer.

Fairchild Semiconductor Corporation (2000). *QRD1113/1114, Reflective Object Sensor*. (visited on 05-12-2016). URL: `http://solarbotics.net/library/datasheets/QRD1114.pdf`.

Fliess, M, J Levine, P Martin, and P Rouchon (1992). "On differentially flat nonlinear systems". In: *IFAC SYMPOSIA SERIES*. (visited on 17-09-2016), pp. 159–163.

Fliess, M., J. Lévine, P. Martin, and P. Rouchon (1999). "A Lie-Bäcklund approach to equivalence and flatness of nonlinear systems". *IEEE Transactions on automatic control* **44**:5. (visited on 17-09-2016), pp. 922–937.

Fresk, E. and G. Nikolakopoulos (2013). "Full quaternion based attitude control for a quadrotor". In: *2013 European Control Conference (ECC), July*. (visited on 23-10-2016), pp. 17–19.

Gentile, C., N. Alsindi, R. Raulefs, and C. Teolis (2013). "Ranging and localization in harsh multipath environments". In: *Geolocation Techniques*. (visited on 23-10-2016). Springer, pp. 17–57.

Glad, T. and L. Ljung (2000). *Control theory*. (visited on 01-06-2016). CRC press.

Goemans, M. X. and D. J. Bertsimas (1991). "Probabilistic analysis of the held and karp lower bound for the euclidean traveling salesman problem". *Mathematics of operations research* **16**:1. (visited on 02-10-2016), pp. 72–89.

Greiff, M. (2016). *Kinect vision project*. (visited on 12-06-2016). URL: `https://github.com/mgreiff/kinect_vision`.

Greiff, M. (2017). *The crazyflie project*. (visited on 06-06-2016). URL: `https://github.com/mgreiff/crazyflie\_project`.

Group, I. W. et al. (2004). "IEEE standard for local and metropolitan area networks. part 16: air interface for fixed broadband wireless access systems". *IEEE Std* **802**. (visited on 23-10-2016), pp. 16–2004.

Hoenig, W., C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian (2015). "Mixed reality for robotics". In: *IEEE/RSJ Intl Conf. Intelligent Robots and Systems*. (visited on 23-10-2016). Hamburg, Germany, pp. 5382 – 5387.

Hol, J. D., T. B. Schon, and F. Gustafsson (2006). "On resampling algorithms for particle filters". In: *Nonlinear Statistical Signal Processing Workshop, 2006 IEEE*. (visited on 08-07-2016). IEEE, pp. 79–82.

Horn, B. K. and B. G. Schunck (1981). "Determining optical flow". *Artificial intelligence* **17**:1-3. (visited on 01-12-2016), pp. 185–203.

Huxel, P. J. and R. H. Bishop (2009). "Navigation algorithms and observability analysis for formation flying missions". *Journal of guidance, control, and dynamics* **32**:4. (visited on 28-09-2016), pp. 1218–1231.

InvenSense (2014). *Mpu-9250 product specification*. (visited on 05-12-2016). URL: `https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU9250REV1.0.pdf`.

Iwasaki, T. and S. Hara (2005). "Generalized KYP lemma: unified frequency domain inequalities with design applications". *IEEE Transactions on Automatic Control* **50**:1. (visited on 21-12-2016), pp. 41–59.

Jedwab, J., D. J. Katz, and K.-U. Schmidt (2013). "Advances in the merit factor problem for binary sequences". *Journal of Combinatorial Theory, Series A* **120**:4. (visited on 25-09-2016), pp. 882–906.

Jiang, Y. and V. C. Leung (2007). "An asymmetric double sided two-way ranging for crystal offset". In: *2007 International Symposium on Signals, Systems and Electronics*. (visited on 23-10-2016). IEEE, pp. 525–528.

Jolly, K., R. S. Kumar, and R Vijayakumar (2009). "A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits". *Robotics and Autonomous Systems* **57**:1. (visited on 19-12-2016), pp. 23–33.

Kaune, R. (2012). "Accuracy studies for TDOA and TOA localization". In: *Information Fusion (FUSION), 2012 15th International Conference on*. (visited on 22-10-2016). IEEE, pp. 408–415.

Kim, H. (2009). "Performance analysis of the sds-twr-ma algorithm". In: *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*. (visited on 25-09-2016). ACM, pp. 399–403.

Kuang, Y., K. Åström, and F. Tufvesson (2013). "Single antenna anchor-free uwb positioning based on multipath propagation". In: *2013 IEEE International Conference on Communications (ICC)*. (visited on 16-10-2016). IEEE, pp. 5814–5818.

Landry, B. et al. (2015). *Planning and control for quadrotor flight through cluttered environments*. (visited on 13-06-2016). PhD thesis. Massachusetts Institute of Technology.

Ledergerber, A., M. Hamer, and R. D'Andrea (2015). "A robot self-localization system using one-way ultra-wideband communication". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. (visited on 08-07-2016). IEEE, pp. 3131–3137.

Lee, T., M. Leoky, and N. H. McClamroch (2010). "Geometric tracking control of a quadrotor UAV on SE (3)". In: *49th IEEE conference on decision and control (CDC)*. (visited on 07-11-2016). IEEE, pp. 5420–5425.

Lepetit, V. and P. Fua (2005). *Monocular model-based 3D tracking of rigid objects*. (visited on 23-10-2016). Now Publishers Inc.

Luukkonen, T. (2011). "Modelling and control of quadcopter". *Independent research project in applied mathematics, Espoo*. (visited on 16-6-2016).

Madgwick, S. O., A. J. Harrison, and R. Vaidyanathan (2011). "Estimation of imu and marg orientation using a gradient descent algorithm". In: *2011 IEEE International Conference on Rehabilitation Robotics*. (visited on 06-10-2016). IEEE, pp. 1–7.

Mellinger, D. and V. Kumar (2011). "Minimum snap trajectory generation and control for quadrotors". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. (visited on 07-29-2016). IEEE, pp. 2520–2525.

Moscato, P. and M. G. Norman (1992). "A memetic approach for the traveling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems". *Parallel computing and transputer applications* **1**. (visited on 02-10-2016), pp. 177–186.

Mueller, M. W. (2016). *Increased autonomy for quadrocopter systems: trajectory generation, fail-safe strategies, and state-estimation*. (visited on 16-06-2016). PhD thesis. ETH Zurich. DOI: 10.3929/ethz-a-010655275.

Mueller, M. W., M. Hamer, and R. D'Andrea (2015). "Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. (visited on 16-06-2016), pp. 1730–1736. DOI: 10.1109/ICRA.2015.7139421.

Mueller, M. W., M. Hehn, and R. D'Andrea (2016). "Covariance correction step for kalman filtering with an attitude". *Journal of Guidance, Control, and Dynamics*. (visited on 12-01-2017), pp. 1–7.

Murray, R. M., M. Rathinam, and W. Sluis (1995). "Differential flatness of mechanical control systems: a catalog of prototype systems". In: *ASME International Mechanical Engineering Congress and Exposition*. (visited on 01-11-2016). Citeseer.

Noraini, M. R. and J. Geraghty (2011). "Genetic algorithm performance with different selection strategies in solving TSP". *International Conference of Computational Intelligence and Intelligent Systems*. (visited on 02-10-2016).

Padberg, M. and G. Rinaldi (1987). "Optimization of a 532-city symmetric traveling salesman problem by branch and cut". *Operations Research Letters* **6**:1. (visited on 02-10-2016), pp. 1–7.

Pahlavan, K., X. Li, and J.-P. Makela (2002). "Indoor geolocation science and technology". *IEEE Communications Magazine* **40**:2. (visited on 18-10-2016), pp. 112–118.

Palais, B. and R. Palais (2007). "Euler's fixed point theorem: the axis of a rotation". *Journal of Fixed Point Theory and Applications* **2**:2. (visited on 08-07-2016), pp. 215–220.

Powers, C., D. Mellinger, A. Kushleyev, B. Kothmann, and V. Kumar (2013). "Influence of aerodynamics and proximity effects in quadrotor flight". In: *Experimental Robotics*. (visited on 13-11-2016). Springer, pp. 289–302.

Raffo, G. V., M. G. Ortega, and F. R. Rubio (2010). "An integral predictive/nonlinear H-infinity control structure for a quadrotor helicopter". *Automatica* **46**:1. (visited on 08-07-2016), pp. 29–39.

Reist, P. and R. Tedrake (2010). "Simulation-based LQR-trees with input and state constraints". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. (visited on 17-06-2016). IEEE, pp. 5504–5510.

Richter, C., A. Bry, and N. Roy (2013). "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments". In: *Proceedings of the International Symposium on Robotics Research (ISRR)*. (visited on 06-17-2016).

Santamaria-Navarro, A., J. Sola, and J. Andrade-Cetto (2015). "High-frequency mav state estimation using low-cost inertial and optical flow measurement units". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. (visited on 14-12-2016). IEEE, pp. 1864–1871.

Sato, K. (2014). "Algebraic controllability of nonlinear mechanical control systems". *SICE Journal of Control, Measurement, and System Integration* **7**:4. (visited on 28-09-2016), pp. 191–198.

Sengijpta, S. K. (1995). "Fundamentals of statistical signal processing: estimation theory". *Technometrics* **37**:4. (visited on 04-11-2016), pp. 465–466.

Sola, J. (2012). "Quaternion kinematics for the error-state KF". *Laboratoire d'Analyse et d'Architecture des Systemes-Centre national de la recherche scientifique (LAAS-CNRS), Toulouse, France, Tech. Rep*. (visited on 27-09-2016).

Srinivasa, S. and M. Haenggi (2009). "Path loss exponent estimation in large wireless networks". In: *Information Theory and Applications Workshop, 2009*. IEEE, pp. 124–129.

STMicroelectronics (2014). *Datasheet - production data*. (visited on 25-11-2016). URL: `http://www.st.com/content/ccc/resource/technical/document/datasheet/en.DM00279086.pdf`.

Tedrake, R. (2009). "LQR-Trees: Feedback motion planning on sparse randomized trees". In: *Robotics: Science and Systems*. (visited on 23-08-2016). Seattle, WA.

Terejanu, G. A. (2008). "Extended kalman filter tutorial". *Online]. Disponible: http://users. ices. utexas. edu/˜ terejanu/files/tutorialEKF. pdf*. (visited on 09-07-2016).

Trefethen, L. N. and D. Bau III (1997). *Numerical linear algebra*. Vol. 50. (visited on 20-11-2016). Siam.

Tse, D. and P. Viswanath (2005). *Fundamentals of wireless communication*. (visited on 23-10-2016). Cambridge university press.

Tully Foote Eitan Marder-Eppstein, W. M. (2016). *Quaternion class reference*. (visited on 23-10-2016). URL: `http://docs.ros.org/jade/api/tf/html/c++/classtf_1_1Quaternion.html`.

Unser, M. (2000). "Sampling-50 years after shannon". *Proceedings of the IEEE* **88**:4, pp. 569–587.

Verbandt, M., B. Theys, and J. De Schutter (2014). "Robust marker-tracking system for vision-based autonomous landing of vtol uavs". In: *Proceedings of the International Micro Air Vehicle Conference and Competition 2014*. Delft University of Technology, pp. 84–91.

Vilfan, B. (1973). "Another proof of the two-dimensional Cayley–Hamilton theorem". *IEEE Trans. Comput* **22**:12. (visited on 07-10-2016), p. 1140.

Walree, P. van (2011). "Channel sounding for acoustic communications: techniques and shallow-water examples". *Norwegian Defence Research Establishment (FFI), Tech. Rep. FFI-rapport* **7**. (visited on 20-10-2016).

Wan, E. A. and R. Van Der Merwe (2000). "The unscented Kalman filter for nonlinear estimation". In: *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*. (visited on 08-07-2016). Ieee, pp. 153–158.

Yin, F., C. Fritsche, F. Gustafsson, and A. M. Zoubir (2013). "Toa-based robust wireless geolocation and cramér-rao lower bound analysis in harsh los/nlos environments". *IEEE transactions on signal processing* **61**:9, pp. 2243–2255.

Young, H. D., R. A. Freedman, and L. Ford (2007). *University Physics Vol 2 (Chapters 21-37)*. Vol. 2. (visited on 17-06-2016). Pearson education.

Zhang, W. and R. E. Korf (1996). "A study of complexity transitions on the asymmetric traveling salesman problem". *Artificial Intelligence* **81**:1. (visited on 02-10-2016), pp. 223–239.

A

Modelling appendix

A.1 Identification of mappings

In previous work done at Bitcraze, an expansion board was created for measuring the quadcopter rotor speeds using the reflective object sensor QRD1114 [Fairchild Semiconductor Corporation, 2000], allowing the measurement of rotor speeds mid-flight. With this tool, the mappings from rotor speeds Ω_i $[rad/s]$ to thrust T $[N]$, and PWM duty cycle, $d \in [0, 1]$ were determined.

The UAV was placed upside down on a scale, and the duty cycle was increased linearly across all motors, logging the rotor speeds and thrusts as read from the scale. A quadratic relationship is expected between rotor speeds, consequently, the regression model

$$T(x) = \beta_2 x^2 + \beta_1 x + \beta_0 + \epsilon. \tag{A.1}$$

was applied to the data, with β_i denoting parameters and ϵ denoting the error terms.

As $d = 0 \Rightarrow \Omega = 0 \Rightarrow T = 0$, we enforce $\beta_0 = 0$. Furthermore, the approximation $\beta_1 = 0$ is commonly used in UAV modelling of the rotor speed to thrust ratio, validated by the fact that any contribution from the β_1, β_0 parameters is very small judging by the small change in the mean squared error, especially close to the operating point where Ω_i is of the order 10^3 $[rad/s]$ (see Table A.1 and Figure A.1).

The full polynomial fit yields a marginally smaller mean squared error (MSE) when compared to the fitting with $\beta_1 = \beta_0 = 0$. However, both are deemed good and the latter is used, as a purely quadratic relationship is assumed in the simplified quadcopter model. With $\beta_1 = \beta_0 = 0$ we find that $k = \beta_2/4 \approx 2.2 \cdot 10^{-8}$ by equation (2.9) under the assumption that $k_i = k \forall i$. When flying the quad-rotor with the small expansion board attached and a close to full battery, it is clear that a hovering state is maintained at a rotor speed of $\Omega_h \approx 1.8 \cdot 10^4$ $[RPM] \approx 1.88 \cdot 10^3 [rad/s]$. Knowing that the quadcopter weighs $m = 0.027$ $[kg]$ and the expansion board weighs approximately

Table A.1 The identified mapping from rotor speed $[rad/s]$ and PWM duty cycle $d \in [0, 1]$ to thrust $[kg \cdot m/s^2]$ when including and excluding the linear and constant relationships.

Mapping	β_0	β_1	β_2	MSE$(T - \hat{T}(\cdot))$
$T(\Omega)$	$1.51 \cdot 10^{-3}$	$-1.97 \cdot 10^{-7}$	$9.78 \cdot 10^{-8}$	$1.2861 \cdot 10^{-5}$
$T(\Omega)$	0	0	$8.87 \cdot 10^{-8}$	$5.46 \cdot 10^{-5}$
$T(d)$	0	0.35	0.26	$1.36 \cdot 10^{-5}$

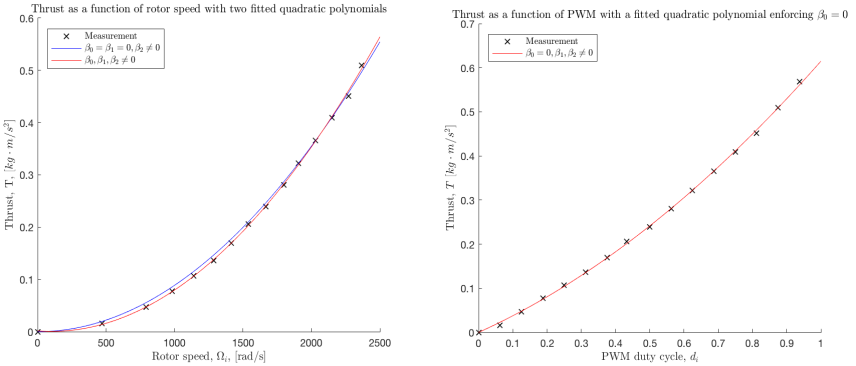

Figure A.1 *Left:* The rotor speed to thrust correspondence fitted with the quadratic polynomials$\beta_2 \neq 0$ (blue) and $\beta_0, \beta_1, \beta_2 \neq 0$ (red). *Right:* The rotor PWM to thrust space fitted with a quadratic polynomial with $\beta_0 = 0$.

$m_b \approx 0.005$ $[kg]$, the thrust generated by one rotor in a hovering state should correspond to

$$T_h = 4k\Omega_h^2 \Rightarrow k \approx \frac{(m + m_b)g}{4\Omega_h^2} = \frac{0.032 \cdot 9.81}{4 \cdot 1880^2} = 2.22 \cdot 10^{-8} \qquad [kg \cdot m/rad^2] \tag{A.2}$$

which is close to the previously estimated rotor speed to thrust ratio. Thereby giving additional support for the determined rotor speed to thrust ratio of $k \approx 2.2 \cdot 10^{-8}$ $[kg \cdot m/rad^2]$.

We may also perform an experiment to determine the b parameter, pertaining to how the torque about the $\hat{\mathbf{z}}_{\mathcal{B}}$-axis is governed (2.9). Consider an experimental setup where the *UAV* is suspended in mid-air by two strings parallel to the z-axis, such that it is free to rotate around the z-axis but all other modes of rotation or movement is prohibited. A string is then attached to any motor axis, held orthogonal to the vector between the centre of mass

and the motor axis in the xy-plane, $\mathbf{u}_{xy}$. The force in the string generated by the rotation of the quadcopter then satisfies

$$|\tau_\psi| = |\mathbf{f}_s \times \mathbf{v}_{xy}| = |\mathbf{f}_s| \cdot l. \tag{A.3}$$

If we further assume that $b_i = b \forall i$ and that at the time of measuring $\dot{\Omega}_i \equiv 0$, then (2.9) yields

$$\tau_\psi = \sum_{i=1}^{4} \tau_{M_i} = b(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \Rightarrow |b| = \frac{|\mathbf{f}_s| \cdot l}{|-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2|} \tag{A.4}$$

Knowing that $l = 0.046$ $[m]$ makes the identification of b simple if measuring the force in the string, $|\mathbf{f}_s|$, at known rotor speeds Ω_i. By this method, the parameter was estimated to be $b \approx 10^{-9}$, on a magnitude similar to k.

A.2 Continuous time DC motor parameters

Consider a continuous time system defined by (2.41) and (2.42), excluding the redundant angular positional state, $\theta(t)$. The continuous time system is then

$$\mathbf{x} = \begin{bmatrix} \Omega(t) \\ i(t) \end{bmatrix}, \quad \mathbf{u}(t) = U(t), \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} -b/J^{\pm} & K_t/J^{\pm} \\ -K_e/L & -R/L \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} \mathbf{u}(t) \tag{A.5}$$

Its discrete time equivalent, by any method of discretisation preserving the order of the characteristic polynomial, is then on the form

$$\begin{cases} \mathbf{x}_k = \mathbf{x}(hk) \\ \mathbf{u}_k = \mathbf{u}(hk) \end{cases}, \quad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix} \tag{A.6}$$

such that $\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k$. The two transfer functions from the applied voltage to rotor speeds and current may be written

$$\begin{aligned} \begin{bmatrix} H_{U\to\Omega}(z) \\ H_{U\to i}(z) = \end{bmatrix} =& (\mathbf{I}z - \mathbf{A})^{-1}\mathbf{B} = \\ =& \frac{1}{(z-a_{11})(z-a_{22}) - a_{12}a_{21}} \begin{bmatrix} (z-a_{22})b_{11} + a_{12}b_{21} \\ a_{21}b_{11} + (z-a_{11})b_{21} \end{bmatrix} \\ =& \frac{1}{a_2 z^2 + a_1 z + a_0} \begin{bmatrix} z b_1^\Omega + b_0^\Omega \\ z b_1^i + b_0^i \end{bmatrix}. \end{aligned} \tag{A.7}$$

Both pulse transfer functions from an input signal u to y both satisfy

$$H_{u\to y}(z) = \frac{B^{(y)}(z)}{A(z)}, \quad \deg(B^{(y)}(z)) = 1, \ \ \deg(A(z)) = 2 \tag{A.8}$$

Now, if $A(z)$ is monic, we may find three coefficients for each of the two transfer functions in (A.7), allowing the identification of all parameters in the discrete time system (A.6) through diophantine equation

$$\begin{cases} a_{11} &= -a_1 - a_{22} \\ a_{12} &= -\dfrac{a_0 + (a_1 + a_{22})a_{22}}{a_{21}} \\ a_{21} &= \dfrac{b_0^i - (a_1 + a_{22})b_{21}}{b_{11}} \\ a_{22} &= \dfrac{a_0 b_{11} b_{21} + b_0^\Omega b_0^i - b_0^\Omega a_1 b_{21}}{b_0^\Omega b_{21} - b_0^i b_{11}} \end{cases}, \qquad \begin{cases} b_{11} &= b_1^\Omega \\ b_{21} &= b_1^i \end{cases}. \tag{A.9}$$

Let

$$A(z) = z^2 + a_1 z + a_0, \quad B^{(y)}(z) = b_1^{(y)} z + b_0^{(y)} \tag{A.10}$$

and consider the linear regression model

$$y_k = \varphi_k^T \theta + e_k \tag{A.11}$$

where the e_k is uncorrelated white gaussian noise and

$$\varphi_k^T = \begin{bmatrix} -y_{k-1} & -y_{k-2} & u_{k-1} & u_{k-2} \end{bmatrix}, \quad \theta = \begin{bmatrix} a_1 & a_0 & b_1^{(y)} & b_0^{(y)} \end{bmatrix} \tag{A.12}$$

with a system of N measurement points, the best parameter vector estimate in a least squares sense may be written

$$\hat{\theta} = (\Phi^T \Phi)^{-1} \Phi^T Y \tag{A.13}$$

where

$$\Phi^T = \begin{bmatrix} \varphi_3 & \cdots & \varphi_N \end{bmatrix}, \quad Y^T = \begin{bmatrix} y_3 & \cdots & y_N \end{bmatrix} \tag{A.14}$$

With this method for identifying the discrete time transfer functions of the system, the continuous time parameters may be found through an inverse discretisation. However, due to the continuous time model having six parameters and only five non-zero entries, some assumption regarding the mechanical equation parameters is necessary. To demonstrate the implementation, we assume that $K_e = K_t$ use the zero-order-hold discretisation and simulate a rotor model $L = 6.0$, $R = 5.0$, $Ke = 4.0$, $Kt = 4.0$, $J = 2.0$, $b = 3.0$ with a persistently exciting sinusoid input signal corrupted by white gaussian noise. From the resulting time series $\{\Omega_k\}, \{i_k\}, \{U_k\}$, the correct continuous time parameters were identified using the above method with an estimator error of $||\theta - \hat{\theta}||_2 = 1.51 \cdot 10^{-11}$. Having verified the functionality of the LS scheme, the algorithm was run on experimental rotor data, yielding the estimated rotor model parameters in Table A.2

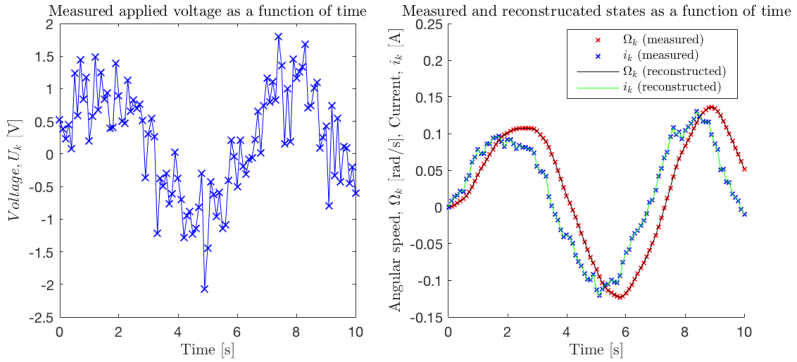

Figure A.2 Open loop simulated rotor states and control signals, and simulation of the identified system from using the same initial conditions.

Table A.2 Continuous time rotor model parameters.

Parameter	R	J^+	J^-	b	K_t	K_e	L
Value	2.3	0.031	0.13	0.10	580	0.0011	0.12

A.3 Coriolis matrix definition

Let c_{ij} denote the element at row i and column j of the matrix $\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})$, where $c_a = \cos(a), s_a = \sin(a)$ and I_{ii} denotes the i^{th} diagonal element of the diagonal moment of inertia tensor. Then

$$
\begin{aligned}
c_{11} =& 0 \\
c_{12} =& (I_{22} - I_{33})(\dot{\theta} c_\phi s_\phi + \dot{\psi} c_\theta (s_\phi^2 - c_\phi^2)) - I_{11} \dot{\psi} c_\theta \\
c_{13} =& (I_{33} - I_{22}) \dot{\psi} c_\phi s_\phi c_\theta^2 \\
c_{21} =& (I_{33} - I_{22})(\dot{\theta} c_\phi s_\phi + \dot{\psi} c_\theta (s_\phi^2 - c_\phi^2)) + I_{11} \dot{\psi} c_\theta \\
c_{22} =& (I_{33} - I_{22}) \dot{\phi} c_\phi s_\phi \\
c_{23} =& (-I_{11} \dot{\psi} + I_{22} \dot{\psi} s_\phi^2 + I_{33} \dot{\psi} c_\phi^2) s_\theta c_\theta \\
c_{31} =& (I_{22} - I_{33}) \dot{\psi} c_\theta^2 s_\phi c_\phi - I_{11} \dot{\theta} c_\theta \\
c_{32} =& (I_{33} - I_{22})(\dot{\theta} c_\phi s_\phi s_\theta + \dot{\phi} c_\theta (s_\phi^2 - c_\phi^2)) + (I_{11} - I_{22} s_\phi^2 - I_{33} c_\phi^2) \dot{\psi} s_\theta c_\theta \\
c_{33} =& (I_{22} - I_{33}) \dot{\phi} c_\phi s_\phi c_\theta^2 + (I_{11} - I_{22} s_\phi^2 - I_{33} c_\phi^2) \dot{\theta} c_\theta s_\theta
\end{aligned}
$$

as computed using Matlab's symbolical toolbox, confirming the derivation in [Luukkonen, 2011] with some notable differences. For instance, the exponent s_ϕ^2 in the expression of c_{12} which we perceive to be an error in previous publications.

A.4 Gimbal lock avoidance with Tait-Bryan angles

Here we will here we will propose a method for getting around the issue of the gimbal lock while still maintaining the Tait-Bryan angle coordinate system in the quad-copter equations. For this discussion, we notate the l^∞-norm of an n-dimensional array and it's supremum with

$$||\mathbf{x}_{[0,T]}||_\infty = \sup_{t\in[0,T]}(||\mathbf{x}(t)||_\infty) = \sup_{t\in[0,T]}(\max\{|x_1(t)|,...,|x_n(t)|\}). \quad \text{(A.15)}$$

We consider the *UAV* system defined by equation (2.22), with a state vector

$$\mathbf{x}^T(t) = \begin{bmatrix}\mathbf{p}^T(t) & \dot{\mathbf{p}}^T(t) & \boldsymbol{\eta}^T(t) & \dot{\boldsymbol{\eta}}^T(t)\end{bmatrix}, \qquad \mathbf{x}(0) \in \mathbf{0} \quad \text{(A.16)}$$

with the maps $f_\mathbf{p} : (\dot{\mathbf{p}}, \boldsymbol{\eta}, \mathbf{T}_B) \to \ddot{\mathbf{p}}$ and $f_{\boldsymbol{\eta}} : (\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \boldsymbol{\tau}_B) \to \ddot{\boldsymbol{\eta}}$ and then note that a dynamical singularity occurs if at any point in time

$$||\mathbf{x}_{[0,T]}||_\infty \to \infty. \quad \text{(A.17)}$$

Operating in infinite time, $t \in [0,\infty)$, we may find singularities satisfying (A.17) which need not depend on poorly conditioned dynamics. For instance, if the thrust is set constant with angular states $\boldsymbol{\eta} = \dot{\boldsymbol{\eta}} = 0$ with $\mathbf{G} \neq \frac{1}{m}\mathbf{T}_B$, there will exist stable equilibrium point $\dot{\mathbf{p}} \neq 0$ where the speeds remain constant indefinitely, implying that $||\mathbf{p}||_\infty \to \infty$ as $t \to \infty$. However, by restricting the system to finite time, where clearly

$$||\mathbf{p}_{[0,T]}||_\infty \leq ||\dot{\mathbf{p}}_{[0,T]}||_\infty t_f \leq ||\ddot{\mathbf{p}}_{[0,T]}||_\infty t_f^2/2 \quad \text{(A.18)}$$

with identical results in the $\boldsymbol{\eta}$-terms. From this, it is evident that

$$\max\{||\{\ddot{\mathbf{p}}_{[0,T]}||_\infty, ||\ddot{\boldsymbol{\eta}}_{[0,T]}\}||_\infty\} < \infty \Rightarrow ||\mathbf{x}_{[0,T]}||_\infty < \infty \quad \text{(A.19)}$$

and we set out to check if this condition can possibly be violated for any combination of states in the state space and finite time. As the mappings define two systems (one linear and one non-linear) coupled in their non-linear input, the approach here will be to examine if any singularities exist by (A.19) in $f_\mathbf{p}$ and $f_{\boldsymbol{\eta}}$ respectively.

First, we note that the mapping $f_\mathbf{p}(\dot{\mathbf{p}}, \boldsymbol{\eta}, \mathbf{T}_B) = H_\mathbf{p}(\dot{\mathbf{p}}) + G_\mathbf{p}(\boldsymbol{\eta}, \mathbf{T}_B)$ in the system (2.22) will not pose any difficulties, as $H_\mathbf{p}(\dot{\mathbf{p}})$ is linear with respect to $\dot{\mathbf{p}}$ with eigenvalues $\lambda_i = \{-\frac{1}{m}D_{11}, -\frac{1}{m}D_{22}, -\frac{1}{m}D_{33}\}$ in the left half plane

as $D_{ii} > 0 \forall i$Consequently, it is bounded-input-bounded-output stable if the non-linear system input $G_{\mathbf{p}}(\boldsymbol{\eta}, \mathbf{T}_B)$ is bounded, as

$$||G_{\mathbf{p}}(\boldsymbol{\eta}, \mathbf{T}_B)_{[0,T]}||_\infty \leq \sup_{t\in[0,T]} (||\mathbf{G} + \frac{1}{m}\mathbf{T}_B||_\infty) \tag{A.20}$$

$$\leq \sup_{t\in[0,T]} (||\mathbf{G}||_\infty + \frac{1}{m}||T\mathbf{R}\hat{\mathbf{z}}||_\infty) \tag{A.21}$$

$$\leq \sup_{t\in[0,T]} (g + \frac{1}{m}|T|), \tag{A.22}$$

by the triangle inequality, we see that $|T| < \infty \Rightarrow ||G_{\mathbf{p}}(\mathbf{p}, \mathbf{T}_B)_{[0,T]}||_\infty < \infty$ regardless of $\boldsymbol{\eta}$.

The second and more troublesome mapping, $f_{\boldsymbol{\eta}}$, can by equation (2.22) be written as the sum of the non-linear system $H_{\boldsymbol{\eta}}(\cdot)$ and the non-linear input $G_{\boldsymbol{\eta}}(\cdot)$ with

$$H_{\boldsymbol{\eta}}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) = -\mathbf{J}^{-1}(\boldsymbol{\eta})\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}} \qquad \text{and} \qquad G_{\boldsymbol{\eta}}(\boldsymbol{\eta}, \boldsymbol{\tau}_B) = \mathbf{J}^{-1}(\boldsymbol{\eta})\boldsymbol{\tau}_{\mathcal{B}}. \tag{A.23}$$

Under the restriction that $\mathbf{I}_{\mathcal{B}}$ is approximately a diagonal matrix with $I_{ii} > 0$, it can be shown by the definition of (2.6) that

$$\det(\mathbf{J}(\boldsymbol{\eta})) = \det(\mathbf{W}^T(\boldsymbol{\eta})\mathbf{I}_B\mathbf{W}(\boldsymbol{\eta})) = I_{11} \cdot I_{22} \cdot I_{33} \cdot \cos^2(\theta) \tag{A.24}$$

is zero valued when $\theta = (1/2 + n) \cdot \pi \quad \forall n \in \mathbb{Z}$, regardless of ϕ, ψ. Here a Gimbal lock occurs as described in equation (2.23). Consequently, the map $f_{\boldsymbol{\eta}}$ becomes singular and the dynamics explode as both the maps $G_{\boldsymbol{\eta}}(\cdot)$ and $H_{\boldsymbol{\eta}}(\cdot)$ contain the inverse $\mathbf{J}^{-1}(\boldsymbol{\eta})$. If we wish to use the Euler-Lagrange model for simulation and control close to, or even beyond this point, we must alter the equation in some way.

To this end, we first introduce the matrix condition number in the l^2-norm of the Jacobian matrix $\mathbf{J}$, here written in terms of the matrix singular values $\sigma(\mathbf{J})$, or the matrix eigenvalues $\lambda(\mathbf{J})$ as

$$\kappa_{\mathbf{J}}(\boldsymbol{\eta}) = \frac{\max(\sigma(\mathbf{J}(\boldsymbol{\eta}))}{\min(\sigma(\mathbf{J}(\boldsymbol{\eta}))} = \frac{\max(|\lambda(\mathbf{J}(\boldsymbol{\eta})|)}{\min(|\lambda(\mathbf{J}(\boldsymbol{\eta})|)} \tag{A.25}$$

where the last equality holds since the eigenvalues are positive by (A.24) and the matrix $\mathbf{J}$ is real-valued. In the angular dynamics, for a fixed point in $\boldsymbol{\eta}\dot{\boldsymbol{\eta}}$-space we will effectively be solving a linear system which is conditioned by $\mathbf{J}$. If we are close to a singularity where the dynamics explode, the condition number will approach infinity as $\det(\mathbf{J}) \to 0 \Rightarrow \min(\lambda(\mathbf{J}(\boldsymbol{\eta})) \to 0$. However, by introducing an upper bound on the condition number we can guarantee that the system is well conditioned regardless of $\theta(t)$.

We first make the assumption that in any sensible flight, the system is highly volatile around the singularity as it can't retain in a position $\theta = (1/2 + n) \cdot \pi$ for too long without crashing due to gravitational acceleration. Consequently, the time spent close to a singularity in the $\boldsymbol{\eta}$-space can be assumed to be very short in any meaningful flight. We also assume that the dynamics of a physical quadcopter is unlikely to change much in a small neighbourhood around the singularity. We can then motivate modifying the governing torque equations to keep the angular accelerations in safe regions if a feasibility condition

$$|\cos(\theta)| > \epsilon \tag{A.26}$$

is violated, for some small numerical limit ϵ. If an infeasible value of θ is detected, Jacobian $\mathbf{J}^{-1}(\boldsymbol{\eta})$ is kept constant at the most recent feasible $\mathbf{J}$ matrix, denoted $\mathbf{J}_f^{-1}$. The modified system dynamics become

$$\ddot{\boldsymbol{\eta}} = \begin{cases} \mathbf{J}^{-1}(\boldsymbol{\eta})(\boldsymbol{\tau}_B - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}}), & \text{if} \quad |\cos(\theta)| > \epsilon \\ \mathbf{J}_f^{-1}(\boldsymbol{\tau}_B - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}}), & \text{if} \quad |\cos(\theta)| \leq \epsilon \end{cases} \tag{A.27}$$

which is relatively simple to implement, both in continuous and discrete time. This effectively puts a sought after upper bound on the condition number

$$\kappa_{\mathbf{J}}(\boldsymbol{\eta}) \leq \frac{\max(\sigma(\mathbf{J}(\boldsymbol{\eta}))}{\min(\sigma(\mathbf{J}(\boldsymbol{\eta}))}\Big|_{\theta=\pi/2\pm\epsilon}, \tag{A.28}$$

and then implicitly a lower positive bound on $0 < d < \det(\mathbf{J})$ for a constant d which can be computed analytically. Then

$$||\mathbf{J}^{-1}(\boldsymbol{\eta})||_\infty = \Big|\Big|\frac{\text{adj}(\mathbf{J}(\boldsymbol{\eta}))}{\det(\mathbf{J}(\boldsymbol{\eta}))}\Big|\Big|_\infty \leq \frac{3(I_{22} + I_{33})I_{11} + 4I_{22}I_{33}}{\det(\mathbf{J}(\boldsymbol{\eta}))} \triangleq \frac{n}{d} \tag{A.29}$$

is bounded in the infinity norm, as $n, d > 0$ with n bounded from above by the inertial terms. Using this modified system, we may show that for $f_{\boldsymbol{\eta}}$ the non-linear system input satisfies

$$||G(\mathbf{p}, \mathbf{T}_B)_{[0,T]}||_\infty = \sup_{t\in[0,T]} (||\mathbf{J}^{-1}(\boldsymbol{\eta})\boldsymbol{\tau}_B||_\infty) \tag{A.30}$$

$$\leq \sup_{t\in[0,T]} \left(\frac{n}{d} + ||\boldsymbol{\tau}_B||_\infty\right) \tag{A.31}$$

showing that $||\boldsymbol{\tau}_B||_\infty < \infty \Rightarrow ||G(\mathbf{p}, \mathbf{T}_B)_{[0,T]}||_\infty < \infty$ when using the modified torque equations (2.24).

We have then shown that the system with modified torque equations in (2.24) will not suffer dynamical singularities induced by the Gimbal lock in finite time if the input signals are also finite.

A.5 Quaternion rotations and relation to Tait-Bryan angles

Consider the rotation of a vector $\mathbf{x}_A$ around the unit length vector $\mathbf{v}$. Let $\mathbf{x}_A$ be the sum of two vectors, one parallel to $\mathbf{v}$ and the other orthogonal to $\mathbf{v}$, such that $\mathbf{x}_{\parallel}^{(A)} = \mathbf{v}\mathbf{v}^T\mathbf{x}_A$ and $\mathbf{x}_{\perp}^{(A)} = \mathbf{x}_A - \mathbf{v}\mathbf{v}^T\mathbf{x}_A$ where then

$$\mathbf{x}_{\parallel}^{(A)} + \mathbf{x}_{\perp}^{(A)} = \mathbf{v}\mathbf{v}^T\mathbf{x}_A + \mathbf{x}_A - \mathbf{v}\mathbf{v}^T\mathbf{x}_A = \mathbf{x}_A. \tag{A.32}$$

The rotation of the vector $\mathbf{x}_A$ by an angle θ around $\mathbf{v}$ to the new vector $\mathbf{x}_B$ will not affect the parallel component, for which $\mathbf{x}_{\parallel}^{(A)} = \mathbf{x}_{\parallel}^{(B)}$. However, the perpendicular component will be rotated in the plane spanned by the basis vectors $\mathbf{e}_1 = \mathbf{x}_{\perp}^{(A)}$ and $\mathbf{e}_2 = \mathbf{v} \times \mathbf{x}_{\perp}^{(A)} = \mathbf{v} \times \mathbf{x}_A$, resulting in the vector rotation formula

$$\mathbf{x}_B = \mathbf{x}_{\parallel}^{(B)} + \mathbf{x}_{\perp}^{(B)} = \mathbf{x}_{\parallel}^{(A)} + \cos(\theta)\mathbf{x}_{\perp}^{(A)} + \sin(\theta)(\mathbf{v} \times \mathbf{x}_A). \tag{A.33}$$

By definition (A.34), the quaternion may be written

$$\mathbf{q} = \begin{bmatrix} \cos(\theta/2) \\ \mathbf{v}\sin(\theta/2) \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ v_x \sin(\theta/2) \\ v_y \sin(\theta/2) \\ v_z \sin(\theta/2) \end{bmatrix} = \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} q_w \\ \mathbf{q}_v \end{bmatrix} \in \mathbb{R}^{4\times 1} \tag{A.34}$$

In which case we may show that an operation

$$\begin{bmatrix} 0 \\ \mathbf{x}_B \end{bmatrix} = \mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \mathbf{q}^* \tag{A.35}$$

is equivalent to rotating a vector $\mathbf{x}_A$ to $\mathbf{x}_B$ according to the vector rotation formula. Let

$$\mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \mathbf{q}^* = \Big(\begin{bmatrix} q_w \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{q}_v \end{bmatrix} \Big) \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \Big(\begin{bmatrix} q_w \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ -\mathbf{q}_v \end{bmatrix} \Big) \tag{A.36}$$

Using the definitions of $[\cdot]_L$ and $[\cdot]_R$, we find that the imaginary part of the above expression may be written

$$\begin{aligned} \Im\Big(\mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \mathbf{q}^*\Big) =& q_w^2\mathbf{x}_A + q_w([\mathbf{q}_v]_\times - [-\mathbf{q}_v]_\times)\mathbf{x}_A + (\mathbf{q}_v\mathbf{q}_v^T - [\mathbf{q}_v]_\times[-\mathbf{q}_v]_\times)\mathbf{x}_A \\ =& q_w^2\mathbf{x}_A + 2q_w[\mathbf{q}_v]_\times\mathbf{x}_A + (\mathbf{q}_v\mathbf{q}_v^T + [\mathbf{q}_v]_\times^2)\mathbf{x}_A \end{aligned} \tag{A.37}$$

Interpreting the quaternion as a rotation of θ about a unit vector $\mathbf{v}$,

$$\begin{aligned}
\Im\Big(\mathbf{q}\otimes\begin{bmatrix}0\\ \mathbf{x}_A\end{bmatrix}\otimes\mathbf{q}^*\Big) &= q_w^2\mathbf{x}_A + q_w([\mathbf{q}_v]_\times - [-\mathbf{q}_v]_\times)\mathbf{x}_A + (\mathbf{q}_v\mathbf{q}_v^T - [\mathbf{q}_v]_\times[-\mathbf{q}_v]_\times)\mathbf{x}_A \\
&= \mathbf{x}_A\cos^2(\theta/2) + 2(\mathbf{v}\times\mathbf{x}_A)\cos(\theta/2)\sin(\theta/2) + (\mathbf{v}\mathbf{v}^T + [\mathbf{v}]_\times^2)\mathbf{x}_A\sin^2(\theta/2) \\
&= \mathbf{x}_A\cos^2(\theta/2) + (\mathbf{v}\times\mathbf{x}_A)2\cos(\theta/2)\sin(\theta/2) + (\mathbf{v}\mathbf{v}^T + \mathbf{v}\mathbf{v}^T - \mathbf{I})\mathbf{x}_A\sin^2(\theta/2) \\
&= \mathbf{x}_A(\cos^2(\theta/2) - \sin^2(\theta/2)) + (\mathbf{v}\times\mathbf{x}_A)\sin(\theta) + \mathbf{v}\mathbf{v}^T\mathbf{x}_A 2\sin^2(\theta/2) \\
&= \mathbf{x}_A\cos(\theta) + (\mathbf{v}\times\mathbf{x}_A)\sin(\theta) + \mathbf{v}\mathbf{v}^T\mathbf{x}_A(1-\cos(\theta) \\
&= (\mathbf{x}_A - \mathbf{v}\mathbf{v}^T\mathbf{x})\cos(\theta) + (\mathbf{v}\times\mathbf{x}_A)\sin(\theta) + \mathbf{v}\mathbf{v}^T\mathbf{x}_A
\end{aligned} \tag{A.38}$$

Recalling the vector rotation formula, it is clear that

$$\Im\Big(\begin{bmatrix}0\\ \mathbf{x}_B\end{bmatrix}\Big) = (\mathbf{x}_A - \mathbf{v}\mathbf{v}^T\mathbf{x})\cos(\theta) + (\mathbf{v}\times\mathbf{x}_A)\sin(\theta) + \mathbf{v}\mathbf{v}^T\mathbf{x}_A = \Im\Big(\mathbf{q}\otimes\begin{bmatrix}0\\ \mathbf{x}_A\end{bmatrix}\otimes\mathbf{q}^*\Big) \tag{A.39}$$

showing that we may perform a three-dimensional rotation of the vector $\mathbf{x}_A$ using two quaternion products, $\otimes$, without ever evaluating the trigonometric functions.

With this operation, the quaternion rotation from $\mathbf{x}_A$ to $\mathbf{x}_B$ may be expressed in relation to the Tait-Bryan angle rotation and the rotation matrix respectively. All methods form an *SO(3)* rotation, related by

$$\mathbf{x}_B = \mathbf{R}\mathbf{x}_A = \mathbf{R}\{\phi,\theta,\psi\}\mathbf{x}_A = \Im\Big(\mathbf{q}\otimes\begin{bmatrix}0\\ \mathbf{x}_A\end{bmatrix}\otimes\mathbf{q}^*\Big) \tag{A.40}$$

Recalling first the rotation matrix in the ZYX Tait-Bryan convention, we let $s_i = \sin(i)$ and $c_i = \cos(i)$, forming

$$\mathbf{R}\{\phi,\theta,\psi\} = \begin{bmatrix} c_\phi c_\psi & s_\psi c_\phi & -s_\theta \\ c_\psi s_\theta s_\phi - s_\psi c_\phi & s_\psi s_\theta s_\phi + c_\phi c_\psi & c_\theta s_\phi \\ c_\psi s_\theta c_\phi + s_\phi s_\psi & s_\psi s_\theta c_\phi - c_\psi s_\phi & c_\phi c_\theta \end{bmatrix} \tag{A.41}$$

By letting $\mathbf{x}_A$ assume the basis vectors $\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}$ in turn, we may then computing the resulting quaternion rotation express the rotation as

$$\mathbf{R}\{\mathbf{q}\} = \begin{bmatrix} q_w^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_w q_z) & 2(q_x q_z + q_w q_y) \\ 2(q_x q_y + q_w q_z) & q_w^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_w q_x) \\ 2(q_x q_z - q_w q_y) & 2(q_y q_z + q_w q_x) & q_w^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix}, \tag{A.42}$$

In doing so, the relationship $\mathbf{q} \to (\phi,\theta,\psi)$ is given by

$$\phi = \arctan2[2(q_w q_x + q_y q_z), 1 - 2(q_x^2 + q_y^2)] \tag{A.43}$$

$$\theta = \arcsin[2(q_w q_y - q_x q_z)] \tag{A.44}$$

$$\psi = \arctan2[2(q_w q_z + q_x q_y), 1 - 2(q_y^2 + q_z^2)] \tag{A.45}$$

where arctan2[· , ·] denotes the four quadrant inverse tangent function [Fresk and Nikolakopoulos, 2013]. Similarly, the reverse mapping $(\phi, \theta, \psi) \rightarrow \mathbf{q}$ becomes

$$\mathbf{q} = \begin{bmatrix} c_{\phi/2}c_{\theta/2}c_{\psi/2} + s_{\phi/2}s_{\theta/2}s_{\psi/2} \\ s_{\phi/2}c_{\theta/2}c_{\psi/2} - c_{\phi/2}s_{\theta/2}s_{\psi/2} \\ c_{\phi/2}s_{\theta/2}c_{\psi/2} + s_{\phi/2}c_{\theta/2}s_{\psi/2} \\ c_{\phi/2}c_{\theta/2}s_{\psi/2} - s_{\phi/2}s_{\theta/2}c_{\psi/2} \end{bmatrix} \tag{A.46}$$

where it should be noted that $||\mathbf{q}|| = 1$ at all times.

A.6 Quaternion rate of change

Let the angular rate vector of the quadcopter in the global frame be written

$$\boldsymbol{\omega}_{\mathcal{G}} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T \tag{A.47}$$

such that

$$\dot{\mathbf{x}} = \boldsymbol{\omega}_{\mathcal{G}} \times \mathbf{x} \tag{A.48}$$

for some vector $\mathbf{x} \in \mathbb{R}^3$ defined in the global reference frame. Consider then a quaternion, $\mathbf{q}$, which rotates a vector $\mathbf{x}_A$ to a vector $\mathbf{x}_B$ within this frame of reference by

$$\begin{bmatrix} 0 \\ \mathbf{x}_B \end{bmatrix} = \mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \mathbf{q}^* \Leftrightarrow \mathbf{q} \otimes^* \begin{bmatrix} 0 \\ \mathbf{x}_B \end{bmatrix} \otimes \mathbf{q} = \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \tag{A.49}$$

Taking the time derivative of the vector $\mathbf{x}_A$ and using the chain rule results in

$$\begin{bmatrix} 0 \\ \dot{\mathbf{x}}_B \end{bmatrix} = \dot{\mathbf{q}} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \mathbf{q}^* + \mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_A \end{bmatrix} \otimes \dot{\mathbf{q}}^* \tag{A.50}$$

by the definition of the quaternion. Inserting the expression in equation (A.49),

$$\begin{bmatrix} 0 \\ \dot{\mathbf{x}}_B \end{bmatrix} = \dot{\mathbf{q}} \otimes \mathbf{q}^* \otimes \begin{bmatrix} 0 \\ \mathbf{x}_B \end{bmatrix} \otimes \underbrace{\mathbf{q} \otimes \mathbf{q}^*}_{=1} + \underbrace{\mathbf{q} \otimes \mathbf{q}^*}_{=1} \otimes \begin{bmatrix} 0 \\ \mathbf{x}_B \end{bmatrix} \otimes \mathbf{q} \otimes \dot{\mathbf{q}}^* \tag{A.51}$$

By the condition $||\mathbf{q}|| = 1$ and the definition of the quaternion product, we may show

$$\begin{cases} \Re\{\dot{\mathbf{q}} \otimes \mathbf{q}^*\} = \Re\{\mathbf{q} \otimes \dot{\mathbf{q}}^*\} = 0 \\ \Im\{\dot{\mathbf{q}} \otimes \mathbf{q}^*\} = -\Im\{\mathbf{q} \otimes \dot{\mathbf{q}}^*\} = -\dot{q}_w \mathbf{q}_v + q_w \dot{\mathbf{q}}_v - \dot{\mathbf{q}}_v \times \mathbf{q}_v \triangleq \mathbf{u} \end{cases} \tag{A.52}$$

Expression (A.49) may then be further simplified using the definitions $[\cdot]_L$ and $[\cdot]_R$, as

$$\begin{bmatrix}0\\ \dot{\mathbf{x}}_B\end{bmatrix} = \begin{bmatrix}0\\ \mathbf{u}\end{bmatrix} \otimes \begin{bmatrix}0\\ \mathbf{x}_B\end{bmatrix} + \begin{bmatrix}0\\ \mathbf{x}_B\end{bmatrix} \otimes \begin{bmatrix}0\\ -\mathbf{u}\end{bmatrix} \Leftrightarrow \dot{\mathbf{x}}_B = 2(\mathbf{u}\times\mathbf{x}_B) = \boldsymbol{\omega}_{\mathcal{G}}\times\mathbf{x}_B \Leftrightarrow 2\mathbf{u} = \boldsymbol{\omega}_{\mathcal{G}} \tag{A.53}$$

by equation (A.48). Susbstituting $\mathbf{u}$, we find the relationship

$$\dot{\mathbf{q}} \otimes \mathbf{q}^* = \begin{bmatrix}\Re\{\dot{\mathbf{q}} \otimes \mathbf{q}^*\}\\ \Im\{\dot{\mathbf{q}} \otimes \mathbf{q}^*\}\end{bmatrix} = \begin{bmatrix}0\\ \mathbf{u}\end{bmatrix} = \begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{G}}/2\end{bmatrix} \Leftrightarrow \dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{G}}\end{bmatrix} \otimes \mathbf{q}. \tag{A.54}$$

However, we are interested in quaternion time derivative with respect to the angular rates in the body coordinate system. As the quaternion throughout this document describes a rotation from the body frame to the the inertial or global frames,

$$\begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{G}}\end{bmatrix} = \mathbf{q} \otimes \begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{B}}\end{bmatrix} \otimes \mathbf{q}^* \tag{A.55}$$

Consequently, the angular rate vector in the body frame of reference may be written

$$\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{G}}\end{bmatrix} \otimes \mathbf{q} = \frac{1}{2}\mathbf{q} \otimes \begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{B}}\end{bmatrix} \otimes \mathbf{q}^* \otimes \mathbf{q} = \frac{1}{2}\mathbf{q} \otimes \begin{bmatrix}0\\ \boldsymbol{\omega}_{\mathcal{B}}\end{bmatrix}. \tag{A.56}$$

A.7 Quaternion state-space representation

To facilitate a discussion on numerical integration and model based control, we define the system using quaternion rotation with the Newton-Euler equations on a state space form. The states and control signals are then

$$\mathbf{x}(t) = \begin{bmatrix}\mathbf{p}(t)\\ \dot{\mathbf{p}}(t)\\ q_w(t)\\ \mathbf{q}_v(t)\\ \boldsymbol{\omega}_{\mathcal{B}}(t)\end{bmatrix} \in \mathbb{R}^{13\times 1}, \quad \text{and} \quad \mathbf{u}(t) = \begin{bmatrix}T(t)\\ \boldsymbol{\tau}(t)\end{bmatrix} \in \mathbb{R}^{4\times 1} \tag{A.57}$$

respectively. The full non-linear system (2.39) can be written

$$\begin{aligned}\dot{\mathbf{x}}(t) =& \mathbf{A}(\mathbf{q},\boldsymbol{\omega})\mathbf{x}(t) + \mathbf{B}(\mathbf{q})\mathbf{u}(t) + \mathbf{G}\\ \mathbf{y}(t) =& \mathbf{C}\mathbf{x}(t)\end{aligned} \tag{A.58}$$

with

$$\mathbf{A}_c = \begin{bmatrix}\mathbf{0} & \mathbf{I} & 0 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & -\frac{1}{m}\mathbf{R}^T_{\mathcal{BG}}\{\mathbf{q}\}\mathbf{D}_{\mathcal{B}} & 0 & \mathbf{0} & \mathbf{0}\\ 0 & 0 & 0 & 0 & -\frac{1}{2}\mathbf{q}_v^T\\ \mathbf{0} & \mathbf{0} & 0 & \mathbf{0} & \frac{1}{2}(q_w\mathbf{I} + [\mathbf{q}_v]_\times)\\ \mathbf{0} & \mathbf{0} & 0 & \mathbf{0} & -\mathbf{I}_{\mathcal{B}}^{-1}[\boldsymbol{\omega}]_\times\mathbf{I}_{\mathcal{B}}\end{bmatrix}, \quad \mathbf{B}_c = \begin{bmatrix}\mathbf{0}_{3\times 1} & \mathbf{0}\\ \frac{1}{m}\mathbf{R}^T_{\mathcal{BG}}\{\mathbf{q}\}\hat{\mathbf{z}}_{\mathcal{G}} & \mathbf{0}\\ 0 & \mathbf{0}_{1\times 3}\\ \mathbf{0}_{3\times 1} & \mathbf{0}\\ \mathbf{0}_{3\times 1} & \mathbf{I}_{\mathcal{B}}^{-1}\end{bmatrix}, \tag{A.59}$$

and

$$\mathbf{G} = \begin{bmatrix}\mathbf{0}_{1\times 5} & -g & \mathbf{0}_{1\times 7}\end{bmatrix}^T \tag{A.60}$$

where all sub-matrices are 3×3 unless explicitly stated otherwise, with the inertial matrix, $\mathbf{I}_{\mathcal{B}}$, and drag matrix $\mathbf{D}_{\mathcal{B}}$ defined with respect to the body frame. The measurement matrix $\mathbf{C}$ is chosen to reflect the available sensory and was discussed in **Chapter 6**,.

A.8 Linearised systems

The Tait-Bryan angle model (2.22) with the states and control signals

$$\mathbf{x}(t) = \begin{bmatrix}\mathbf{p} & \dot{\mathbf{p}} & \boldsymbol{\eta} & \dot{\boldsymbol{\eta}}\end{bmatrix}^T \in \mathbb{R}^{12\times 1}, \qquad \mathbf{u}(t) = \begin{bmatrix}T & \tau_\phi & \tau_\theta & \tau_\psi\end{bmatrix}^T \in \mathbb{R}^{4\times 1} \tag{A.61}$$

was shown to be well conditioned close to the stable hovering point $\boldsymbol{\eta} = \dot{\boldsymbol{\eta}} = \mathbf{0}$ and $T = mg$. At this point, the linearised system matrices reduce to

$$\tilde{\mathbf{A}}_{\Delta\mathbf{x}} = \begin{bmatrix}\mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\frac{1}{m}\mathbf{D} & \frac{1}{m}\hat{\mathbf{T}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\end{bmatrix}, \qquad \tilde{\mathbf{B}}_{\Delta\mathbf{u}} \begin{bmatrix}\mathbf{0}_{5\times 1} & \mathbf{0}_{5\times 3} \\ \frac{1}{m} & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 1} & \mathbf{I}_{\mathcal{B}}^{-1}\end{bmatrix} \tag{A.62}$$

where

$$\hat{\mathbf{T}} = \begin{bmatrix}0 & mg & 0 \\ -mg & 0 & 0 \\ 0 & 0 & 0\end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{A.63}$$

In contrast, to enable model based control and state estimation in the entire $\boldsymbol{\eta}$-space, we should use the singularity free quaternion representation as derived by the Newton-Euler equations. Here instead, the state vector and control signals are defined as

$$\mathbf{x}(t) = \begin{bmatrix}\mathbf{p} & \dot{\mathbf{p}} & \mathbf{q} & \boldsymbol{\omega}_{\mathcal{B}}\end{bmatrix}^T \in \mathbb{R}^{13\times 1}, \qquad \mathbf{u}(t) = \begin{bmatrix}T & \tau_x & \tau_y & \tau_z\end{bmatrix}^T \in \mathbb{R}^{4\times 1} \tag{A.64}$$

where the thrust remains the same, but the torques are now defined with respect to the basis of the body coordinate system. The linearised system matrices can then be written

$$\tilde{\mathbf{A}}_{\Delta\mathbf{x}} = \begin{bmatrix}\mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\frac{1}{m}\mathbf{R}_{\mathcal{BG}}\{\mathbf{q}\}\mathbf{D}_{\mathcal{B}} & \mathbf{A}_{23} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{33} & \mathbf{A}_{34} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{44}\end{bmatrix}, \quad \tilde{\mathbf{B}}_{\Delta\mathbf{u}} = \begin{bmatrix}\mathbf{0}_{3\times 1} & \mathbf{0} \\ \frac{1}{m}\mathbf{R}_{\mathcal{BG}}^T\{\mathbf{q}\}\hat{\mathbf{z}}_{\mathcal{G}} & \mathbf{0} \\ \mathbf{0}_{4\times 1} & \mathbf{0} \\ \mathbf{0}_{3\times 1} & \mathbf{I}_{\mathcal{B}}^{-1}\end{bmatrix}, \tag{A.65}$$

where

$$\mathbf{A}_{23} = \frac{2}{m}\left[\ (q_w + [\mathbf{q}_v]_\times \mathbf{T}_\mathcal{B}) \ \middle|\ \mathbf{q}_v^T\mathbf{T}_\mathcal{B}\mathbf{I} + \mathbf{q}_v\mathbf{T}_\mathcal{B}^T - \mathbf{T}_\mathcal{B}\mathbf{q}_v^T - q_w[\mathbf{T}_\mathcal{B}]_\times\ \right] \in \mathbb{R}^{3\times 4}$$

$$\mathbf{A}_{33} = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{\omega}_\mathcal{B}^T \\ \boldsymbol{\omega}_\mathcal{B} & [\boldsymbol{\omega}_\mathcal{B}]_\times \end{bmatrix} \in \mathbb{R}^{4\times 4}$$

$$\mathbf{A}_{34} = \frac{1}{2}\begin{bmatrix} -\mathbf{q}_v^T \\ \hdashline [\mathbf{q}_v]_\times \end{bmatrix} \in \mathbb{R}^{4\times 3}$$

$$\mathbf{A}_{44} = \mathbf{I}_\mathcal{B}^{-1}([\mathbf{I}_\mathcal{B}\boldsymbol{\omega}_\mathcal{B}]_\times - [\boldsymbol{\omega}_\mathcal{B}]_\times \mathbf{I}_\mathcal{B}) \in \mathbb{R}^{3\times 3}.$$

A.9 Closed form system integration with constant terms

Consider a general system

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G} \tag{A.66}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) \tag{A.67}$$

such that

$$\mathbf{A} \in \mathbb{R}^{M\times M}, \quad \mathbf{B} \in \mathbb{R}^{M\times N}, \quad \mathbf{G} \in \mathbb{R}^{M\times 1} \tag{A.68}$$

which then encompasses all the considered systems, both quaternion and Tait-Bryan, including their respective linearizations (where then $\mathbf{G} \equiv 0$). Then, using the matrix exponential $e^{-\mathbf{A}t}$ as an integrating factor,

$$e^{-\mathbf{A}t}\dot{\mathbf{x}}(t) - e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t) = \frac{d}{dt}(e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t)) = e^{-\mathbf{A}t}(\mathbf{B}\mathbf{u}(t) + \mathbf{G}) \tag{A.69}$$

and by the fundamental theorem of calculus

$$e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t) = e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t_0) + \int_{t_0}^{t} e^{-\mathbf{A}\tau}(\mathbf{B}\mathbf{u}(\tau) + \mathbf{G})d\tau \tag{A.70}$$

Here we use zero order hold approximation, assuming that $\mathbf{u}(t) = \mathbf{u}(t_k)\ \ \forall t \in [t_k, t_k + \Delta t]$. Furthermore, we consider $\mathbf{x}(t_k)$ to be known and use the parametrisation $s = \tau - t_k$, the equation (A.69) may then be re-written as

$$\mathbf{x}(t_{k+1}) = e^{\mathbf{A}\Delta t}x(t_k) + \int_0^h e^{\mathbf{A}s}ds\mathbf{B}\mathbf{u}(t_k) + \int_0^h e^{\mathbf{A}s}ds\mathbf{G} = \boldsymbol{\Phi}x(t_k) + \boldsymbol{\Gamma}\mathbf{u}(t_k) + \boldsymbol{\Psi} \tag{A.71}$$

By rewriting $\hat{\mathbf{u}}^T(t_k) = \begin{bmatrix}\mathbf{u}^T(t_k) & 1\end{bmatrix}^T$ and $\hat{\mathbf{B}} = \begin{bmatrix}\mathbf{B} & \mathbf{G}\end{bmatrix} \in \mathbb{R}^{M\times N+1}$, the resulting system qualifies for the standard exponential matrix solution by **Theorem 3.1.1** in [Chen and Francis, 2012]. We let

$$\mathbf{M} = \begin{bmatrix}\mathbf{A} & \hat{\mathbf{B}} \\ \mathbf{0} & \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{A} & \mathbf{B} & \mathbf{G} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0}\end{bmatrix} \in \mathbb{R}^{(M+N+1)\times(M+N+1)} \tag{A.72}$$

where then

$$e^{\mathbf{M}\Delta t} = \sum_{k=0}^{\infty} \frac{1}{k!}(\mathbf{M}\Delta t)^k \approx \mathbf{I} + \mathbf{M}\Delta t + O(||(\mathbf{M}\Delta t)||_2^2) = \begin{bmatrix} \mathbf{\Phi} & \mathbf{\Gamma} & \mathbf{\Psi} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}. \quad \text{(A.73)}$$

The discrete time system may then be written

$$\mathbf{x}(hk+h) = \mathbf{\Phi}\mathbf{x}(hk) + \mathbf{\Gamma}\mathbf{u}(hk) + \mathbf{\Psi} \quad \text{(A.74)}$$

$$\mathbf{y}(hk) = \mathbf{C}\mathbf{x}(hk). \quad \text{(A.75)}$$

A.10 Cross product identities

In this section, we list derive an identity necessary for develop the attitude regulation in the geometric tracking control system. Consider three vectors $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$ with the notation $\mathbf{u} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T$ and an invertible matrix $\mathbf{A} \in \mathbb{R}^{3\times 3}$. Having previously defined the cross product

$$\mathbf{u} \times \mathbf{v} = [\mathbf{u}]_\times \mathbf{v} = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \mathbf{v} \quad \text{(A.76)}$$

we now show that the identity

$$[\mathbf{A}\mathbf{v}]_\times = det(\mathbf{A})\mathbf{A}^{-T}[\mathbf{v}]_\times \mathbf{A}^{-1}. \quad \text{(A.77)}$$

holds at all times. Developing both sides yields,

$$[\mathbf{A}\mathbf{v}]_\times = det(\mathbf{A})\mathbf{A}^{-T}[\mathbf{v}]_\times \mathbf{A}^{-1} \quad \text{(A.78)}$$

$$\mathbf{A}^T[\mathbf{A}\mathbf{v}]_\times \mathbf{A} = det(\mathbf{A})[\mathbf{v}]_\times \quad \text{(A.79)}$$

$$\mathbf{u}^T\mathbf{A}^T[\mathbf{A}\mathbf{v}]_\times \mathbf{A} = det(\mathbf{A})\mathbf{u}^T[\mathbf{v}]_\times \quad \text{(A.80)}$$

$$\mathbf{u}^T\mathbf{A}^T[\mathbf{A}\mathbf{v}]_\times \mathbf{A}\mathbf{w} = det(\mathbf{A})\mathbf{u}^T[\mathbf{v}]_\times \mathbf{w} \quad \text{(A.81)}$$

$$(\mathbf{A}\mathbf{u}) \cdot (\mathbf{A}\mathbf{v} \times \mathbf{A}\mathbf{w}) = det(\mathbf{A})\mathbf{u} \cdot (\mathbf{v} \times \mathbf{w}) \quad \text{(A.82)}$$

$$\quad \text{(A.83)}$$

By the properties of the scalar triple product, the equations may be written

$$det(\mathbf{A}\mathbf{u}, \mathbf{A}\mathbf{v}, \mathbf{A}\mathbf{w}) = det(\mathbf{A})det(\mathbf{u}, \mathbf{v}, \mathbf{w}) \quad \text{(A.84)}$$

$$det(\mathbf{A})det(\mathbf{u}, \mathbf{v}, \mathbf{w}) = det(\mathbf{A})det(\mathbf{u}, \mathbf{v}, \mathbf{w}) \quad \text{(A.85)}$$

$$\quad \text{(A.86)}$$

showing that the identity (A.77) holds for all invertible matrices $\mathbf{A} \in \mathbb{R}^{3\times 3}$. In the special case of the rotational operator, $\mathbf{R} \in SO(3)$, satisfying $det(\mathbf{R}) = 1$ and $\mathbf{R}^T\mathbf{R} = \mathbf{I}$, the identity reduces to

$$[\mathbf{R}\mathbf{v}]_\times = \mathbf{R}[\mathbf{v}]_\times \mathbf{R}^T. \quad \text{(A.87)}$$

A.11 Time derivative of the rotation matrix

Consider a rotation matrix parametrisation $\mathbf{R}_{\mathcal{BG}} \in SO(3)$, rotating a body frame $\mathcal{B}$ to an inertial frame $\mathcal{I}$ by the definitions in **Section 2.1**. Writing this rotation in terms of small angle increments $\Delta\phi, \Delta\theta, \Delta\psi$ about the ZYX angles respectively, we can make good use of the extrinsic Tait Bryan ZYX angles. By (2.3) in **Section 2.1**

$$\mathbf{R}_{\mathcal{BG}}(\Delta\phi, \Delta\theta, \Delta\psi) = \mathbf{R}_{\mathcal{GB}}^T(\Delta\phi, \Delta\theta, \Delta\psi) = \begin{bmatrix} 1 & -\Delta\psi & \Delta\theta \\ \Delta\psi & 1 & -\Delta\phi \\ -\Delta\theta & \Delta\phi & 1 \end{bmatrix}. \quad \text{(A.88)}$$

The small change from the body to inertial frame caused by the angles is then

$$\Delta\mathbf{R}_{\mathcal{BG}} = \mathbf{R}_{\mathcal{BG}}(\Delta\phi, \Delta\theta, \Delta\psi)\mathbf{R}_{\mathcal{BG}} - \mathbf{R}_{\mathcal{BG}} = [\mathbf{R}_{\mathcal{BG}}(\Delta\phi, \Delta\theta, \Delta\psi) - \mathbf{I}]\mathbf{R}_{\mathcal{BG}} \quad \text{(A.89)}$$

and the change of this rotation with respect to time is

$$\frac{\Delta\mathbf{R}_{\mathcal{BG}}}{\Delta t} = \frac{1}{\Delta t}\begin{bmatrix} 0 & -\Delta\psi & \Delta\theta \\ \Delta\psi & 0 & -\Delta\phi \\ -\Delta\theta & \Delta\phi & 0 \end{bmatrix}\mathbf{R}_{\mathcal{BG}}. \quad \text{(A.90)}$$

using (A.88). Clearly, as the angles are extrinsic and defined with respect to the inertial frame, we let

$$\boldsymbol{\omega}_{\mathcal{G}} = \begin{bmatrix} \frac{\Delta\phi}{\Delta t} & \frac{\Delta\theta}{\Delta t} & \frac{\Delta\psi}{\Delta t} \end{bmatrix}^T \quad \text{(A.91)}$$

simplifying (A.90), the fact that $\omega_{\mathcal{B}} = \mathbf{R}_{\mathcal{GB}}\omega_{\mathcal{G}}$ and the identity (A.77),

$$\frac{\Delta\mathbf{R}_{\mathcal{BG}}}{\Delta t} = \boldsymbol{\omega}_{\mathcal{G}} \times \mathbf{R}_{\mathcal{GB}} \Leftrightarrow \begin{cases} \dot{\mathbf{R}}_{\mathcal{BG}} = [\boldsymbol{\omega}_{\mathcal{G}}]_\times \mathbf{R}_{\mathcal{BG}} \\ \dot{\mathbf{R}}_{\mathcal{BG}} = \mathbf{R}_{\mathcal{BG}}[\boldsymbol{\omega}_{\mathcal{B}}]_\times \\ \dot{\mathbf{R}}_{\mathcal{GB}} = -\mathbf{R}_{\mathcal{GB}}[\boldsymbol{\omega}_{\mathcal{G}}]_\times \\ \dot{\mathbf{R}}_{\mathcal{GB}} = -[\boldsymbol{\omega}_{\mathcal{B}}]_\times \mathbf{R}_{\mathcal{GB}} \end{cases}. \quad \text{(A.92)}$$

B

Controller appendix

B.1 Set-point weighed PID

The simple and useful model independent proportional-integral-derivative *(PID)* controller is here derived in discrete time, partly as a reference for the other controllers and also due to it requiring little computational power in comparison to the other considered controllers. The controller implements set-point weighing, conditional anti windup and a low-pass filter on the derivative term, putting an upper bound on the derivative gain. In this thesis, any reference to *PID*-type controllers will always use this form, where for instance the *PD* controller simply omits the I term in the control law, effectively letting $T_i \to \infty$.

We first define an arbitrary *SISO* system in the Laplace domain, with a control signal $\mathcal{L}\left\{u(t)\right\}_s = U(s)$, a set-point reference $\mathcal{L}\left\{r(t)\right\}_s = R(s)$ and a measurement $\mathcal{L}\left\{y(t)\right\}_s = Y(s)$. The controller can then be written as a map of the measurement error to the control signal as

$$U(s) = K\Big[(\beta R(s) - Y(s)) + \frac{1}{sT_i}(R(s) - Y(s)) + \frac{sT_d}{1 + sT_d/N}(\gamma R(s) - Y(s))\Big] \tag{B.1}$$

with controller parameters $K, T_i, T_d, N, \beta, \gamma$. Discretising the controller with forward differences for the I-part and backward differences for the D-part yields

$$\begin{cases} P_k = K(\beta r_k - y_k) \\ I_k = I_{k-1} + \frac{Kh}{T_i}(r_k - y_k) \\ D_k = \frac{T_d}{T_d + Nh} D_{k-1} - \frac{KT_dN}{T_d + Nh}\Big[\gamma(r_k - r_{k-1}) - (y_k - y_{k-1})\Big] \\ u_k = P_k + I_k + D_k \end{cases} \tag{B.2}$$

where set-point weighing is introduced by γ, β and the filter coefficient N defines the maximum derivative gain. Here, the conditional anti windup is

also included, saturating the control signal to a bound $[u_{min}, u_{max}] \in \mathbb{R}$, such that

$$u_k = \begin{cases} u_{max} & \text{if} \quad u_k > u_{max} \\ u_k & \text{if} \quad u_k \in [u_{min}, u_{max}] \\ u_{min} & \text{if} \quad u_k < u_{min} \end{cases} \tag{B.3}$$

the integral part is updated only when $u \in [u_{min}, u_{max}]$.

B.2 MRAC with MIT synthesis

The model reference adaptive controller *(MRAC)* used in the rotor and inner loops is here derived using the *MIT*-rule [Åström and Murray, 2010]. The main purpose of the controller is to get some unknown system, $G_p(s)$, which could be linear or non-linear, to behave as a well known reference model $G_m(s)$. Here we consider *SISO* continuous time systems defined by a differential operator p and the corresponding model error

$$e(t) = y(t) - y_m(t) = G_p(p)u(t) - G_m(p)u_c(t). \tag{B.4}$$

the objective is to drive the model error to zero by means of an adaptive control law

$$u(t) = \theta_u u_c(t) - \theta_y y(t) = \boldsymbol{\theta}^T \begin{bmatrix} u_c(t) \\ y(t) \end{bmatrix} \tag{B.5}$$

where the adaptive gain matrix $\boldsymbol{\theta} \in \mathbb{R}^{2\times 1}$. For this purpose, we define a cost function

$$J(\boldsymbol{\theta}) = \frac{1}{2}e^2(\boldsymbol{\theta}) \geq 0 \quad \forall \boldsymbol{\theta} \tag{B.6}$$

and try to find conditions on the sensitivity derivatives so that the adaptive gains change to minimise the cost function according to the MIT rule,

$$\frac{\partial \theta_i}{\partial t} = -\Gamma_i \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_i} = -\Gamma_i e(\boldsymbol{\theta}) \frac{\partial e(\boldsymbol{\theta})}{\partial \theta_i}. \tag{B.7}$$

Using (B.4) and (B.5) we find that

$$e(t) = \Big(\frac{G_p(p)\theta_u}{1 + G_p(p)\theta_y} - G_m(p) \Big) u_c(t) \triangleq (\theta_u G_{cl}(p) - G_m(p))u_c(t) \tag{B.8}$$

and differentiating this expression with respect to the adaptive gains yields

$$\begin{cases} \dfrac{\partial e}{\partial \theta_u} = \dfrac{G_p(p)}{1 + G_p(p)\theta_y} u_c(t) = G_{cl} u_c(t) \\ \dfrac{\partial e}{\partial \theta_y} = -\dfrac{G_p(p)\theta_u}{(1 + G_p(p)\theta_y)^2} u_c(t) = -\dfrac{G_p(p)}{1 + G_p(p)\theta_y} u_c(t) = -G_{cl}(p) y(t) \end{cases} \tag{B.9}$$

then in the *MRAC* synthesis some approximation needs to be made in order to find the transfer function G_{cl}. In some cases it is feasible to assume $G_{cl} \approx G_m$, but here, in our the cases we consider, the numerator of the process $B_p(p)$ is constant. Denoting the reference models monic numerator polynomial by $A_m(p)$ with a degree N, and the polynomial $A'_m(p) = A_m(p) - p^N$, a more valid approximation is then to let

$$G_{cl} = \frac{B_p(p)}{A_p(p) + \theta_y B_p(p)} \approx \frac{A'_m(p)}{A_m(p)} \tag{B.10}$$

Thus resulting in the parameter update laws

$$\begin{cases} \dfrac{\partial \theta_u}{\partial t} = -\Gamma_u e(t) \dfrac{\partial e}{\partial \theta_u} = -\Gamma_u e(t) \frac{A'_m(p)}{A_m(p)} u_c(t) \\ \dfrac{\partial \theta_y}{\partial t} = -\Gamma_y e(t) \dfrac{\partial e}{\partial \theta_y} = \Gamma_y e(t) \frac{A'_m(p)}{A_m(p)} y(t) \end{cases} \tag{B.11}$$

The design specifications in this *MRAC* derivation are the adaptive gains Γ_u, Γ_y and the reference model $G_m(s)$ which needs to be chosen with care in order for the assumption $G_p(p) \approx G_m(p)$ to hold. It should be noted that this derivation is in continuous time, and that some discretisation of the process and model will have to be done in a real time implementation. In our implementation, discretisation of the transfer functions is done by *ZOH*, and the integration of $\dot{\theta}_u, \dot{\theta}_y$ is discretised using forward differences (see Figure B.1).

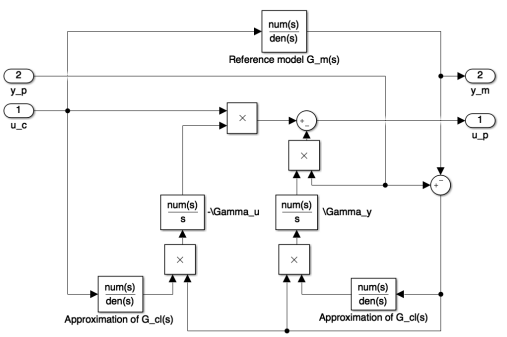

Figure B.1 Simulink implementation of the continuous-time MRAC.

B.3 Linear quadratic reaulators

For the sake of generality, we consider a non-linear *MIMO* system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{B.12}$$

which can be efficiently linearised around a given state $\mathbf{x}_0$, $\mathbf{u}_0$. This system is then sampled and will be referred to in its discrete time form

$$\tilde{\mathbf{x}}_{k+1} = \mathbf{A}_0\tilde{\mathbf{x}}_k + \mathbf{B}_0\tilde{\mathbf{u}}_k. \tag{B.13}$$

By the standard discrete-time approach presented in [Reist and Tedrake, 2010], we set out to minimise the cost function

$$J(\mathbf{x}_0, \mathbf{u}_0) = \sum_{n=0}^{\infty} \left[\tilde{\mathbf{x}}_n^T\mathbf{Q}\tilde{\mathbf{x}}_n + \tilde{\mathbf{u}}_n^T\mathbf{R}\tilde{\mathbf{u}}_n\right]. \tag{B.14}$$

assuming $\mathbf{Q} = \mathbf{Q}^T \geq 0, \mathbf{R} = \mathbf{R}^T > 0$. The solution to the optimisation problem can found through dynamical programming, where we define $\mathbf{S}_0 = \mathbf{S}_0^T$ as the positive definite solution to the associated Riccati equation,

$$\mathbf{A}_0^T\mathbf{S}_0\mathbf{A}_0 - \mathbf{S}_0 - \mathbf{A}_0^T\mathbf{S}_0\mathbf{B}_0(\mathbf{B}_0^T\mathbf{S}_0\mathbf{B}_0 + \mathbf{R})^{-1}\mathbf{B}_0^T\mathbf{S}_0\mathbf{A}_0 + \mathbf{Q} = 0. \tag{B.15}$$

The optimal control law for (B.14) is then formed by

$$\tilde{\mathbf{u}}_k^* = -(\mathbf{R} + \mathbf{B}_0^T\mathbf{S}_0\mathbf{B}_0)^{-1}\mathbf{B}_0^T\mathbf{S}\mathbf{A}_0\tilde{\mathbf{x}}_k \triangleq -\mathcal{K}_0\tilde{\mathbf{x}}_k. \tag{B.16}$$

Here a few things should be noted. First, if the point $(\mathbf{x}_0, \mathbf{u}_0)$ is fixed for all times, the linearised matrices will be constant for all times and we only need to compute the *LQR*-gain $\mathcal{K}_0$ once. This is the standard time invariant formulation, and the gain matrix can easily be computed by solving the Riccati equation using Matlab's `dlqr` offline. The second thing to note is that the linearised dynamics are unaffected by the positional states of the quadcopter. As such, we could here simply linearise the system around a stable hovering point $(\mathbf{x}_0, \mathbf{u}_0)$ and form a controller which can be run with computational ease regardless of the goal state positions and velocities. It comes with the drawback of having to operate with angular dynamics close to the goal state at all times, thus putting great restrictions on how aggressively we may fly.

To support more aggressive flights, a recursive, discrete-time varying, finite horizon, iterative *LQR* scheme is also presented as originally derived in [Reist and Tedrake, 2010]. Here we set up a cost function similar to the time invariant case (B.14) under the same assumptions, but with a finite horizon and a punishing cost on the terminal state $\mathbf{Q}_N = \mathbf{F}^T > 0$, such that

$$J(\tilde{\mathbf{x}}_k) = \tilde{\mathbf{x}}_N^T\mathbf{Q}_N\tilde{\mathbf{x}}_N + \sum_{n=0}^{k+N-1} \left[\tilde{\mathbf{x}}_n^T\mathbf{Q}\tilde{\mathbf{x}}_n + \tilde{\mathbf{u}}_n^T\mathbf{R}\tilde{\mathbf{u}}_n\right] \tag{B.17}$$

where we know and evaluate state trajectory $(\mathbf{x}_n, \mathbf{u}_n)$ for N equidistant times $n \in [k, k+N-1]$ to predict the time-varying governing dynamics $(\mathbf{A}_n, \mathbf{B}_n)$. The optimal cost-to-go at a time k,

$$J(\mathbf{x}_k, \mathbf{u}_k) = \tilde{\mathbf{x}}_k^T \mathbf{S}_k \tilde{\mathbf{x}}_k \tag{B.18}$$

is then minimised by applying the update

$$\mathbf{S}_k = \mathbf{A}_k^T \mathbf{S}_{k+1} \mathbf{A}_k - \mathbf{A}_k^T \mathbf{S}_{k+1} \mathbf{B}_k (\mathbf{B}_k^T \mathbf{S}_{k+1} \mathbf{B}_k + \mathbf{R})^{-1} \mathbf{B}_k^T \mathbf{S}_{k+1} \mathbf{A}_k + \mathbf{Q}. \tag{B.19}$$

with the boundary condition $\mathbf{S}_N = \mathbf{Q}_N$. The optimal control policy at time k becomes

$$\tilde{\mathbf{u}}_k^* = -(\mathbf{R} + \mathbf{B}_k^T \mathbf{S}_{k+1} \mathbf{B}_k)^{-1} \mathbf{B}_k^T \mathbf{S}_{k+1} \mathbf{A}_k \tilde{\mathbf{x}}_k \triangleq -\mathcal{K}_k \tilde{\mathbf{x}}_k. \tag{B.20}$$

C

State estimation appendix

C.1 Cramer-Rao lower bound in TOA

Consider the *TOA* case, where a communication time delay, $\hat{t}_i$, is measured between a robot located at a point $\mathbf{p} \in \mathbb{R}^3$ and an anchor located $\mathbf{p}_i \in \mathbb{R}^3$ a described in **Section 6.4** and **Appendix C.2**. If the implemented *SDS-TWR* protocol removes clock offsets and drifts in the robot relative the anchor, the distance between the robot and the anchor may be written

$$\hat{d}_i = c\hat{t}_i = c\Big(\frac{||\mathbf{p} - \mathbf{p}_i||_2}{c} + w_i\Big) = d_i + n_i \tag{C.1}$$

denoting $d_i = ||\mathbf{p} - \mathbf{p}_i||_2$, where $n_i \sim N(0, {}^d\sigma_i^2)$ and ${}^d\sigma_i = \sqrt{\mathbb{V}[d_i + n_i]} = \sqrt{\mathbb{V}[ct_i + cw_i]} = |c|\sqrt{\mathbb{V}[w_i]} = c \cdot {}^t\sigma_i$. With a total of N anchors, we define

$$\hat{\mathbf{d}} \triangleq \begin{bmatrix}\hat{d}_1 & \cdots & \hat{d}_N\end{bmatrix}^T = \begin{bmatrix}d_1 & \cdots & d_N\end{bmatrix}^T + \begin{bmatrix}n_1 & \cdots & n_N\end{bmatrix}^T = \mathbf{d} + \mathbf{n} \in \mathbb{R}^{N\times 1}. \tag{C.2}$$

Now, by the previous assumption of the measurement noise being gaussian and independent across the anchors, the conditional distribution of N distance measurements given a position $\mathbf{p}$ may be written

$$P(\hat{\mathbf{d}}|\mathbf{p}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}({}^d\sigma_i)} \exp\Big(-\frac{1}{2({}^d\sigma_i^2)}(\hat{d}_i - d_i)^2\Big) \tag{C.3}$$

as shown in [Gentile et al., 2013]. We are now ready to express the Cramer-Rao lower bound (*CRLB*), a lower bound on the variance of any unbiased estimate $\mathbb{E}[\hat{\mathbf{p}}] = \hat{\mathbf{p}}$, given by

$$\mathbb{E}[(\hat{\mathbf{p}} - \mathbf{p})(\hat{\mathbf{p}} - \mathbf{p})^T] \geq \mathbf{I}^{-1}(\mathbf{p}).\text{,} \tag{C.4}$$

with $\mathbf{I}(\mathbf{p})$ being the Fisher Information Matrix (*FIM*) on the positional measurements [Yin et al., 2013]. The *FIM* is given by

$$\mathbf{I}(\mathbf{p}) = \mathbb{E}\Big[\Big(\nabla_{\mathbf{p}} \ln\big(f(\hat{\mathbf{d}}|\mathbf{p})\big)\Big)^2\Big] \tag{C.5}$$

where $f(\hat{\mathbf{d}}|\mathbf{p})$ denotes the conditional probability density function of the distribution $P(\hat{\mathbf{d}}|\mathbf{p})$ defined in (C.3). With our assumptions of the estimator being unbiased and the noise being gaussian and identically independently distributed, the chain rule yields

$$\mathbf{I}(\mathbf{p}) = \frac{\partial \mathbf{d}}{\partial \mathbf{p}} \mathbb{E}\Big[\Big(\nabla_{\mathbf{p}} \ln\big(f(\hat{\mathbf{p}}|\mathbf{p})\big)\Big)\Big(\nabla_{\mathbf{p}} \ln\big(f(\hat{\mathbf{p}}|\mathbf{p})\big)\Big)^T\Big] \frac{\partial \mathbf{d}^T}{\partial \mathbf{p}} = \frac{\partial \mathbf{d}}{\partial \mathbf{p}} \mathbf{R}^{-2} \frac{\partial \mathbf{d}^T}{\partial \mathbf{p}} \tag{C.6}$$

where, $\mathbf{R} \in \mathbb{R}^{N \times N}$ is a diagonal matrix with ${}^t\sigma_i$ on the diagonal. Differentiating the measurement equation (C.2) and removal of clock drift,

$$\frac{\partial \mathbf{d}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial d_1}{\partial x} & \dots & \frac{\partial d_N}{\partial x} \\ \frac{\partial d_1}{\partial y} & \dots & \frac{\partial d_N}{\partial y} \\ \frac{\partial d_1}{\partial z} & \dots & \frac{\partial d_N}{\partial z} \end{bmatrix} = \begin{bmatrix} \frac{x-x_1}{d_1} & \dots & \frac{x-x_N}{d_N} \\ \frac{y-y_1}{d_1} & \dots & \frac{y-y_N}{d_N} \\ \frac{z-z_1}{d_1} & \dots & \frac{z-z_N}{d_N} \end{bmatrix} \tag{C.7}$$

with $d_i = ||\mathbf{p} - \mathbf{p}_i||_2$, from which the inverse *FIM* is easily computed. This result is validated by alternate derivations yielding similar results as in $\mathbb{R}^2$ [Kaune, 2012].

C.2 Robust protocols considering clock drift

Many ranging protocols of varying complexity have been considered for time of flight based *UWB* localisation in previous research [Kim, 2009] [Jiang and Leung, 2007]. In the time of flight case supported in the *DW1000* chip, the time stamps when packets are transmitted, T^{Tx}, and received, T^{Rx}, are logged through a process of sending packets between the anchor and the robot, whose clocks need not be synchronised (see Figure C.1). Communication is done by sending a polling packet (blue) containing data with an anchor identifier, $\{i\}$, an answer packet (red) and a final packet (green) containing the anchor identifier as well as a set of timestamps $\{i, T_1^{Tx}, T_1^{Rx}, T_2^{Tx}, T_2^{Rx}, T_3^{Tx}, T_3^{Rx}\}$ which is retrieved in the robot (see Figure C.1). With this information available, we will proceed to discuss some common approaches of estimating the *TOA* and implicitly *TDOA* time of flight, $\hat{t}_i$, between the robot and the anchor i.

The approaches will be compared in terms of robustness to (i) constant clock offsets $\Theta_i(T_1^{Tx}) = \Theta_{i,0} \neq 0$ and (ii) clock drift $\Theta_i'(t) \neq 0$ on the ranging time interval. The concept of variable relative time may seem dubious, as the time surely moves just as fast in the two systems, but it occurs naturally due to the offsets in the oscillator crystals relative their nominal value. For this discussion, we denote the nominal clock frequency of the *DW1000* chip as f_c $[Hz]$, where the frequency of the clock in the anchor i is given by $f_{A,i} = (1 + e_{A,i}(t))f_c$ and similarly, the frequency of the clock in the robot is

given by $f_R = (1+e_R(t))f_c$. Locally, on $t \in [T_1^{Tx}, T_3^{Rx}]$, we assume $e_R(t) \equiv e_R$ and $e_{A,i}(t) \equiv e_{A,i}$ are constant, resulting in the clock offset

$$\Theta_i(t) = (e_{A,i} - e_R)t + \Theta_{i,0} \quad \forall t \in [T_1^{Tx}, T_3^{Rx}]. \tag{C.8}$$

This approximation is not essential, but greatly simplifies the estimator error analysis.

One way ranging In order to determine the estimated time of flight, $\hat{t}_i$, an intuitive approach is to use a simple one-way ranging (OWR). In the robot time scale, the measurement equation is then

$$\hat{t}_i = t_i^{(1)} = T_1^{Rx} + \Theta_i(t) - T_1^{Tx} \tag{C.9}$$

which only uses a single packet, enabling higher sample rates at the cost of making the system incredibly sensitive to the unknown clock offsets and drift. Despite this fact, previous work has been done with successful implementations where the relative clock skew between the anchors has been modelled as a random walk process and been included in a Kalman filter formulation [Ledergerber et al., 2015].

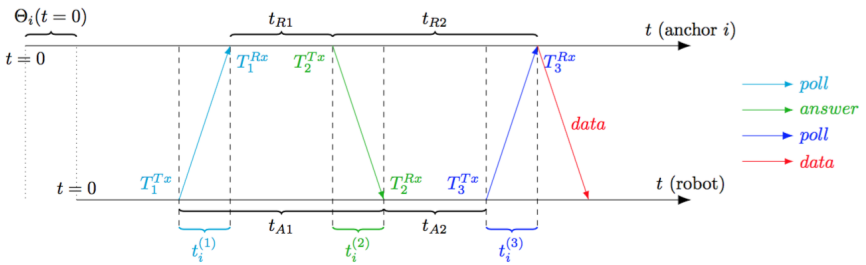

Figure C.1 Two way communication with polling (cyan), answer (green) a second polling packet (blue) and a final data packet (red) sending the timestamp data to the *UAV*. The timestamps can be combined in various ways to determine the time of flight, which in practice is assumed to be $t_i^{(1)} \approx t_i^{(2)} \approx t_i^{(3)}$ as $\hat{t}_i \ll t_{A2} \approx t_{R1}$.

Two way ranging A slightly more robust and common approach is to use the two way ranging scheme (TWR), which completely removes the need for clock synchronisation between the robot and the anchor by letting

$$\hat{t}_i = \frac{t_i^{(1)} + t_i^{(2)}}{2} = \frac{T_2^{Rx} - T_1^{Tx} + (T_2^{Rx} + \Theta_i - T_1^{Tx} - \Theta_i)}{2} = \frac{t_{A1} - t_{R1}}{2}. \tag{C.10}$$

Robust to constant offsets, the method is still sensitive to errors induced by clock drift, as shown in the estimation error

$$\hat{t}_i - t_i = e_{A,i} f_i + \frac{1}{2}(e_{A,i} - e_R) t_{R1} \approx \frac{1}{2}(e_{A,i} - e_R) t_{R1} \tag{C.11}$$

where we have assumed that $t_i \ll t_{R,1}$. The analysis shows that the *TWR* protocol yields an estimation error which depends on $e_{A,i}$ and e_R where $\hat{t}_i - t_i \propto t_{R1}$ if $\Theta_i'(t) \neq 0$ on the ranging time interval. In conclusion, while sensitive to clock drift, the method can be made more robust by decreasing t_{R1} and is a great improvement on the *OWR* protocol due to the cancellation of any constant clock offsets.

Symmetric double sided two way ranging A way to combat the issue of drift is to instead use the moving average symmetric double sided (*SDS*) *TWR* protocol [Emami, 2013]., where a total of three packages are sent between the anchors and the robot, such that

$$\hat{t}_i = \frac{1}{2}\Big(\frac{t_i^{(1)} + t_i^{(2)}}{2} + \frac{t_i^{(2)} + t_i^{(3)}}{2}\Big) = \frac{t_{A1} - t_{R1} + t_{R2} - t_{A2}}{4} \tag{C.12}$$

In some literature, the assumption $t_{A1} + t_{A2} = t_{R1} + t_{R2}$ is made, allowing the time-delay to be written

$$\hat{t}_i = \frac{t_{A1} t_{R2} - t_{R1} t_{A2}}{t_{A1} + t_{R2} + t_{R1} + t_{A2}}. \tag{C.13}$$

Proceeding with the error analysis, see that

$$\hat{t}_i - t_i = \frac{1}{2}(e_{A,i} + e_R) t_i + \frac{1}{4}(t_{R1} - t_{A2})(e_{A,i} - e_R) \approx \frac{1}{4}(t_{R1} - t_{A2})(e_{A,i} - e_R) \tag{C.14}$$

if we similarly to the *TWR* case assume that t_i is small. Here we see that even if $\Theta_i'(t) \neq 0$, the estimation error can be made very small by letting $t_{R1} \approx t_{A2}$. Naturally, the approximation of linear frequency drift is more valid for smaller times t_{R1}, t_{A2}. In general, we cannot guarantee that $t_{R1} \approx t_{A2}$ as the times depend on the clock, but recalling equation (C.11), we can say that if

$$|t_{R1} - t_{A2}| < |2t_{R1}| \tag{C.15}$$

the *SDS-TWR* should be used over the *TWR* protocol. In practice, this condition will be met at virtually all times and the protocol, as given by equation (C.12), is therefore implemented using the time division multiple access *(TDMA)*.

C.3 The Extended kalman Filter

There exist many different derivations for the *EKF*. In this section, we present a brief but intuitive version, referring to [Terejanu, 2008] for more details. Consider the discrete time system

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \in \mathbb{R}^{N\times 1} \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k) \in \mathbb{R}^{M\times 1} \end{aligned} \tag{C.16}$$

with gaussian noise $\mathbf{w}_k$ and $\mathbf{v}_k$ with the covariance properties

$$\mathbb{E}\left\{ \begin{bmatrix} \mathbf{w}_k \\ \mathbf{v}_k \end{bmatrix} \right\} = \mathbf{0} \qquad \mathbb{E}\left\{ \begin{bmatrix} \mathbf{w}_k \\ \mathbf{v}_k \end{bmatrix} \begin{bmatrix} \mathbf{w}_k \\ \mathbf{v}_k \end{bmatrix}^T \right\} = \begin{bmatrix} \mathbf{Q} & 0 \\ 0 & \mathbf{R} \end{bmatrix} \tag{C.17}$$

for some positive definite $\mathbf{Q} \in \mathbb{R}^{N\times N}$, $\mathbf{R} \in \mathbb{R}^{M\times M}$, thereby assuming uncorrelated state- and measurement noise. In the *EKF*, just as in the standard Kalman filter, state vector probability density function, $p(\mathbf{x}_k|\mathcal{S}_k)$ conditioned by the sequence of past measurements and control signals $\mathcal{S}_k = \{\mathbf{y}_0, \cdots, \mathbf{y}_k, \mathbf{u}_0, \cdots, \mathbf{u}_{k-1}\}$, is propagated through time in two steps. As the system (C.16) is assumed to be non-linear, the *EKF* uses a first order multivariate Taylor-approximation to predict a future state, $\mathbf{x}_k^f$, which is then corrected through filtering using a Kalman gain resulting in the sub-optimal estimate $\hat{\mathbf{x}}_k$. For future reference, we define the system Jacobians

$$\mathbf{F}_k = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \mathbf{0}} \qquad \mathbf{H}_k = \frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{v})}{\partial \mathbf{x}}\Big|_{\mathbf{x}_k^f, \mathbf{u}_k, \mathbf{0}} \tag{C.18}$$

where $\mathbf{F}_k$ is well defined and may be expressed analytically at all times in the quaternion dynamics, and $\mathbf{H}_k$ depends on the available measurements. In addition, if assuming that noise may be non-additive, we let

$$\mathbf{W}_k = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{w}}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \mathbf{v}_k} \qquad \mathbf{V}_k = \frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{v})}{\partial \mathbf{v}}\Big|_{\mathbf{x}_k^f, \mathbf{u}_k, \mathbf{w}_k}. \tag{C.19}$$

Prediction

Let $\hat{\mathbf{x}}_{k-1} = \mathbb{E}\left[\mathbf{x}_{k-1}|\mathcal{S}_{k-1}\right]$ be the optimal estimate at the time $k-1$ with a corresponding covariance matrix $\mathbf{P}_{k-1}$. Taylor expanding the non-linear transition function around this optimal estimate and assuming $\mathbb{E}\left[\mathbf{v}_{k-1}\right] = 0$ yields

$$\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) = \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) + \mathbf{F}_{k-1}\mathbf{e}_{k-1} + \mathbf{W}_{k-1}\mathbf{w}_{k-1} + O(||\cdot||_2^2) \tag{C.20}$$

where $\mathbf{e}_{k-1} = \mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}$. The expected value of the forecasted state conditioned by $\mathcal{S}_{k-1}$ is then

$$\mathbf{x}_k^f = \mathbb{E}\left[\mathbf{x}_k|\mathcal{S}_{k-1}\right] \approx \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) + \mathbb{E}\left[\mathbf{F}_{k-1}\mathbf{e}_{k-1}|\mathcal{S}_{k-1}\right] \approx \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) \tag{C.21}$$

This allows the error in the state forecast to be expressed as

$$\mathbf{e}_k^f = \mathbf{x}_k - \mathbf{x}_k^f = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) - \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) \approx \mathbf{F}_{k-1}\mathbf{e}_{k-1} + \mathbf{W}_{k-1}\mathbf{w}_{k-1}, \tag{C.22}$$

and the optimal covariance prediction is then

$$\mathbf{P}_k^f = \mathbb{E}\left[\mathbf{e}_k^f (\mathbf{e}_k^f)^T\right] \approx \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{W}_{k-1}\mathbf{Q}\mathbf{W}_{k-1}^T. \tag{C.23}$$

Correction

In order to correct the prediction, we seek best the unbiased estimate $\hat{\mathbf{x}}$ in a least-squares sense. This can be done done by assuming that

$$\hat{\mathbf{x}}_k = \mathbf{b} + \mathbf{K}_k \mathbf{y}_k \tag{C.24}$$

which, assuming unbiasedness in the estimator, yields

$$\begin{aligned}
0 =& \mathbb{E}[\mathbf{x}_k - \hat{\mathbf{x}}_k | \mathcal{S}_k] \\
0 =& \mathbb{E}[(\mathbf{x}_k^f + \mathbf{e}_k^f) - (\mathbf{b} + \mathbf{K}_k \mathbf{y}_k) | \mathcal{S}_k] \\
0 =& \mathbb{E}[(\mathbf{x}_k^f + \mathbf{e}_k^f) - (\mathbf{b} + \mathbf{K}_k \mathbf{h}(\mathbf{x}_k, \mathbf{0}) + \mathbf{K}_k \mathbf{v}_k) | \mathcal{S}_k] \\
0 =& \mathbf{x}_k^f - \mathbf{b} - \mathbf{K}_k \mathbb{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{0}) | \mathcal{S}_k] \\
\mathbf{b} =& \mathbf{x}_k^f - \mathbf{K}_k \mathbb{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{0}) | \mathcal{S}_k]
\end{aligned} \tag{C.25}$$

By the assumption of unbiasedness, $\mathbb{E}[\mathbf{e}_k^f | \mathcal{S}_k] = 0$, implying that

$$\mathbb{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{0}) | \mathcal{S}_k] \approx \mathbf{h}(\mathbf{x}_k^f) + \mathbf{H}_k \mathbb{E}[\mathbf{e}_k^f | \mathcal{S}_k] \approx \mathbf{h}(\mathbf{x}_k^f) \tag{C.26}$$

With this result, equation (C.24) may be simplified using (C.25),

$$\hat{\mathbf{x}}_k = \mathbf{b} + \mathbf{K}_k \mathbf{y}_k \approx \mathbf{x}_k^f + \mathbf{K}_k (\mathbf{y}_k - \mathbf{h}(\mathbf{x}_k^f, \mathbf{0})) \tag{C.27}$$

The estimator error is then

$$\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{x}}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{F}_{k-1}\mathbf{e}_{k-1} + \mathbf{K}_k \mathbf{V}_k \mathbf{v}_k \tag{C.28}$$

from which we may form the posterior covariance as

$$\mathbf{P}_k = \mathbb{E}\left[\mathbf{e}_k (\mathbf{e}_k)^T\right] = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{P}_k^f (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{V}_k \mathbf{R} \mathbf{V}_k^T \mathbf{K}_k^T \tag{C.29}$$

We now set out to minimise the error $\mathbf{e}_k$ in a squared sense with respect to the gain matrix $\mathbf{K}_k$,

$$\min_{\mathbf{K}_k} (||\mathbf{e}_k||_2^2). \tag{C.30}$$

With the assumption of uncorrelated noise, is equivalent to minimising the trace of the posterior covariance matrix $\mathbf{P}_k = \mathbb{E}[\mathbf{e}_k(\mathbf{e}_k)^T]$, and an extremal point is given at

$$0 = \frac{\partial tr(\mathbf{P}_k^f)}{\partial \mathbf{K}_k} \Rightarrow \mathbf{K}_k = \mathbf{P}_k^f(\mathbf{H}_k)^T[\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{V}_k\mathbf{R}\mathbf{V}_k^T]^{-1} \tag{C.31}$$

which is indeed a minimiser as the second partial derivative with respect to the trace of the forecasted covariance is positive semi-definite. Insertion of (C.31) in (C.29) yields

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f \tag{C.32}$$

greatly simplifying the covariance correction at the cost of numerical stability, as the symmetry of the covariance matrix is less likely to be preserved.

C.4 Multi-camera LS regression

Consider a set of i cameras with by their static positions in the room, $\mathbf{c}_i \in \mathbb{R}^3$, and unit vectors $\mathbf{n}_i \in \mathbb{R}^3$ such that the quadcopter resides on the lines $\mathbf{l}_i = \mathbf{c}_i + t\mathbf{n}_i, \quad t \in \mathbb{R}$ (see Figure C.2).

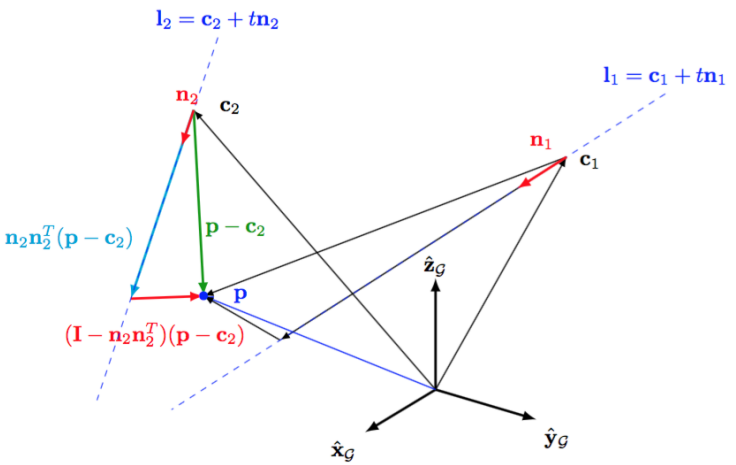

Figure C.2 The coordinates considered for the *LS* regression.

The closest distance squared between a point $\mathbf{p}$ and a line $\mathbf{l}_i$ may then be written in terms of the idempotent projector $\mathbf{P}_i = \mathbf{I} - \mathbf{n}_i\mathbf{n}_i^T$, which can be shown to satisfy $\mathbf{P}_i^2 = \mathbf{P}_i$. In the two-norm, this distance is

$$\begin{aligned} ||\mathbf{p} - \mathbf{l}_i||_2^2 &= ||(\mathbf{p} - \mathbf{c}_i) - ((\mathbf{p} - \mathbf{c}_i)^T\mathbf{n}_i)\mathbf{n}_i||_2^2 && \text{(C.33)} \\ &= ||(\mathbf{p} - \mathbf{c}_i) - \mathbf{n}_i\mathbf{n}_i^T(\mathbf{p} - \mathbf{c}_i)||_2^2 && \text{(C.34)} \\ &= ||\mathbf{P}_i(\mathbf{p} - \mathbf{c}_i)||_2^2 && \text{(C.35)} \\ &= (\mathbf{p} - \mathbf{c}_i)^T\mathbf{P}_i\mathbf{P}_i(\mathbf{p} - \mathbf{c}_i)] && \text{(C.36)} \\ &= (\mathbf{p} - \mathbf{c}_i)^T\mathbf{P}_i(\mathbf{p} - \mathbf{c}_i) && \text{(C.37)} \end{aligned}$$

The total cost of an arbitrary point $\mathbf{p}$ for a total of N lines can then be defined as

$$J(\mathbf{p}) = \sum_{i=1}^{N} ||\mathbf{p} - \mathbf{l}_i||_2^2 = \sum_{i=1}^{N} (\mathbf{c}_i - \mathbf{p})^T(\mathbf{I} - \mathbf{n}_i\mathbf{n}_i^T)(\mathbf{c}_i - \mathbf{p}) \qquad \text{(C.38)}$$

which, when differentiated with respect to $\mathbf{p}$ gives an extremal point at

$$\frac{\partial J(\mathbf{p})}{\partial \mathbf{p}} = -2(\mathbf{I} - \mathbf{n}_i\mathbf{n}_i^T)(\mathbf{c}_i - \mathbf{p}) = 0 \qquad \text{(C.39)}$$

which is indeed a minimum, as

$$\frac{\partial^2 J(\mathbf{p})}{\partial \mathbf{p}^2} = 2(\mathbf{I} - \mathbf{n}_i\mathbf{n}_i^T) = 2\mathbf{P} \qquad \text{(C.40)}$$

is positive semidefinite on account of $\mathbf{P}$ being idempotent. With this result, we simply form the linear system

$$\mathbf{A} = \sum_{i=1}^{N} (\mathbf{I} - \mathbf{n}_i\mathbf{n}_i^T), \qquad \mathbf{b} = (\mathbf{I} - \mathbf{n}_i\mathbf{n}_i^T)\mathbf{c}_i \qquad \text{(C.41)}$$

from which the *LS* estimate of the position can be written

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} ||\mathbf{A}\mathbf{p} - \hat{\mathbf{b}}||_2 = \mathbf{A}^{\dagger}\hat{\mathbf{b}}. \qquad \text{(C.42)}$$

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER´S THESIS

Date of issue
February 2017

Document Number
ISRN LUTFD2/TFRT--6026--SE

Author(s)
Marcus Greiff

Supervisor
Fredrik Bagge Carlson, Dept. of Automatic Control, Lund University, Sweden
Anders Robertsson, Dept. of Automatic Control, Lund University, Sweden
Bo Bernhardsson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle

Modelling and Control of the Crazyflie Quadrotor for Aggressive and Autonomous Flight by Optical Flow Driven State Estimation

Abstract

The master thesis seeks to develop a control system for the Crazyflie 2.0 unmanned aerial vehicle to enable aggressive and autonomous flight. For this purpose, different rigid-body models are considered, differing primarily in their parametrisation of rotation. The property of differential flatness is explored and several means of parametrising trajectories in flat output space are implemented. A new method of rotor control with parameter estimation is developed and geometric controllers are implemented for rigid-body control. Finally, state estimation is accomplished through a scalar-update extended Kalman filter, where information from the internal measurement unit is fused with positional information from camera systems, ultra-wide band systems, optical flow measurements and laser ranging measurements. Capable of sustaining flight with any combination of the previously mentioned sensors, the real-time implementation is showcased using polynomial motion-planning to avoid known obstacles.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-139

Recipient's notes

Security classification

http://www.control.lth.se/publications/