Loss of Control Prediction for Quadrotors with Single Rotor Failure

MSc Thesis

T. Hoppenbrouwer

TU Delft

Loss of Control Prediction for Quadrotors with Single Rotor Failure

MSc Thesis

Thesis report

by

T. Hoppenbrouwer

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on June 15, 2023 at 14:00.

Thesis committee:

Chair:	Dr. ir. C.C. de Visser
Supervisors:	Dr. ir. C.C. de Visser
External Examiner:	Dr. ir. E. Mooij
Committee Member:	Dr. ir. X. Wang
Location:	Faculty of Aerospace Engineering, Delft
Project Duration:	November, 2020 - June, 2023
Student number:	4297628

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Faculty of Aerospace Engineering · Delft University of Technology

Preface

After a very eventful time on a personal level, I am extremely proud and delighted to be able to present my thesis to all readers. Whilst this is an extremely joyous occasion, this moment also marks the end of an amazing era filled with many happy memories as a student at the TU Delft.

I would like to extend my gratitude to Coen de Visser for his excellent guidance, kindness and useful advice and particularly for our many interesting conversations. Furthermore, I would to thank my girlfriend, Amanda, for her patience, thoughtfulness and compassion that has kept me going strong over the course of the last few months. Additionally, I want to express my greatest gratitude to my friends Riemer, Jop and Job, whose mental support has been essential in achieving this milestone. Finally, I would like to thank the staff at the faculty together with the many fellow students and friends who have shared this journey with me for the many valuable lessons and the countless happy memories.

I am delighted to share my results and I sincerely hope that this final milestone as a student proves to be a useful contribution to an incredibly exciting field of science.

Contents

1 Introduction **1**
1.1 Research Objective 1
1.2 Research Questions 1

I Scientific Article **3**

II Closure **42**

2 Conclusion **43**
2.1 Answers to the Sub-Questions. 43
2.2 Answer to Main Question 44

3 Recommendations **46**

References **47**

A Verification **48**
A.1 Unit Verification. 48
A.2 Controller Verification 48
A.3 Prediction Method Verification 49

1 Introduction

This report presents the complete overview of the thesis effort on the prediction of loss of control due to actuator saturation. In this general introduction, the research motivation and objective are presented along with the defining research questions. Part II is the scientific paper which describes the methodology, analyses and findings of the thesis project. Part III presents the preliminary report that provides the literature study and research set up that were defined at the start of the project. Part IV provides a conclusion to the results by means of answers to the research questions, followed by recommendations for further research and improvements to the current methodology.

1.1. Research Objective

Over the past years, numerous fault-tolerant control solutions have been developed to enable safe flight for quadrotors upon single rotor failure.

Lanzon [1] and Mueller [2] first developed controllers that allow damaged quadrotors to hover and regain limited position control, while the application of nonlinear control techniques by Sun [3][4] enables the continuation of high speed flight. FTC designed by Lu [5] incorporates active fault detection and isolation, enabling real world application of quadrotor control under rotor failure[5].

The physical limitations due to rotor loss, however, make the quadrotor very vulnerable to loss of control, as stated by Sun[4]. In case of large deviations from hover condition, upsets can occur that are difficult to recover from.[4] This means that loss of control prevention can be considered critical for safety. [4]

Considering this vulnerability, there is value in the prediction and prevention of loss of control conditions. Although maximum safe flight speeds have been identified in earlier works by Sun et al [3], no detailed analysis into the failure modes has been conducted, nor has a physical indicator of loss of control performance been identified. The main objective of this research is to gain a proper understanding of these physical principles behind loss-of-control occurrences for quadrotors for a single rotor failure case and to develop a loss-of-control prediction method that can be used in prevention of such occurrences.

1.2. Research Questions

Given the literature gap and the main research objective, a research question has been defined that encompasses the scope of the research effort. This main research question has been divided into five sub-questions that reflect the different facets of the research effort.

> **Main Question:** How can Loss-of-Control occurrence theoretically be predicted using control theory for quadrotors experiencing single rotor failure?

The five sub-questions define the individual milestones of the research effort and are more or less answered sequentially as the research progresses and address the main unknowns that exist regarding this topic. The subquestions are further specified with additional sub-questions to elaborate the research effort in more detail.

RQ1: : How can a loss-of-control scenario be defined?

- What are the existing definitions of a loss-of-control scenario for a quadrotor experiencing rotor failure?
- Are these definitions sufficient, given observations in experiments and simulation?

RQ2: : How does the rotor failure impact the stability of the quadrotor?

- How can the theoretical stability of the quadrotor controller be determined?
- What is the theoretical domain of attraction for the quadrotor fault-tolerant control system upon single rotor failure?
- What factors influence the stability of the quadrotor upon rotor failure?

RQ3: What factors are the leading causes of Loss-of-Control?

- What external factors contribute to the the occurrence of a loss-of-control situation?
- How can these factors lead to a loss-of-control situation?
- What scenarios or trajectories place the quadrotor and its control system at risk of a loss of control?

RQ4: How can the occurrence of loss-of-control scenarios be predicted?

- Are there observable variables that are indicative of the loss-of-control performance?
- If so, are there observable patterns that indicate the onset of loss-of-control before its actual occurrence?
- Are these patterns visible for all of the envisioned loss-of-control scenarios of *RQ3*?

RQ5: How can loss-of-control be prevented?

- Can safe constraints be defined on the flight envelope?
- Can the onset of loss-of-control be predicted sufficiently early to prevent its occurrence?
- How can these constraints be incorporated into existing fault-tolerant-control algorithms for quadrotors experiencing single rotor failure?
- How does loss-of-control prevention impact the performance, compared to FTC without such flight envelope protection?

Part I

Scientific Article

Loss-of-Control Prediction for Quadrotors Experiencing Single Rotor Failure using Hedged Virtual Control Signals

T. Hoppenbrouwer*
MSc Student, Control & Simulation, Faculty of Aerospace Engineering, Delft University of Technology

Upon rotor loss, fault-tolerant quadrotors are subjected to a severe restriction of the set of attainable virtual controls, making them vulnerable to loss of control at high speed flight. Due to the high nonlinearity of the reduced attitude controller and incremental nature of the fault-tolerant controller, determination of the safe flight envelope is exceedingly complex. To prevent loss-of-control, a novel method is presented to detect stability degradation due to actuator saturation that can provide an early warning of the onset of loss-of-control. This research presents an analysis of the effects of actuator dynamics and rotor saturation on the stability of fault-tolerant quadrotor control. Expanding on this analysis, a method is presented to predict loss-of-control, prior to its occurrence.

Nomenclature

Kinematic Variables

$\boldsymbol{F}^B$	Forces in the body frame of reference..
$\boldsymbol{F}^I$	Forces in the inertial frame of reference..
$\boldsymbol{F}_a$	Aerodynamic forces.
$\boldsymbol{G}_M$	Control effectiveness matrix with respect to body frame moments.
$\boldsymbol{M}^B$	Moments in the body frame of reference.
$\boldsymbol{M}_G^B$	Gyroscopic moments in the body frame of reference.
$\boldsymbol{M}_a^B$	Aerodynamic moments in the body frame of reference.
$\boldsymbol{R}$	Rotation matrix between the body and inertial frame of reference.
T	Thrust magnitude along the body frame z-axis..
$\boldsymbol{V}^I$	Velocity vector defined in the inertial frame of reference.
$\boldsymbol{X}^I$	Position expressed in the inertial frame of reference.
$\boldsymbol{a}^I$	Acceleration vector defined in the inertial frame of reference.
$\boldsymbol{h}$	Desired primary axis vector defined in the body frame of reference..
p, q, r	Roll, pitch and yaw rate in the body frame of reference.
$\boldsymbol{u}$	Input vector to quadrotor dynamics..
$\dot{\boldsymbol{\Omega}}^B$	Angular acceleration vector defined in the body frame of reference.
$\boldsymbol{\Omega}^B$	Angular velocity vector defined in the body frame of reference.
$\boldsymbol{\lambda}(\boldsymbol{t})$	Derivative of $\boldsymbol{n}_{des}^I$ transformed to the body frame of reference..
$\boldsymbol{\omega}$	Vector of rotor speeds.

Quadrotor Parameters

A_x, A_y, A_z	Dynamic constants related to the mass-moment of inertia of the quadrotor.
G_p, G_q, G_r	Control effectiveness of individual rotor with respect to roll, pitch and yaw moment.

*MSc Student, Control & Simulation, Faculty of Aerospace Engineering, Delft University of Technology

$\boldsymbol{I}_p$	Mass moment of inertia matrix of an individual rotor.
$\boldsymbol{I_v}$	Mass moment of inertia matrix.
$\boldsymbol{I}$	Identity matrix.
b	Distance between rotor axis and quadrotor center of gravity along the body frame y-axis.
$\boldsymbol{g}$	Gravitational acceleration vector in the inertial frame of reference.
l	Distance between rotor axis and quadrotor center of gravity along the body frame x-axis.
m	Mass.
$\boldsymbol{y}^{(j)}$	Vector of j-th derivative of the outputs.
$\boldsymbol{y}$	Output vector of quadrotor reduced attitude dynamics..
$\boldsymbol{\gamma}$	Yaw damping vector in the body frame of reference.
$\overline{\kappa}$	Rotor thrust coefficient.
ω_B	Control bandwidth of the rotors.
σ	Drag torque coefficient of individual rotors.

Control Variables

$\boldsymbol{A_a}, \boldsymbol{B_a}, \boldsymbol{C_a}$	State-space matrices for the canonical from representation of the extended state transformation $\boldsymbol{\mathcal{T}}(\boldsymbol{x_a})$.
$\boldsymbol{A_c}, \boldsymbol{B_c}, \boldsymbol{C_c}$	State-space matrices for the canonical from representation of state transformation $\boldsymbol{\mathcal{T}}(\boldsymbol{x})$.
$\mathcal{B}(\boldsymbol{x})$	Nonlinear control effectiveness matrix of the state transformation $\boldsymbol{\mathcal{T}}(\boldsymbol{x})$.
C_s	Stability coefficient for loss-of-control prediction.
$\boldsymbol{G}(\boldsymbol{x})$	Control effectiveness matrix of the quadrotor state space system.
$\boldsymbol{H}(\boldsymbol{x})$	Output equation of the quadrotor state-space system.
$\boldsymbol{\mathcal{L}}_{\boldsymbol{f}}^{\boldsymbol{k}}(\boldsymbol{x})$	k-th order Lie-derivative with respect to vector field $\boldsymbol{f}$.
$\boldsymbol{P}$	Symmetric matrix in Lyapunov equation.
$\mathbb{R}^2$	Correlation coefficient..
$\boldsymbol{S}_R$	Set of feasible control realisations.
$\boldsymbol{\mathcal{T}}(\boldsymbol{x_a})$	Nonlinear state transformation.
$\boldsymbol{\mathcal{T}}(\boldsymbol{x})$	Nonlinear state transformation.
$V(\boldsymbol{\zeta})$	Lyapunov function as a function of the external states $\boldsymbol{\zeta}$.
$\boldsymbol{f}_0(\eta)$	Zero-dynamics of internal state η.
$\boldsymbol{f}_\eta(\boldsymbol{\xi};\eta)$	Dynamics of internal state η.
$\boldsymbol{f}(x)$	Vector of state equations of the quadrotor state space system.
$\boldsymbol{n}^B$	Constant primary axis vector in the body frame of reference.
$\boldsymbol{n^I}$	Primary axis vector in the inertial frame of reference.
s	Complex frequency in the Laplace domain.
t	Time.
$\boldsymbol{x_a}$	State vector of the augmented quadrotor dynamics with actuator dynamics.
$\boldsymbol{x}$	State vector of the quadrotor dynamics.
$\boldsymbol{\alpha}(\boldsymbol{x})$	Nonlinear state dynamics for state transformation $\boldsymbol{\mathcal{T}}(\boldsymbol{x})$.
$\boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t)$	Perturbation term due to neglected terms in INDI Taylor-series expansion.
η	Internal state of the state transformation $\boldsymbol{\mathcal{T}}(\boldsymbol{x})$.
$\boldsymbol{\nu}$	Virtual control vector for the original INDI controller.
$\boldsymbol{\nu}_r$	Virtual control realisation.
$\boldsymbol{\nu_a}$	Virtual control vector for the actuator compensated INDI controller.
$\boldsymbol{\nu_h}$	Hedged virtual control vector for the actuator compensated INDI controller.

ρ_i	Relative degree of output y_i.
$\overline{\rho}$	Sum of the relative degrees of all outputs.
$\boldsymbol{\xi}$	External state vector of the state transformation $\mathcal{T}(\boldsymbol{x})$.
$\boldsymbol{\zeta}$	External state vector of the extended state transformation $\mathcal{T}(\boldsymbol{x_a})$.

Control Parameters

$\boldsymbol{K_a}$	INDI control gain matrix.
K_p, K_i, K_d	Outer-loop PID control gains.
c_i	i-th constant in the internal state η.
Δt	Sampling period of INDI controller.
$\Delta(.)$	Increment of variable over one sampling period Δt.
k_i	INDI control gains.
μ_i	i-th parameter in the internal state η.

Descriptors

$(.)_{des}$	Subscript defining desired value.
$(.)_{ref}$	Subscript defining reference value.
$(.)_0$	Variable measured at $t = t_0$.

I. Introduction

In recent times, lightweight commercial Unmanned Aerial Vehicles (UAV) have been developing at a rapid pace. Due to their adaptability, simple operations and agility, small quadrotor drones have become popular commercial assets in the class below 5 kg, as highlighted by Mahony [1]. With this increased use of consumer UAVs, safety is becoming an ever more critical aspect in the development of drones. For quadrotors, this is strongly the case, because this type of UAV immediately becomes uncontrollable after rotor loss, as demonstrated by Lanzon[2] and Mueller [3][4].

In recent years, many advancements have been made in the development of fault tolerant control (FTC) systems that enable control over a quadrotor experiencing a single rotor failure (SRF). Lanzon [2] and Mueller [4] first developed controllers that allow damaged quadrotors to hover and regain limited position control, while the application of nonlinear control techniques by Sun [5][6] enables the continuation of high speed flight with single [5] or even double opposing rotor failure (DRF) [6]. The real world application of such algorithms is further improved with active fault detection and isolation [7] and nonlinear control techniques to recover from large attitude upsets [8].

For the state-of-the-art fault-tolerant controller by Sun [6], global mathematical stability has been proven using a stability analysis method developed by Wang et al [9]. Theoretically, the proof of stability for this controller holds as long as the pitch or roll angle does not exceed $\frac{\pi}{2}$ rad [6]. In this analysis, however, the effects of the actuator dynamics and rotor speed limits have not been taken into account, although it has been shown in high speed flight tests that actuator saturation can lead to loss of control scenarios [5][6].

In the past, the effect of actuator dynamics and actuator saturation on control performance have been studied extensively. For quadrotor UAVs, Raffler describes the effect of rotor saturation on the tracking performance of a nominal quadrotor and presents a trajectory generation method that compensates for these deficiencies [10]. Smeur describes the effect of actuator dynamics on the tracking performance of an incremental nonlinear dynamic inversion (INDI) controller for a nominal quadrotor [11] [12] and presents a control allocation method to reduce the effect of rotor saturation for nominal quadrotors [12]. Finally, Steffensen presents a method to compensate for the effects of actuator dynamics for Nonlinear Dynamic Inversion and Incremental Nonlinear Dynamic Inversion controllers [13].

For quadrotors in the SRF case, however, no extensive research has been conducted into the effects of actuator dynamics and rotor saturation on the stability around a trajectory. Considering that the quadrotor is vulnerable to upsets in the SRF case [8], there is value in gaining a better understanding in the effects of the actuators on the stability of fault-tolerant quadrotor controllers. Subsequently, this analysis could be used to predict loss of control before its occurrence. The main contribution of this research is the derivation of a loss-of-control prediction method using measured variables that can be used on-board in real-time.

For this purpose, a dynamic model is defined for a quadrotor experiencing single rotor loss in section II. Subsequently, a non-linear incremental dynamic inversion algorithm is presented for the control of the quadrotor upon single rotor failure in section III. Section IV disserts the effects of the actuator limitations on the controlled dynamics of the quadrotor and presents an augmented INDI controller that compensates for actuator lag. Subequently, the stability of the system under actuator related perturbation is analysed in section V, where a method is presented for the prediction of loss-of-control for quadrotors in the SRF case. In section VI the results of this method are presented for a simulated loss of control situation, followed by the conclusion and discussion in section VII and section VIII.

II. Quadrotor Model

This research has been conducted using the Parrot Bebop 2 quadrotor, a commercially available fixed pitch quadrotor platform. This quadrotor has been selected for this research due to the availability of extensive high-fidelity aerodynamic models for both the nominal [14] single rotor failure case [15]. First, the 6-degree-of-freedom rigid body equations of motion are described for the nominal Bebop 2 quadrotor in section II.A. When subjected to rotor failure, these equations of motion are unsuitable for quadrotor control, due to the loss of yaw control. Therefore, a reduced attitude representation is presented in section II.B that enables attitude control despite rotor failure.

A. Equations of Motion

A dynamic and kinematic model is defined in two different frames; the inertial frame, $\mathcal{F}^{\boldsymbol{I}}$, and the body frame $\mathcal{F}^{\boldsymbol{B}}$. The inertial frame is fixed at the origin, $O^{\boldsymbol{I}}$, with the x, y and $z-$ axes pointing North, East and Down (NED) respectively. The origin of the body-fixed frame, $\boldsymbol{O^B}$, is placed at the center of gravity of the quadrotor, with the x-axis pointing forward, the y-axis pointing right and the z-axis pointing downwards, as can be seen in fig. 1.

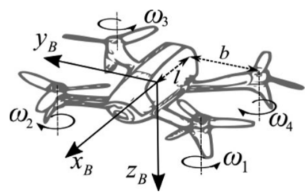

Fig. 1 The Parrot Bebop 2 quadrotor and the body frame origin and axis definitions [5].

Equation (1)-[4] depict the 6-degree-of-freedom equations of motion for the Bebop 2 quadrotor [1][5]. In these equations, $\boldsymbol{X^I}$ is the position in the inertial frame, V^I is the inertial frame velocity. The body frame angular velocity vector containing the pitch, roll and yaw rates is given by $\boldsymbol{\Omega}^B = [\ p \quad q \quad r\ ]^T$, whereas $\boldsymbol{\Omega}^B_{\times}$ signifies a skew-symmetric matrix for which the matrix vector product $\boldsymbol{\Omega}^B_{\times}\boldsymbol{v}$ is equal to the cross-product of $\boldsymbol{\Omega}^B \times \boldsymbol{v}$. The transformation matrix between the body frame to the inertial frame is given by $\boldsymbol{R}$, while $\boldsymbol{I}$ is the mass-moment of inertia matrix of the quadrotor. Finally, m signifies the mass of the quadrotor, whereas g is the gravitational acceleration vector [1][5].

$$\dot{\boldsymbol{X}}^{\boldsymbol{I}} = \boldsymbol{V}^I \tag{1}$$

$$\dot{\boldsymbol{V}}^I = \frac{1}{m}\boldsymbol{R}\boldsymbol{F}^B - g \tag{2}$$

$$\dot{\boldsymbol{R}} = \boldsymbol{R}\boldsymbol{\Omega}^B_{\times} \tag{3}$$

$$\dot{\Omega}^B = \boldsymbol{I}^{-1}\left(\boldsymbol{\Omega}^B_{\times}\boldsymbol{I}\boldsymbol{\Omega}^B + \boldsymbol{M}^B\right) \tag{4}$$

$\boldsymbol{F^B} = [\ F_x \quad F_y \quad F_z\ ]^T$ is the vector of forces acting on the quadrotor in the body frame, as seen in eq. (5)[5][6], in which ω_i is the rotational speed of the i-th rotor and $\bar{\kappa}$ is the thrust coefficient. $\boldsymbol{F_a^B}$ is the vector of external aerodynamic forces. For the Parrot Bebop 2 quadrotor with fixed-pitch, non-vectored propellors, the thrust coefficient is assumed constant and the thrust direction is parallel to the body-frame Z-axis at all times [5][6].The aerodynamic forces have been modelled in detail for the Bebop 2 quadrotor in prior research by Sun et al. for both the undamaged quadrotor [14] and the SRF case [15].

$$\boldsymbol{F^B} = \begin{bmatrix} 0 \\ 0 \\ -\bar{\kappa}\sum_{i=1}^{4}\omega_i^2 \end{bmatrix} + \boldsymbol{F_a^B} \tag{5}$$

The moments acting on the quadrotor, $\boldsymbol{M^B}$, are described in eq. (6) and consist of the vector of rotor speeds $\boldsymbol{u}$ multiplied by the control effectiveness matrix $\boldsymbol{G_M}$, the gyroscopic moments, $\boldsymbol{M_G^B}$, the yaw damping, $\boldsymbol{\gamma r}$, and the external aerodynamic moments, $\boldsymbol{M_a^B}$. In this equation, b and l are geometric constants related to the length and width of the quadrotor, as seen in fig. 1. σ is the rotor drag coefficient, I_p is the moment of inertia of an individual propeller around its axis of rotation and γ signifies the yaw-damping coefficient.

The highly nonlinear aerodynamic moments have been modelled for the Bebop 2 quadrotor for both the nominal quadrotor [14], as well as the SRF case [15].

$$\begin{aligned} \boldsymbol{M^B} &= \begin{bmatrix} \bar{\kappa}b & -\bar{\kappa}b & -\bar{\kappa}b & \bar{\kappa}b \\ \bar{\kappa}l & \bar{\kappa}l & -\bar{\kappa}l & -\bar{\kappa}l \\ \sigma & -\sigma & \sigma & -\sigma \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} + \begin{bmatrix} I_p q\,(\omega_1 - \omega_2 + \omega_3 - \omega_4) \\ -I_p p\,(\omega_1 - \omega_2 + \omega_3 - \omega_4) \\ I_p\,(\dot\omega_1 - \dot\omega_2 + \dot\omega_3 - \dot\omega_4) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\gamma r \end{bmatrix} + \boldsymbol{M_a^B} \\ &= \boldsymbol{G_M u} + \boldsymbol{M_G^B} + \boldsymbol{\gamma r} + \boldsymbol{M_a^B} \end{aligned} \tag{6}$$

Due to the symmetric placement of the rotors, the control effectiveness constants of the individual rotors in in eq. (6) can be simplified to:

$$G_p = \overline{\kappa}b \quad G_q = \overline{\kappa}k \quad G_r = \sigma$$

The control effectiveness matrix, $\boldsymbol{G_M}$ can then be simplified to:

$$\boldsymbol{G_M} = \begin{bmatrix} G_p & -G_p & -G_p & G_p \\ G_q & G_q & -G_q & -G_q \\ G_r & -G_r & G_r & -G_r \end{bmatrix} \tag{7}$$

B. Reduced Attitude Dynamics

Conventionally, quadrotor position and velocity control is performed through a combination of attitude and thrust control. By controlling the Euler attitude angles, the quadrotor is oriented in the appropriate direction, with control of thrust providing the desired magnitude of acceleration [1]. However, rotor loss leads to the loss of yaw control, causing the quadrotor to spin continuously [2] [3]. Therefore, conventional attitude control is infeasible, necessitating an alternative approach to maintain control over the quadrotor attitude. For this purpose, reduced attitude control is employed as first used for quadrotors in the SRF case by Mueller [3][4]. This approach was subsequently implemented in combination with nonlinear control methods by Lu [7] and Sun [5][6].

First, a constant unit vector is defined in the body frame called the primary axis, denoted by $\boldsymbol{n^B}$. The primary axis defines the axis of rotation of the quadrotor angular velocity and signifies the average direction of thrust over one rotational period, as stated by Mueller [4]. By converting this body frame vector to the inertial frame of reference, the average orientation of the thrust vector in inertial space, $\boldsymbol{n^I}$, is found[3][4]. By manipulating $\boldsymbol{n^I}$, the thrust vector can be aligned with a desired acceleration vector, allowing for control of quadrotor position despite the loss of yaw control. This method is graphically represented in fig. 2 [4].

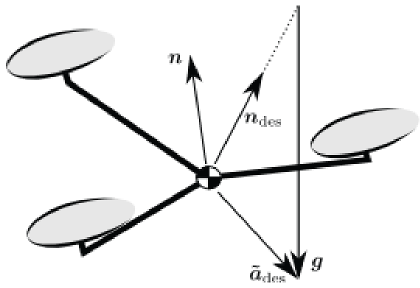

Fig. 2 A graphical representation of the position control by means of aligning the primary axis n with the desired primary axis n$_{des}$ by Mueller[4].

It is important that the primary axis and the orientation of the thrust in the body frame are similar. Aligning the primary axis with the thrust vector would lead to $\boldsymbol{n^B} = [\begin{array}{ccc} 0 & 0 & -1 \end{array}]^T$. However, in some cases, it can be advantageous to have small offset with the thrust vector by introducing small non-zero horizontal components, for example $\boldsymbol{n^B} = [\begin{array}{ccc} 0.2 & 0.2 & -0.96 \end{array}]^T$. As explained in detail in section IV.D, a small offset with respect to the thrust vector induces a stabilising precession moment [4] [8].

Substituting the reduced attitude representation into the quadrotor equations of motion in eq. (1) and eq. (2) leads to the following translation dynamics: [6].

$$\begin{aligned} \dot{\boldsymbol{X}}^{\boldsymbol{I}} &= \boldsymbol{V^I} \\ \dot{\boldsymbol{V}}^{\boldsymbol{I}} &= \frac{1}{m}\left(\boldsymbol{n^I} T + \boldsymbol{R}\boldsymbol{F}_a^B\right) + \boldsymbol{g} \end{aligned} \tag{8}$$

The acceleration vector in the inertial frame is seen to be proportional to the magnitude of thrust multiplied by the direction of the inertial primary axis vector, $\boldsymbol{n^I}$. This shows that inertial position and velocity control is possible through control of the reduced attitude and thrust of the quadrotor[2][3][4].

For the purpose of position control, a reference value for the primary axis, $\boldsymbol{n^I_{des}}$ and thrust, T_{des} can be derived using the position dynamics in eq. (8). Ignoring the effect of external aerodynamic forces, the required thrust and inertial frame primary axis can be computed for a given inertial acceleration reference vector, $\boldsymbol{a^I_{des}}$ using the following equation:

$$\begin{aligned} T_{des} &= ||\boldsymbol{a^I_{des}} - \boldsymbol{g}|| \\ \boldsymbol{n^I_{des}} &= \frac{\boldsymbol{a^I_{des}} - \boldsymbol{g}}{||\boldsymbol{a^I_{des}} - \boldsymbol{g}||} \end{aligned} \tag{9}$$

This provides the primary axis reference in the inertial frame, which provides the desired position dynamics. This primary axis reference vector $\boldsymbol{n^I_{des}}$, however, is defined in the inertial frame of reference, whereas attitude and thrust control is performed in the body frame. Therefore, the inertial primary axis reference $\boldsymbol{n^I_{des}}$ is converted to a reference value for the primary axis in the body frame, $\boldsymbol{h} = \boldsymbol{n}^B_{des}$ using eq. (10). [5]

$$\boldsymbol{h} = \boldsymbol{n}^B_{des} = \boldsymbol{R}^T \boldsymbol{n^I_{des}} \tag{10}$$

The attitude of the quadrotor can then be controlled by letting $\boldsymbol{h}$ track the constant reference value $\boldsymbol{n^B}$, such that the desired body-frame primary axis vector aligns with the fixed primary axis [7][5]. This ultimately aligns the direction of the quadrotor acceleration with its required orientation. This principle can be observed graphically in fig. 2 [4][5].

The kinematics of this manoeuvre are found through computation of the time derivative of $\boldsymbol{h}$ using eq. (3) and eq. (10). The kinematics are shown in eq. (11), in which $\lambda(t)$ signifies the first derivative of $\boldsymbol{n^I_{des}}$ transformed to the body frame [6].

$$
\begin{aligned}
\dot{\boldsymbol{h}} &= \frac{d}{dt}\left(\boldsymbol{R^T}\boldsymbol{n}^{\boldsymbol{I}}_{des}\right)\\
&= \dot{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{n}^{\boldsymbol{I}}_{des} + \boldsymbol{R^T}\dot{\boldsymbol{n}}^{\boldsymbol{I}}_{des}\\
&= -\boldsymbol{\Omega}^B_{\times}\boldsymbol{R}^T\boldsymbol{n}^I_{des} + \boldsymbol{R}^T\dot{\boldsymbol{n}}^I_{des}\\
&= -\boldsymbol{\Omega}^B_{\times}\boldsymbol{h} + \boldsymbol{\lambda}(t)
\end{aligned}
\tag{11}
$$

$$
\begin{bmatrix} \dot{h}_1 \\ \dot{h}_2 \\ \dot{h}_3 \end{bmatrix} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix}\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} + \boldsymbol{\lambda}(t)
$$

Equation (11) shows that the kinematics of $\boldsymbol{h}$ are dependent on the pitch, roll and yaw rates, p, q and r. By controlling the angular rates, the primary axis reference $\boldsymbol{h}$ can therefore be stabilised around $\boldsymbol{n}^B$, aligning the reference vector with the primary axis. Combining the reduced attitude control with control of the thrust, the acceleration of the quadrotor in inertial space can be controlled.

However, as explained by Lu [7] and Sun [5][6], these kinematics cannot directly be used for attitude control. First of all, with three attitude variables and the thrust combined, there are four outputs for only three remaining inputs, leaving the problem underactuated. Secondly, the reduced attitude dynamics in eq. (11) cannot be inverted as long as the order of the skew-symmetric matrix on the right-hand side is uneven, which is ultimately problematic in the design of an INDI controller for the inner-loop.

To counteract this problem, the reduced attitude kinematics are rewritten. Considering that $\boldsymbol{h}$ is a unit-vector, accurate tracking of h_1 and h_2 also implies accurate tracking of h_3, since $\sqrt{h_1^2 + h_2^2 + h_3^2} = 1$. The kinematics of h_1 and h_2 consequently are sufficient to describe the reduced attitude of the quadrotor [5]. Therefore, taking h_3 as a measured external variable, the kinematics can simplified to [5]:

$$
\begin{bmatrix} \dot{h}_1 \\ \dot{h}_2 \end{bmatrix} = \begin{bmatrix} 0 & -h_3 \\ h_3 & 0 \end{bmatrix}\begin{bmatrix} p \\ q \end{bmatrix} + \begin{bmatrix} h_2 \\ -h_1 \end{bmatrix} r + \begin{bmatrix} \lambda_1(t) \\ \lambda_2(t) \end{bmatrix}
\tag{12}
$$

With the elimination of the third order skew-symmetric matrix, the attitude dynamics become invertible [5][6][7]. Taking the uncontrollable yaw rate, r, and h_3 as external measured variables, the reduced attitude can be controlled using the pitch and roll rates. With this simplification, the reduced attitude kinematics can now be used in the design of the attitude control [7].

III. Controller Design

Using the reduced attitude kinematics described in the previous section, a control algorithm for the damaged quadrotor can be designed. The fault-tolerant controller design presented here is based on an earlier fault-tolerant controller by Sun et al. [6] and is presented schematically in fig. 3

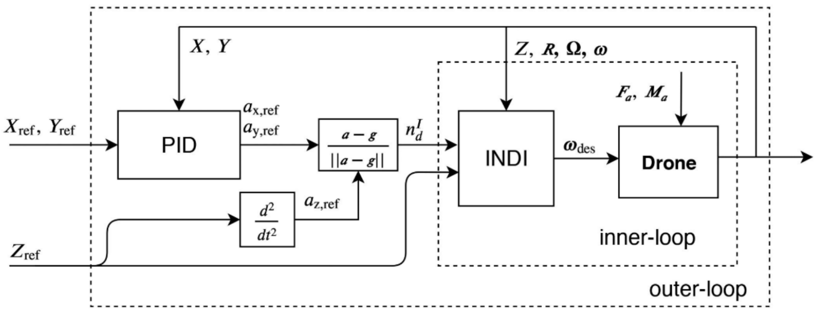

Fig. 3 Schematic representation of the cascaded fault-tolerant control loop [6]

An outer loop proportional, integral and differential (PID) controller controls the position, velocity and acceleration of the quadrotor and provides a reference signal to an inner loop controller that controls the reduced attitude, in addition to the altitude. This inner loop controller is designed using an INDI approach. First a state-space system is presented for the quadrotor attitude and altitude dynamics in section III.A. In section III.B, the outer-loop position control is presented. Subsequently, the inner-loop INDI controller design methodology is presented in section III.C, followed by the design of the controller in section III.D. Finally, the stability of the internal dynamics are analysed in section III.E.

A. State-Space System

For the design of the controller, the dynamics of the quadrotor are rewritten in the form of a time-invariant, nonlinear, control affine state-space system:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(x)\boldsymbol{u} \\ \boldsymbol{y} &= \boldsymbol{H}(\boldsymbol{x}) \end{aligned} \tag{13}$$

In this system with n states, m inputs $\boldsymbol{u}$ and l outputs, the first equation describes the state dynamics. In this equation, the state equation $\boldsymbol{f}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^n$ is a smooth vector field, and the control effectiveness matrix $\boldsymbol{G}(x) : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ is a smooth function. The second equation describes the output dynamics, in which the smooth vector field $\boldsymbol{H}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^l$ is the output equation [9].

A useful nonlinear state-space system for quadrotor dynamics upon rotor failure was first presented by Sun et al.[6]. Originally, this model was derived for dual opposing rotor failure, but has been adapted to suit single rotor failure for this research. The state-space system describes the altitude dynamics of the damaged quadrotor, in addition to the reduced attitude dynamics as presented in section II.B. This state-space model is subsequently used to design the outer-loop PID and inner-loop INDI control algorithms.

First, the outputs to this system are chosen. As first proposed by Sun [6], the outputs are chosen to be the altitude tracking error, $Z - Z_{ref}$, together with the tracking errors of the reduced attitude components, $h_1 - n_x^B$ and $h_2 - n_y^B$. Altitude control is prioritised over horizontal position control, considering that maintaining sufficient clearance with the ground is most critical. If altitude and attitude control are provided accurately, horizontal position control automatically follows and can be controlled by adjusting the tilt of the thrust vector. The output equation therefore becomes:

$$\boldsymbol{y} = \boldsymbol{H}(\boldsymbol{x}) = \begin{bmatrix} Z - Z_{ref}, & h_1 - n_x^B, & h_2 - n_y^B \end{bmatrix}^T \tag{14}$$

Based on these desired outputs, Sun et al. [6] select the following states:

$$\boldsymbol{x} = \begin{bmatrix} Z & V_z & h_1 & h_2 & p & q & r \end{bmatrix}^T \tag{15}$$

The states include the altitude and vertical velocity, Z and V_z, two components of the primary axis reference h_1 and h_2, together with the angular velocities p, q and r. Given this selection of states, the state-space system for the altitude and attitude dynamics of the quadrotor can be defined [6]. Using the 6-DoF equations of motion from eq. (1)-(4) and the reduced attitude kinematics of eq. (12), the state dynamics can be written in the form of eq. (13) as seen below [6]:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} = \begin{bmatrix} \dot{Z} \\ \dot{V}_z \\ \dot{h}_1 \\ \dot{h}_2 \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} V_z \\ g + F_{a,z}^I \\ h_2 r - h_3 q + \lambda_1 \\ h_3 p - h_1 r + \lambda_2 \\ A_x qr + \frac{1}{I_x}\left(M_{G_x}^B + M_{a_x}^B\right) \\ A_y pr + \frac{1}{I_y}\left(M_{G_y}^B + M_{a_y}^B\right) \\ A_z pq + \frac{1}{I_z}\left(M_{G_z}^B + M_{a_z}^B - \gamma r\right) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ -\frac{n_z^I \overline{\kappa}}{m} & -\frac{n_z^I \overline{\kappa}}{m} & -\frac{n_z^I \overline{\kappa}}{m} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ G_p & -G_p & -G_p \\ G_q & G_q & -G_q \\ G_r & -G_r & -G_r \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \end{bmatrix} \tag{16}$$

In the state space system of eq. (16), the following constants related to the quadrotor moments of inertia are introduced [6]:

$$A_x = \frac{I_y - I_z}{I_x} \quad A_y = \frac{I_z - I_x}{I_y} \quad A_z = \frac{I_x - I_y}{I_z}$$

The state-space system defined in eq. (16) can subsequently be used to design a controller for the damaged quadrotor.

B. Outer Loop Control

First, an outer loop controller is designed for position control. The input to this controller is a twice-differentiable trajectory, which can ideally be defined using trajectory planning algorithms, such as defined by Hehn[16] or Mueller [17][18].

The outer-loop control is provided by a PID controller that controls horizontal position only. Vertical position control is executed by the inner-loop INDI controller, due to the prioritisation of altitude control mentioned in the previous section. The acceleration reference signal provided by the outer-loop controller therefore becomes [6]:

$$\boldsymbol{a}^I_{des} = \begin{bmatrix} K_p(x^I - x^I_{ref}) + K_i \int (x^I - x^I_{ref})dt + K_d(V^I_x - V^I_{x_{ref}}) \\ K_p(y^I - y^I_{ref}) + K_i \int (y^I - y^I_{ref})dt + K_d(V^I_y - V^I_{y_{ref}}) \\ a_{z_{ref}} \end{bmatrix} \tag{17}$$

In eq. (17), K_p, K_i, and K_d are the proportional, integral and differential control gains, respectively. In this case, equal weight is given to position control in the X^I and Y^I directions, so both controllers have the same PID gains. The acceleration reference signal in the inertial frame is then converted to an inertial frame primary axis reference signal, $\boldsymbol{n}^I_{des}$ using eq. (9). Subsequently, this vector is converted to a primary axis reference in the body frame of reference, $\boldsymbol{h}$, using eq. (10).

C. INDI Design Method

An inner-loop controller is designed to control the altitude and reduced attitude of the quadrotor, based on the state-space system from section III.A. Since the quadrotor is subjected to large uncertainties as a rezult of highly nonlinear aerodynamic forces and moments, $\boldsymbol{F}_a$ and $\boldsymbol{M}_a$, the controller is designed using an INDI method to provide robustness against these uncertainties [5][6]. This type of controller is a sampling-based variant of the Nonlinear Dynamic Inversion (NDI) methodology. Due to the use of high-frequency sampling of the output and its derivatives, the need for a high-fidelity model of state dynamics and external disturbances is eliminated, providing additional robustness against uncertainties [9].

The design of this INDI control law requires a system in a control canonical form, consisting of a chain of output derivatives that provide a direct relationship between the outputs, $\boldsymbol{y}$, and the control inputs, $\boldsymbol{u}$. For this purpose, a state transformation with a transformed state vector $z = \boldsymbol{\mathcal{T}}(\boldsymbol{x})$ is defined:

$$z = \boldsymbol{\mathcal{T}}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{bmatrix} \tag{18}$$

The state transformation in eq. (18), maps the system from eq. (16) into a control canonical form that provides the required relation between outputs and inputs [19]. The chain of output derivatives is represented by the external states, $\boldsymbol{\xi}$, which signify the controllable part that is to be used in control design [19]. In some cases, an uncontrollable part remains that is independent of the inputs, represented by the internal states, $\boldsymbol{\eta}$. The stability of the internal states cannot be influenced by the controller and must therefore be verified up front [9][19].

First, the controllable external states $\boldsymbol{\xi}$ are derived and placed into a canonical form. For this purpose the time-derivatives of the outputs $\boldsymbol{y}$ are computed until a direct relationship with the input $\boldsymbol{u}$ appears [19]. This is done by computing the Lie-derivatives of the output function $\boldsymbol{H}(\boldsymbol{x})$ with respect to the state function $\boldsymbol{f}(\boldsymbol{x})$ and control effectiveness matrix $\boldsymbol{G}(\boldsymbol{x})$ [9][19]. The first order time derivative of the i-th element of $\boldsymbol{H}(\boldsymbol{x})$ is given by [19]:

$$\begin{aligned} \dot{y}_i &= \frac{\partial H_i(\boldsymbol{x})}{\partial \boldsymbol{x}} \left[\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} \right] \\ &= \mathcal{L}_f H_i(\boldsymbol{x}) + \mathcal{L}_G H_i(\boldsymbol{x})\boldsymbol{u} \end{aligned} \tag{19}$$

In eq. (19), $\mathcal{L}_f H_i(\boldsymbol{x}) = \frac{\partial H_i(\boldsymbol{x})}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x})$ is the first order Lie-derivative of $H_i(\boldsymbol{x})$ with respect to $\boldsymbol{f}(\boldsymbol{x})$. Similarly, $\mathcal{L}_G H_i(\boldsymbol{x})$ is the first order Lie-derivative of $H_i(\boldsymbol{x})$ with respect to $\boldsymbol{G}(\boldsymbol{x})$. If $\dot{y}_i$ is independent of the input $\boldsymbol{u}$, the

Lie-derivative $\mathcal{L}_G H_i(\boldsymbol{x}) = 0$. In that case, the process is to be repeated with the second order Lie-derivative. This process is continued for the k-th order output derivative, until a direct relation with the input is found[19]. The k-th order time derivative of output y_i, is denoted by $y_i^{(k)}$ and can be written as:

$$\begin{aligned} y_i^{(k)} &= \frac{\partial(\mathcal{L}_f^{k-1} H_i(\boldsymbol{x}))}{\partial \boldsymbol{x}} \left[\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}\right] \\ &= \mathcal{L}_f \mathcal{L}_f^{k-1} H_i(\boldsymbol{x}) + \mathcal{L}_f^{k-1} \mathcal{L}_G H_i(\boldsymbol{x})\boldsymbol{u} \\ &= \mathcal{L}_f^{k} H_i(\boldsymbol{x}) + \mathcal{L}_f^{k-1} \mathcal{L}_G H_i(\boldsymbol{x})\boldsymbol{u} \end{aligned} \tag{20}$$

If the output derivative $y_i^{(k)}$ is independent of the input $\boldsymbol{u}$, the k-th order Lie-derivative with respect to $\boldsymbol{G}(\boldsymbol{x})$ is 0 [19]. In that case, $y_i^{(k)}$ can be rewritten as:

$$y_i^{(k)} = \mathcal{L}_f^{k} H_i(\boldsymbol{x}) \tag{21}$$

This process is repeated until $\boldsymbol{u}$ appears in the ρ_i-th output derivative $y_i^{(\rho_i)}$, when $\mathcal{L}_f^{\rho_i - 1} \mathcal{L}_G H_i(\boldsymbol{x})\boldsymbol{u} \neq 0$ [9][19]. The constant ρ_i is called the relative degree and signifies the lowest derivative order of output y_i in which a direct relation with the inputs $\boldsymbol{u}$ exists [19].

For each of the individual outputs y_i in $\boldsymbol{H}(\boldsymbol{x})$, the system can then be rewritten into a control canonical form as a ρ_i-th order chain output time derivatives [9]. These output time derivatives constitute the external states belonging to output y_i. The external state vector $\boldsymbol{\xi}_i$ belonging to output y_i is therefore defined as:

$$\begin{aligned} \boldsymbol{\xi}_i &= \begin{bmatrix} H_i(\boldsymbol{x}) & \mathcal{L}_f H_i(\boldsymbol{x}) & \mathcal{L}_f^2 H_i(\boldsymbol{x}) & ..., & \mathcal{L}_f^2 H_i(\boldsymbol{x}) & ,..., & \mathcal{L}_f^{\rho_i - 1} H_i(\boldsymbol{x}) \end{bmatrix}^T \\ &= \begin{bmatrix} y_i & y_i^{(1)} & y_i^{(2)} & ..., & y_i^{(k)}) & ,..., & y_i^{(\rho_i - 1)} \end{bmatrix}^T \end{aligned} \tag{22}$$

Given this state vector, the external state dynamics can defined. For each of the outputs y_i, the external dynamics $\dot{\boldsymbol{\xi}}_i$ can be written as [9]:

$$\begin{aligned} \dot{\boldsymbol{\xi}}_i = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(k)} \\ \vdots \\ y^{(\rho_i)} \end{bmatrix} = \begin{bmatrix} \mathcal{L}_f H_i(\boldsymbol{x}) \\ \mathcal{L}_f^2 H_i(\boldsymbol{x}) \\ \vdots \\ \mathcal{L}_f^k H_i(\boldsymbol{x}) \\ \vdots \\ \mathcal{L}_f^{\rho_i} H_i(\boldsymbol{x}) + \mathcal{L}_f^{\rho_i - 1} \mathcal{L}_G H_i(\boldsymbol{x})\boldsymbol{u} \end{bmatrix} = \begin{bmatrix} \xi_{i_2} \\ \xi_{i_3} \\ \vdots \\ \xi_{i_{k-1}} \\ \vdots \\ \boldsymbol{\alpha_i}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}_i}(\boldsymbol{x})\boldsymbol{u} \end{bmatrix} \\ = \boldsymbol{A_{c_i}} \boldsymbol{\xi}_i + \boldsymbol{B_{c_i}} \left(\boldsymbol{\alpha_i}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}_i}(\boldsymbol{x})\boldsymbol{u}\right) \end{aligned} \tag{23}$$

In eq. (23), $\boldsymbol{A_{c_i}}$ and $\boldsymbol{B_{c_i}}$ signify the state and input matrices of the control canonical form belonging to a chain of integrators. $\boldsymbol{\alpha_i}(\boldsymbol{x})$ is equal to Lie-derivative $\mathcal{L}_f^{\rho} H_i(\boldsymbol{x})$ and signifies the state dependent terms in the ρ_i-th output derivative $y_i^{(\rho_i)}$. $\boldsymbol{\mathcal{B}_i}(\boldsymbol{x})$ denotes the Lie-derivative $\mathcal{L}_f^{\rho_i - 1} \mathcal{L}_G H_i(\boldsymbol{x})$ and signifies the control effectiveness matrix for $y_i^{(\rho)}$ [9].

The process in this section is repeated for each of the individual outputs, y_i, in the output equation $\boldsymbol{H}(\boldsymbol{x})$. Completing the process for each of the three outputs ultimately leads to the the dynamics seen below [9]:

$$\begin{aligned} \boldsymbol{y}^{(\rho)} &= \begin{bmatrix} y_1^{(\rho_1)} \\ y_2^{(\rho_2)} \\ y_3^{(\rho_3)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha_1}(\boldsymbol{x}) \\ \boldsymbol{\alpha_2}(\boldsymbol{x}) \\ \boldsymbol{\alpha_3}(\boldsymbol{x}) \end{bmatrix} + \begin{bmatrix} \boldsymbol{\mathcal{B}_1}(\boldsymbol{x}) \\ \boldsymbol{\mathcal{B}_2}(\boldsymbol{x}) \\ \boldsymbol{\mathcal{B}_3}(\boldsymbol{x}) \end{bmatrix} \boldsymbol{u} \\ &= \boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u} \end{aligned} \tag{24}$$

Given these dynamics of the output derivatives $\boldsymbol{y}^{(\rho)}$, the external dynamics of the full system can be rewritten as: [9]

$$\dot{\boldsymbol{\xi}} = \boldsymbol{A}_c \boldsymbol{\xi} + \boldsymbol{B}_c \left[\alpha(x) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u}\right] \tag{25}$$

With this derivation, the output dynamics of the system have been transformed into a canonical series of integrators that provide a direct relationship between the outputs and the inputs. This transformed state representation can be used to design a controller that provides the desired output dynamics [9].

This INDI methodology that is used is a variation of nonlinear dynamic inversion (NDI). NDI inverts the nonlinear system dynamics and provides a linear output response, even in case of highly nonlinear system dynamics [19][9]. A conventional NDI control law is designed by inverting the dynamics of eq. (24) [9][19]:

$$\boldsymbol{u}_{ndi} = \boldsymbol{\mathcal{B}}(\boldsymbol{x})^{-1}\left(\boldsymbol{v} - \boldsymbol{\alpha}(\boldsymbol{x})\right) \tag{26}$$

In this equation, $\boldsymbol{v}$ is the virtual control signal, which signifies the desired dynamics of $\boldsymbol{y}^{(\rho)}$. The dynamics of $\boldsymbol{y}^{(\rho)}$ using this control law become: [9]

$$\begin{aligned} \boldsymbol{y}^{(\rho)} &= \boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{\mathcal{B}}(\boldsymbol{x})^{-1}\left(\boldsymbol{v} - \boldsymbol{\alpha}(\boldsymbol{x})\right) \\ &= \boldsymbol{v} \end{aligned} \tag{27}$$

Under the control law of eq. (26), the dynamics of $\boldsymbol{y}^{(\rho)}$ follow the virtual control $\boldsymbol{v}$. The external state dynamics then become: [9]

$$\dot{\boldsymbol{\xi}} = \boldsymbol{A}_c\boldsymbol{\xi} + \boldsymbol{B}_c\boldsymbol{v} \tag{28}$$

Considering that the outputs $\boldsymbol{y}$ are tracking errors, the virtual control can be designed as a regulator around $\boldsymbol{y} = 0$ [9]. Since $\boldsymbol{\xi}$ signifies an integrator chain of output derivatives, this regulator can be written as:

$$\boldsymbol{v} = -\boldsymbol{K}\boldsymbol{\xi} \tag{29}$$

In eq. (29), $\boldsymbol{K}$ signifies a feedback gain matrix. Applying this virtual control to the system dynamics in eq. (28), the external dynamics of the system become [9]:

$$\dot{\boldsymbol{\xi}} = (\boldsymbol{A}_c - \boldsymbol{B_c}\boldsymbol{K})\boldsymbol{\xi} \tag{30}$$

The controlled system dynamics in eq. (30) are globally stable around the origin at $\boldsymbol{\xi} = 0$ if $(\boldsymbol{A_c} - \boldsymbol{BcK})$ is Hurwitz [9][19]. However, NDI control requires perfect knowledge of the system dynamics $\boldsymbol{\alpha}(\boldsymbol{x})$ to perform the inversion in eq. (26). In reality, these dynamics contain highly nonlinear aerodynamic effects, in addition to model uncertainties. Including these effects in the control law is prohibitively complex and could lead to large modeling errors. Therefore, incremental nonlinear dynamic inversion is applied [6]. It follows the same principles, but does not require accurate modelling of state dependent effects $\boldsymbol{\alpha}(\boldsymbol{x})$. Instead, the output derivatives $\boldsymbol{y}^{(\rho)}$ are measured at high frequency, negating the need for high accuracy dynamic models of $\boldsymbol{\alpha}(\boldsymbol{x})$ [6][9]. Wang et al [9] have mathematically proven that this type of controller is robust to both external disturbances and modelling uncertainties, given a sufficient sample rate, bounded disturbances and stable internal dynamics.

The first step of INDI design is to perform a Taylor series expansion the output derivatives $\boldsymbol{y}^{(\rho)}$ at sampling time $t = t_0$[6] [9]:

$$\begin{aligned} \boldsymbol{y}^{(\rho)} &\approx \boldsymbol{y}_0^{\rho} + \frac{\partial \boldsymbol{y}^{(\rho)}}{\partial u}\boldsymbol{\Delta u} + \frac{\partial \boldsymbol{y}^{(\rho)}}{\partial x}\boldsymbol{\Delta x} + \boldsymbol{\Delta d} + O(\boldsymbol{\Delta x^2}) \\ &\approx \boldsymbol{y}_0^{\rho} + \mathcal{B}(x)\boldsymbol{\Delta u} + \frac{\partial[\boldsymbol{\alpha}(x) + \mathcal{B}(x)\boldsymbol{u}]}{\partial x}\boldsymbol{\Delta x} + \boldsymbol{\Delta d} + O(\boldsymbol{\Delta x^2}) \\ &\approx \boldsymbol{y}_0^{\rho} + \mathcal{B}(x)\boldsymbol{\Delta u} + \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t) \end{aligned} \tag{31}$$

In this equation, $\boldsymbol{y}_0^{(\rho)}$ signifies the measured output derivatives $\boldsymbol{y}^{(\rho)}$ at $t = t_0$. The increments of the states and the inputs over the sampling period Δt are represented by $\boldsymbol{\Delta x}$ and $\boldsymbol{\Delta u}$. Assuming that there are external disturbances, $\boldsymbol{\Delta d}$ represents the increments of these disturbances over the sampling period Δt. Finally, $\boldsymbol{O(\Delta x^2)}$ represents the residual of neglected higher order terms in the Taylor series expansion. To simplify these dynamics, the terms dependent on the increments of states $\boldsymbol{\Delta x}$, disturbance increments $\boldsymbol{\Delta d}$ and the higher order terms $\boldsymbol{O(\Delta x^2)}$ are grouped in the perturbation term $\boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t)$ [9].

It is assumed that the state increment $\boldsymbol{\Delta x}$ and disturbance increment $\boldsymbol{\Delta d}$ are significantly slower than the input increments $\boldsymbol{\Delta u}$ [6][9]. Therefore, for a sufficiently small sampling interval Δt, it can be assumed that the state dynamics and external disturbances are sufficiently captured by the measurement of $\boldsymbol{y}_0^{(\rho)}$ [6][9]. Assuming that $\boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t)$ is bounded, the limit of this perturbation as $\Delta t \to 0$ can be taken to be [9]:

$$\lim_{\Delta t \to 0} \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t) = 0$$

In earlier iterations of fault-tolerant quadrotor control, a sampling frequency of 512 Hz was found to be adequate for making this assumption [5][6][11]. Therefore, the effect of the perturbation $\boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t)$ is ignored for the design of the controller under the assumption that the measurement of $\boldsymbol{y}_0^{(\rho)}$ captures the state and disturbance dynamics[9]. Under this assumption, the control increment $\boldsymbol{\Delta u}$ can be determined similarly to the NDI control law using the output dynamics in eq. (31) [9]:

$$\boldsymbol{\Delta u} = \mathcal{B}\left(\boldsymbol{x}_f\right)^{-1}\left(\nu - \boldsymbol{y}_f^{(\rho)}\right) \tag{32}$$

In this case, $\boldsymbol{x}_f$ and $\boldsymbol{y}_f^{(\rho)}$ signify the filtered measurements of $\boldsymbol{x}$ and $\boldsymbol{y}^{(\rho)}$. Due to the multiple time-derivatives, the measurements of $\boldsymbol{y}^{(\rho)}$ can be very noisy. To synchronise all the measurements to the same time-step t_0, all signals are filtered with a low-pass filter of the same frequency [9][11]. Similarly, $\boldsymbol{\Delta u} = \boldsymbol{u_{des}} - \boldsymbol{u_f}$. It should be noted that whilst its possible to ignore the perturbation term, $\boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t)$, in the controller synthesis, Wang [9] notes that it is non-trivial for stability analysis and can have a profound effect on the stability of the controller.

Using the virtual control law from eq. (29), the dynamics of $\boldsymbol{y}^{(\rho)}$ can be derived[9]:

$$\begin{aligned}
\boldsymbol{y}^{(\rho)} &\approx \boldsymbol{y}_m^{\rho} + \mathcal{B}(x)\mathcal{B}^{-1}\left(\boldsymbol{x}_0\right)\left(\nu - \boldsymbol{y}_m^{(\rho)}\right) + \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t) \\
&\approx \boldsymbol{\nu} + \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t) \\
&\approx -\boldsymbol{K\xi} + \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t)
\end{aligned} \tag{33}$$

Applying the dynamics in eq. (33) to the external dynamics in eq. (34) leads to the external state dynamics under INDI control, as seen below [9]:

$$\begin{aligned}
\dot{\boldsymbol{\xi}} &= \left(\boldsymbol{A_c} - \boldsymbol{B_c K}\right)\boldsymbol{\xi} + \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t) \\
&\approx \left(\boldsymbol{A_c} - \boldsymbol{B_c K}\right)\boldsymbol{\xi}
\end{aligned} \tag{34}$$

If $(\boldsymbol{A_c} - \boldsymbol{B_c K})$ is Hurwitz, this control law globally stabilises the external states $\boldsymbol{\xi}$ around the origin at $\boldsymbol{\xi} = \boldsymbol{0}$ [9]. The controlled external states are now stabilized, which means that the controlled system tracks the reference signals. However, this state transformation $\boldsymbol{\mathcal{T}(x)}$ is not completed. When the total relative degree of the outputs, $\sum_{i=0}^{i=m} \rho_i = \overline{\rho}$, is smaller than the number of states, n, the state transformation $\boldsymbol{\mathcal{T}(x)}$ contains uncontrollable internal states[9][19]. The required number of internal states to complete the transformation $\mathcal{T}(x)$ is equal to $n - \overline{\rho}$, the number of states minus the total relative degree, as seen in eq. (35) [9][19].

$$\boldsymbol{\eta} = \begin{bmatrix} \eta_1 & \eta_2 & \dots, & \eta_{n-\bar{\rho}} \end{bmatrix}^T \tag{35}$$

For the states $\boldsymbol{x}$ and the corresponding external states $\boldsymbol{\xi}$ the internal state selection is not unique. However, for $\mathcal{T}(x)$ to be a valid state transformation, the internal states are required to be independent of the input $\boldsymbol{u}$ as seen in eq. (36) [9] [19].

$$\frac{\partial \eta_i}{\partial x}\boldsymbol{G}(x)u = 0 \tag{36}$$

For the system to be stable, the internal dynamics $\dot{\boldsymbol{\eta}} = \boldsymbol{f_\eta(\xi, \eta)}$ must be stable. Because the internal dynamics are not directly controllable, instability of the internal dynamics cannot be compensated and will ultimately lead to full system instability. The stability of the internal dynamics must therefore be verified up front. The internal dynamics are determined as follows [9]:

$$\begin{aligned}
\dot{\boldsymbol{\eta}} = f_\eta(\boldsymbol{\xi}, \boldsymbol{\eta}) &= \frac{\partial \boldsymbol{\eta}}{\partial \boldsymbol{x}}\left(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}\right) \\
&= \frac{\partial \boldsymbol{\eta}}{\partial \boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x})
\end{aligned} \tag{37}$$

With the addition of an internal state, the full system dynamics of $\mathcal{T}(\boldsymbol{x})$ can be defined, as seen below [9]:

$$
\begin{aligned}
\dot{\boldsymbol{\eta}} &= \boldsymbol{f}_{\eta}(\boldsymbol{\eta}, \boldsymbol{\xi}) \\
\dot{\boldsymbol{\xi}} &= (\boldsymbol{A}_c - \boldsymbol{B_c K})\boldsymbol{\xi} + \boldsymbol{B}_c \boldsymbol{\delta}(z, \Delta t) \\
\boldsymbol{y} &= \boldsymbol{C}_c \boldsymbol{\xi}
\end{aligned} \tag{38}
$$

D. Inner-Loop Control

The INDI design methodology from section III.C is now applied to the quadrotor state-space system in eq. (16) given the output vector $\boldsymbol{H}(\boldsymbol{x})$ in eq. (14). First, the external states are determined, which transforms the outputs into the canonical form through the use of the Lie-derivatives. Subsequently, an INDI control law is designed that controls the altitude and reduced attitude of the quadrotor, based on this canonical form system [6].

For each of the outputs, the output time derivatives are determined using the Lie-derivatives with respect to $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{G}(\boldsymbol{x})$, until a relation to the input $\boldsymbol{u}$ appears. For the first output, $Z - Z_{ref}$ this starts with:

$$
\begin{aligned}
y_1^{(1)} = \dot{\xi}_{1_1} &= \mathcal{L}_f H_1(\boldsymbol{x}) + \mathcal{L}_g H_1(\boldsymbol{x}) \\
&= V_z \\
&= \xi_{1_2}
\end{aligned} \tag{39}
$$

It can be seen that no direct relation with inputs $\boldsymbol{u}$ appears, so an additional derivative is required. This means that an additional external state ξ_{1_2} is added to the system, which is equal to the derivative of the first external state, $\dot{\xi}_{1_1}$. The second order time derivative of the output is given by:

$$
\begin{aligned}
y_1^{(2)} \quad &= \dot{\xi}_{1_2} = \mathcal{L}_f^2 H_1(\boldsymbol{x}) + \mathcal{L}_f \mathcal{L}_g H_1(\boldsymbol{x}) \\
&= g + F_{a_z} + \begin{bmatrix} -\frac{n_z^I \overline{\kappa}}{m_v} & -\frac{n_z^I \overline{\kappa}}{m_v} & -\frac{n_z^I \overline{\kappa}}{m_v} \end{bmatrix} \boldsymbol{u} \\
&= \boldsymbol{\alpha_1}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}_1}(\boldsymbol{x})\boldsymbol{u}
\end{aligned} \tag{40}
$$

A direct link appears with the inputs $\boldsymbol{u}$ after two derivations of y_1. This means that the relative degree of the first output, $\rho_1 = 2$. The external state vector related to the altitude then becomes $\boldsymbol{\xi_Z} = \begin{bmatrix} Z & V_z \end{bmatrix}^T$. The same steps are now repeated for the second output, $y_2 = h_1 - n_x^B$:

$$
\begin{aligned}
y_2^{(1)} = \dot{\xi}_{2_1} &= \mathcal{L}_f H_2(\boldsymbol{x}) + \mathcal{L}_g H_2(\boldsymbol{x}) \\
&= \dot{h}_1 = h_2 r - h_3 q + \lambda_1 \\
&= \xi_{2_2}
\end{aligned} \tag{41}
$$

It is similarly seen that no direct relation with the input appears after the first order time derivative of y_2. Therefore, a second external state is added and the second order derivative is determined:

$$
y_2^{(2)} = \dot{\xi}_{2_2} = \mathcal{L}_f^2 H_2(\boldsymbol{x}) + \mathcal{L}_f \mathcal{L}_g H_2(\boldsymbol{x}) \tag{42}
$$

$$
\begin{aligned}
\mathcal{L}_f^2 H_2(\boldsymbol{x}) \quad &= (h_2 r - h_3 q + \lambda_1)\, r + h_2 \left(A_z pq + \tfrac{1}{I_z} \left(M_{G_z}^B + M_{a_z}^B - \gamma r \right) \right) - h_3 \left(A_y pr + \tfrac{1}{I_y} \left(M_{G_y}^B + M_{a_y}^B \right) \right) \\
&= \boldsymbol{\alpha_2}(\boldsymbol{x}) \\
\mathcal{L}_f \mathcal{L}_g H_2(\boldsymbol{x}) \quad &= \begin{bmatrix} h_2 G_r - h_3 G_q & -h_2 G_r - h_3 G_q & h_2 G_r + h_3 G_q \end{bmatrix} \boldsymbol{u} \\
&= \boldsymbol{\mathcal{B}_3}(\boldsymbol{x})\boldsymbol{u}
\end{aligned} \tag{43}
$$

Similarly, a direct relation with the input vector $\boldsymbol{u}$ is found in the second order time derivative of the second output, the relative degree is therefore $\rho_2 = 2$. The external state vector of the second output, $y_2 = h_1 - n_x^B$ can be taken as $\boldsymbol{\xi_x} = \begin{bmatrix} h_1 - n_x^B & \dot{h}_1 \end{bmatrix}^T$. Finally, the process is repeated for the last output, $y_3 = h_2 - n_y^B$.

$$
\begin{aligned}
y_3^{(1)} &= \dot{\xi}_{3_1} = \mathcal{L}_f H_3(\boldsymbol{x}) + \mathcal{L}_g H_3(\boldsymbol{x}) \\
&= h_3 p - h_1 r + \lambda_2 \\
&= \xi_{3_2}
\end{aligned} \tag{44}
$$

Here it is also found that the direct relation with the output appears in the second order time derivative of the output, $y_3^{(2)}$, as can be seen below:

$$
y_3^{(2)} = \dot{\xi}_{3_2} = \mathcal{L}_f^2 H_3(\boldsymbol{x}) + \mathcal{L}_f \mathcal{L}_g H_3(\boldsymbol{x}) \tag{45}
$$

$$
\begin{aligned}
\mathcal{L}_f^2 H_3(\boldsymbol{x}) &= (h_3 p - h_1 r + \lambda_2)\, r + h_3 \left(A_x q r + \frac{1}{I_x} \left(M_{G_x}^B + M_{a_x}^B \right) \right) - h_1 \left(A_z p q + \frac{1}{I_z} \left(M_{G_z}^B + M_{a_z}^B - \gamma r \right) \right) \\
&= \boldsymbol{\alpha_3(x)} \\
\mathcal{L}_f \mathcal{L}_g H_3(\boldsymbol{x}) &= \begin{bmatrix} h_3 G_p - h_1 G_r & -h_3 G_p + h_1 G_r & -h_3 G_p - h_1 G_r \end{bmatrix} \boldsymbol{u} \\
&= \boldsymbol{\mathcal{B}_3(x) u}
\end{aligned} \tag{46}
$$

The relative degree of the third output is therefore equal to $\rho_3 = 2$. The external state vector related to the third output y_3 can then be written as $\boldsymbol{\xi_y} = \begin{bmatrix} h_2 - n_y^B & \dot{h}_2 \end{bmatrix}^T$.

Now, the external state transformation has been completed and the system dynamics of the full system can be determined. The full external state vector is given by: [9]

$$
\begin{aligned}
\boldsymbol{\xi} &= \begin{bmatrix} \xi_1 & \xi_2 & \xi_3 & \xi_4 & \xi_5 & \xi_6 \end{bmatrix}^T \\
&= \begin{bmatrix} Z - Z_{ref} & V_z & h_1 - n_x^B & \dot{h}_1 & h_2 - n_y^B & \dot{h}_2 \end{bmatrix}^T
\end{aligned} \tag{47}
$$

Finally, based on this state vector and the Lie-derivatives of the output with respect to $\boldsymbol{f(x)}$ and $\boldsymbol{G(x)}$, the external dynamics of the system can be determined. Using the Lie-derivatives from eq. (40), eq. (43) and eq. (46), the external dynamics can be written as:

$$
\begin{aligned}
\dot{\boldsymbol{\xi}} &= \begin{bmatrix} \xi_2 \\ 0 \\ \xi_4 \\ 0 \\ \xi_6 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \boldsymbol{\alpha_1(x)} \\ 0 \\ \boldsymbol{\alpha_2(x)} \\ 0 \\ \boldsymbol{\alpha_3(x)} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ -n_z^I \kappa & -n_z^I \kappa & -n_z^I \kappa \\ 0 & 0 & 0 \\ h_2 G_r - h_3 G_q & -h_2 G_r - h_3 G_q & h_2 G_r + h_3 G_q \\ 0 & 0 & 0 \\ h_3 G_p - h_1 G_r & -h_3 G_p + h_1 G_r & -h_3 G_p - h_1 G_r \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^3 \end{bmatrix} \\
&= \boldsymbol{A_c \xi} + \boldsymbol{B_c} \left(\boldsymbol{\alpha(x)} + \boldsymbol{\mathcal{B}(x) u} \right) \\
\boldsymbol{y} &= \boldsymbol{C_c \xi}
\end{aligned} \tag{48}
$$

The canonical form state space matrices in eq. (48), $\boldsymbol{A_c}$,$\boldsymbol{B_c}$ and $\boldsymbol{C_c}$ are defined as:

$$
\boldsymbol{A_c} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{B_c} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{C_c} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{49}
$$

This concludes the state transformation of the external state dynamics, which provides a canonical form state space system that can be used for INDI control design. It must be noted that the system also has an uncontrollable part, because the total relative degree of the system, $\overline{\rho} = 6$, whereas the number of states $n = 7$. Therefore, an internal state is required to complete the state transformation. The derivation of the internal dynamics and its stability is elaborated upon in section III.E.

With the external state transformation completed, an INDI control law can be designed that controls the quadrotor altitude and attitude. First a virtual control law $\boldsymbol{\nu}$ is designed that provides the desired dynamics of $\boldsymbol{\xi}$. This virtual control law is designed as a regulator with a feedback gain matrix $\boldsymbol{K}$, as seen below:

$$\begin{aligned}\boldsymbol{\nu} &= -\boldsymbol{K}\boldsymbol{\xi} \\ &= -\begin{bmatrix} k_{0_z} & k_{1_z} & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{0_x} & k_{1_x} & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{0_y} & k_{1_y} \end{bmatrix}\boldsymbol{\xi}\end{aligned} \tag{50}$$

Given the virtual control law from eq. (50) and the external dynamics of eq. (48), the INDI control law can then be designed using eq. (32). This control law can be defined as:

$$\Delta\boldsymbol{u} = \mathcal{B}^{-1}\left(\boldsymbol{x}_f\right)\left(\nu - \boldsymbol{y}_f^{(\rho)}\right) \tag{51}$$

Based on the external dynamics from eq. (48), the nonlinear control effectiveness matrix $\boldsymbol{\mathcal{B}}(\boldsymbol{x})$ for this system is given by:

$$\boldsymbol{\mathcal{B}}(\boldsymbol{x}) = \begin{bmatrix} -n_z^I\kappa & -n_z^I\kappa & -n_z^I\kappa \\ h_2G_r - h_3G_q & -h_2G_r - h_3G_q & h_2G_r + h_3G_q \\ h_3G_p - h_1G_r & -h_3G_p + h_1G_r & -h_3G_p - h_1G_r \end{bmatrix} \tag{52}$$

As explained in eq. (33) and eq. (34), the system dynamics can now be rewritten as:

$$\begin{aligned}\dot{\boldsymbol{\xi}} &= (\boldsymbol{A_c} - \boldsymbol{B_c}\boldsymbol{K})\,\boldsymbol{\xi} + \boldsymbol{\delta}(\boldsymbol{\Delta x}, \Delta t) \\ \boldsymbol{y} &= \boldsymbol{C_c}\boldsymbol{\xi}\end{aligned} \tag{53}$$

This controller provides globally stable tracking of its reference signals if $(\boldsymbol{A_c} - \boldsymbol{B_c}\boldsymbol{K})$ is Hurwitz. In practice, however, the control effectiveness matrix $\boldsymbol{\mathcal{B}}(\boldsymbol{x})$ contains a singularity at $n_z^I = 0$. This occurs when the average thrust vector is pointed horizontally, which corresponds to a pitch or roll angle of $\pm\frac{\pi}{2}$ rad. Additionally, fixed-pitch rotors are incapable of providing negative thrust. Due to these factors, it is essential that the thrust vector in inertial space has an upwards vertical component, which means that $n_z^I < 0$ at all times. In scenarios whereby $n_z^I > 0$, an upset occurs and the quadrotor becomes uncontrollable. In that case an upset recovery controller is required to recover altitude control, such as the design presented by Sun et al. [8].

E. Internal Dynamics

Since the relative degree of the system is smaller than the number of states, i.e. $\overline{\rho} < n$, the system is not fully feedback-linearisable. With $n - \overline{\rho} = 1$, one internal state is needed to complete the state transformation [9]. This internal state is of critical importance, because the controlled system in eq. (53) can only be stable if the internal dynamics, $\dot{\eta} = f_\eta(\boldsymbol{\xi}, \eta)$ are stable. Because of the independence of the control inputs, the internal dynamics are uncontrollable, and the stability must be guaranteed up front [9][19].

As described in section III.C, there is no unique internal state belonging to the state transformation $\boldsymbol{\mathcal{T}}(\boldsymbol{x})$. It is however required that the selected internal state is independent of the control inputs $\boldsymbol{u}$ [9][19], which means that:

$$\frac{\partial\eta_i}{\partial x}\boldsymbol{G}(x)u = 0 \tag{54}$$

A candidate internal state is given in eq. (55), which is similar to the internal state selection by Sun et al. [6]. In this equation, the parameters μ_i are introduced. These constants are used to ensure that the internal state is independent of the output, in adherence to eq. (36)[6][9].

$$\eta = V_z + \mu_1 p + \mu_2 q + \mu_3 r \tag{55}$$

For the candidate internal state in eq. (55), eq. (54) is derived using the matrix $\boldsymbol{G(x)}$ from eq. (16). This shows that the candidate state can only be independent of $\boldsymbol{u}$ with proper selection of the constants μ_i [6].

$$\frac{\partial \eta_i}{\partial x}\boldsymbol{G}(x)u = \begin{bmatrix} 0 & 1 & 0 & 0 & \mu_1 & \mu_2 & \mu_3 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ \frac{n_z^I \overline{\kappa}}{m} & \frac{n_z^I \overline{\kappa}}{m} & \frac{n_z^I \overline{\kappa}}{m} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ G_p & -G_p & -G_p \\ G_q & G_q & -G_q \\ G_r & -G_r & G_r \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \end{bmatrix} \tag{56}$$

Selecting the parameters μ_i as seen in eq. (57) reduces the equality in eq. (56) to 0, which means that the condition in eq. (36) is met by the candidate function η. This means that the internal state η is independent of the inputs $\boldsymbol{u}$, thus making η a suitable internal state for the state transformation $\boldsymbol{\mathcal{T}(x)}$.

$$\mu_1 = \frac{n_z^I \kappa}{G_p m}, \quad \mu_2 = -\frac{n_z^I \kappa}{G_q m}, \quad \mu_3 = -\frac{n_z^I \kappa}{G_r m} \tag{57}$$

The parameters μ_i can be simplified to eq. (58), by separating the external variable n_z^I from the constant quadrotor parameters m, $\overline{\kappa}$, G_p, G_q and G_r, which are be contained in the constants c_i.

$$\mu_1 = n_z^I c_1, \quad \mu_2 = n_z^I c_3, \quad \mu_3 = n_z^I c_2 \tag{58}$$

Given the parameters μ_i, the internal state η of the state transformation can then be written as eq. (59).

$$\eta = V_z + n_z^I c_1 p + n_z^I c_2 + n_z^I c_3 r \tag{59}$$

Given the function $\boldsymbol{f(x)}$ in eq. (16) and the internal state in eq. (59), the internal dynamics as a function of $\boldsymbol{x}$ are derived to be:

$$\begin{aligned} \dot{\eta} = f_\eta(\boldsymbol{x}) &= \frac{\partial \eta}{\partial \boldsymbol{x}}\boldsymbol{f(x)} \\ &= \dot{V}_z + n_z^I\left(c_1\dot{p} + c_2\dot{q} + c_3\dot{r}\right) \end{aligned} \tag{60}$$

To assess the stability of the internal dynamics, however, the function f_η is to be assessed as a function of the transformed states $z = [\boldsymbol{\xi}; \eta]$[9]. For this purpose, the internal dynamics $f_\eta(\boldsymbol{x})$ are rewritten leading to the internal dynamics in terms of the transformed states, $\dot{\eta} = f_\eta(\boldsymbol{\xi}; \eta)$[9][6]. Due to the complexity, the full derivation of the internal dynamics is presented in the Appendix.

When the internal dynamics are input-to-state stable, the controlled external dynamics will be globally asymptotically stable as proven by Wang et al. [9]. Due to the complexity of the internal dynamics, however, it is infeasible to assess this property. However, it is feasible to prove a weaker form of stability, through the use of the so-called zero-dynamics. The zero dynamics, $f_0(\boldsymbol{0}, \eta)$, are the internal dynamics at the origin of the external states $\boldsymbol{\xi} = \boldsymbol{0}$[9][19] and signify the internal dynamics when the external states are tracked perfectly without error.

If the zero-dynamics are asymptotically stable, the full system is proven to be asymptotically stable within a region $\boldsymbol{D}$ near the origin $\boldsymbol{\xi} = \boldsymbol{0}$[9][19]. The internal dynamics can therefore be assumed stable under limited tracking error.

To obtain the zero dynamics, the external states in $\dot{\eta} = f(\boldsymbol{\xi}; \eta)$[9] shown in the Appendix are set to 0, leading to the zero-dynamics $\dot{\eta} = f_0(\eta)$ in eq. (61):

$$\dot{\eta} = f_0(\eta) = g - V_{z_{ref}} + \frac{c_3 \gamma n_z^B}{I_z(c_1 n_x^B + c_2 n_y^B + c_3 n_z^B)}\eta + \frac{n_z^B(c_2 A_y n_x^B + c_1 A_x n_y^B) + c_3 A_z n_x^B n_y^B}{n_z^I(c_1 n_x^B + c_2 n_y^B + c_3 n_z^B)^2}\eta^2 \tag{61}$$

What is immediately visible in eq. (61) is that a singularity exists for $n_z^I = 0$, leading to unbounded zero dynamics. This singularity also exists for the INDI controller of the outputs in section III.D, further demonstrating that the system can only be stable as long as the primary axis vector n^I has a vertical component pointing upwards. Assuming that

$n_z^I < 0$, the internal dynamics depend on the selection of the body frame primary axis vector $\boldsymbol{n}^B$. If the body frame primary axis vector is selected as $\boldsymbol{n}^B = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}$, the internal dynamics become:

$$\dot{\eta} = f_0(\eta) = g - V_{z_{ref}} - \frac{\gamma}{I_z}\eta \tag{62}$$

This equation is stable for all values of η, as long as $\boldsymbol{\xi} = \mathbf{0}$. However, as detailed by Mueller [4], it can be preferable to select nonzero values for n_x^B and n_y^B, since this decreases the required rotor inputs, whilst Sun [8] shows that doing so provides a stabilising precession moment. Therefore, a more efficient primary axis is selected, as previously used by Sun et al. [5], given by $\boldsymbol{n}^B = \begin{bmatrix} 0.2 & 0.2 & -0.96 \end{bmatrix}$. Inserting the value of $\boldsymbol{n}^B$ into eq. (61) leads to eq. (63):

$$\dot{\eta} = f_0(\eta) = g - V_{z_{ref}} - 0.59\eta + \frac{0.029}{n_z^I}\eta^2 \tag{63}$$

In this case, there is a dependence on n_z^I which affects the stability of the internal dynamics. For different values of n_z^I, the value of $\dot{\eta}$ from eq. (63) is plotted as a function of η in fig. 4.

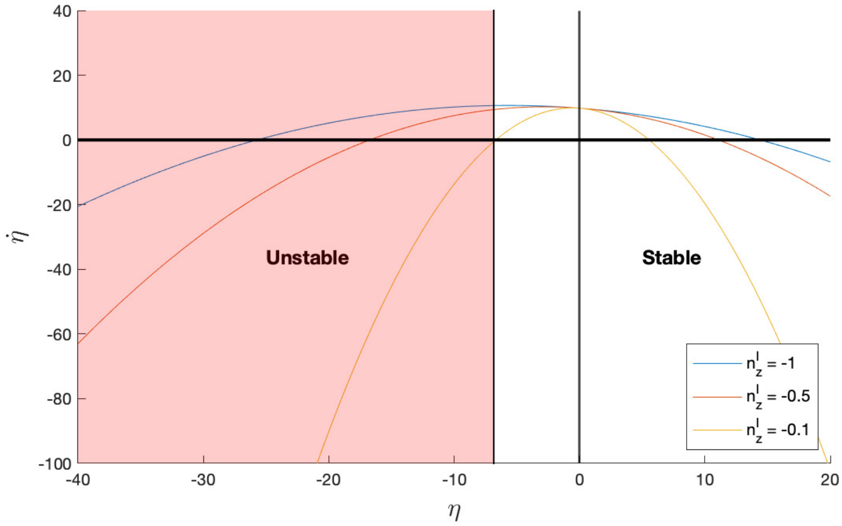

Fig. 4 The value of the derivative of the internal state $\dot{\eta}$ as a function of internal state η, depicted for various values of the inertial primary axis component n_z^I.

It can be seen that a stable equilibrium exists for $\dot{\eta}$ at the positive root of $f_0(\eta)$. Below this equilibrium value of η, the derivative $\dot{\eta}$ is positive, whilst above the equilibrium, the derivative $\dot{\eta}$ is negative, steering the internal state towards the equilibrium. The zero dynamics become unstable as soon as η is smaller than the second root of $f_0(\eta)$, where the derivative $\dot{\eta}$ becomes negative below its equilibrium value. The area shaded in red marks the values for which the internal dynamics are unstable in the worst case scenario of $n_z^I = -0.1$.

The equilibrium and the minimum value of $f_0(\eta)$ can be determined by finding the roots of the zero dynamics in eq. (63). For this selection of $\boldsymbol{n}^B$ the stable equilibrium η_0 and the unstable equilibrium η_{min} can be found by:

$$\eta_0 = \frac{0.59 + \sqrt{0.59^2 - 4(g - V_{z_{ref}})\frac{0.029}{n_z^I}}}{\frac{0.058}{n_z^I}}, \quad \eta_{min} = \frac{0.59 - \sqrt{0.59^2 - 4(g - V_{z_{ref}})\frac{0.029}{n_z^I}}}{\frac{0.058}{n_z^I}} \tag{64}$$

Therefore, the internal state η is stable as long as $\eta > \eta_{min}$ and $n_z^I < 0$. This means that the transformed states $z(\boldsymbol{\xi};\eta)$ can be stabilised by the INDI controller around $\boldsymbol{\xi} = 0$ for as long as the stability conditions for the internal states are met [9]. The exact region for which the internal dynamics are stable, however, is unknown and would be a recommendation for further work.

IV. Actuator Effects

It is now proven that the INDI controller from section III.C is stable around the origin $\boldsymbol{\xi}$, as long as $n_z^I < 0$. However, this is assuming affine control, ignoring the effect of actuator dynamics and rotor speed limits [6]. In reality, the actuators have a finite bandwidth and add delay to the system, whilst their rotational speed is limited by $\omega_{min} \leq \omega_i \leq \omega_{max}$, leading to saturation. Due to these actuator related factors, discrepancies between the command and the control realisation may occur, leading to unexpected tracking error and potentially to instability. Flight tests of the Bebop 2 quadrotor in the SRF case have already shown that actuator effects have a strong impact of the stability at high speed and can ultimately lead to loss-of-control [5][6].

To quantify this effect on stability, the effect of actuator dynamics and rotor saturation on the system dynamics is investigated. First the effects of the actuator dynamics are analysed, leading up to the implementation of actuator compensation for the inner loop INDI controller in section IV.B. Subsequently, the effects of rotor saturation on the error dynamics is investigated. This culminates in a method to assess the stability of the quadrotor under perturbation due to rotor saturation.

A. Effect of Actuator Dynamics

Despite ignoring the actuator dynamics, INDI control can provide stability around a trajectory. This requires sufficient robustness to the singular perturbation caused by the actuator lag, which is generally the case when the actuator bandwidth is greater than the control bandwidth [20]. However, doing so may lead to unpredictable error dynamics as demonstrated by Steffensen[13], who proposes a method to quantify the effects of actuator dynamics on INDI tracking performance.

Conventionally, it is assumed that the INDI control increment, $\boldsymbol{\Delta u}$ is achieved instantaneously within one time step, Δt. Since the control frequency for the quadrotor equals 512 Hz, it can be assumed that the actuators are generally not fast enough to realise such swift increments. Steffensen [13] therefore proposes to rewrite the error dynamics using the derivative of the input vector $\dot{\boldsymbol{u}}$, instead of using $\boldsymbol{\Delta u}$ from eq. (31).

The first step is to assess the dynamics of the rotor speeds, ω, which have previously been modelled as first order lag models by Smeur [11] and Lu [7]. Given the actuator bandwidth ω_B, the rotor speed dynamics can be written as [7][11]:

$$\dot{\omega} = \omega_B(\omega_c - \omega_0) \tag{65}$$

Since the input $\boldsymbol{u}$ is the vector of squared rotor speeds, $\boldsymbol{\omega}^2$, the actuator dynamics of eq. (65) do not yet equal $\dot{\boldsymbol{u}}$. Therefore, these input dynamics are derived using the rotor dynamics of eq. (65), as seen below [12]:

$$\begin{aligned} \dot{\boldsymbol{u}} &= \frac{\partial \boldsymbol{u}}{\partial \omega}\dot{\omega} \\ &= \text{diag}\left(\begin{bmatrix} 2\omega_1 & 2\omega_2 & 2\omega_3 \end{bmatrix}\right)\omega_B\left(\boldsymbol{\omega_c} - \boldsymbol{\omega_0}\right) \\ &= 2\omega_B\text{diag}(\boldsymbol{\omega_0})\left(\boldsymbol{\omega_c} - \boldsymbol{\omega_0}\right) \end{aligned} \tag{66}$$

As demonstrated by Smeur [11], the input increment $\boldsymbol{\Delta u}$ can be approximated by a Taylor series expansion, leading to a reformulation of the input increment $\boldsymbol{\Delta u}$:

$$\begin{aligned} \boldsymbol{\Delta u} &= \boldsymbol{u} - \boldsymbol{u}_0 \\ &\approx \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\omega}}\boldsymbol{\Delta\omega} \\ &\approx 2\text{diag}(\boldsymbol{\omega_0})\left(\boldsymbol{\omega}_c - \boldsymbol{\omega_0}\right) \end{aligned} \tag{67}$$

Substituting eq. (67) into eq. (66), subsequently leads to a first order approximation of the actuator dynamics based on $\boldsymbol{\Delta u}$:

$$\dot{\boldsymbol{u}} \approx \omega_B\boldsymbol{\Delta u} \tag{68}$$

Smeur [11] and Steffensen [13] have demonstrated that for first order actuator dynamics, the error dynamics track the desired error dynamics with a first order lag too. To demonstrate this effect, the output time derivative $\boldsymbol{y}^{(\rho)}$ from eq. (31) is differentiated once more using the methods described by Steffensen [13], leading to the output derivative of order$(\rho + 1)$[13]:

$$y^{(\rho+1)} = \frac{\partial[\alpha(x) + \mathcal{B}(x)u]}{\partial x}\dot{x} + \mathcal{B}(x)\dot{u} \tag{69}$$

The assumption can be made that the state-dependent term in the output derivative $y^{(\rho+1)}$ in eq. (69) is negligible for a sufficiently high sample rate, as detailed by Steffensen [13]. Therefore, these terms are substituted by the derivative of the perturbation term $\dot{\delta}(z, \Delta t)$. Substituting the actuator dynamics of eq. (68) into the error dynamics of eq. (69) then leads to:

$$y^{(\rho+1)} = \mathcal{B}(x_0)\omega_B \Delta u + \dot{\delta}(z, \Delta t) \tag{70}$$

Finally, the control law from eq. (51) is substituted for Δu in eq. (71), which shows the approximated output dynamics when including the effect of actuator lag:

$$y^{(\rho+1)} = \omega_B \left(\nu - y^{(\rho)}\right) + \dot{\delta}(z, \Delta t) \tag{71}$$

This ultimately shows that the ρ-th output derivative follows the commanded virtual control signal ν with a first order lag due to the ignored actuator dynamics [11][13]. This effect can be illustrated clearly in the Laplace domain. For this purpose, both the ideal dynamics without actuator delay from eq. (31) and the delayed system from eq. (71) are transformed to the Laplace domain [13]. This transform is performed on the dynamics of output y_i, given the virtual control law of eq. (29) . Given that the relative degree, $\rho = 2$, for all of the outputs, the ideal Laplace domain output dynamics of y_i given the virtual control ν from eq. (53) become:

$$Y_{i_{ideal}}(s)\left(s^2 + k_1 s + k_0\right) = 0 \tag{72}$$

In eq. (72) s signifies the complex frequency in the Laplace domain. When taking into account the first order actuator lag with frequency ω_B, these output dynamics become eq. (73) [13]

$$Y_{i_{actual}}(s)\left(s^2 + k_1 s + k_0\right)(s + \omega_B) = 0 \tag{73}$$

Expanding eq. (73), the actual error dynamics under the influence of actuator dynamics become:

$$Y_{i_{actual}}(s)\left(s^3 + (k_1 + \omega_B)s^2 + (k_1\omega_B + k_0)s + k_0\omega_B\right) = 0 \tag{74}$$

This means that the error dynamics from eq. (53) do not reflect the true error dynamics and can lead to discrepancies between the desired and realised error dynamics. Analysis of the effect of actuator dynamics and saturation on the stability of the quadrotor becomes more complicated if the desired controlled dynamics do not reflect the actual dynamics.

B. INDI with Actuator Compensation

It is now demonstrated that the external dynamics in reality do not correspond with the external dynamics shown in eq. (53) and are instead delayed by a first order lag as a result of actuator dynamics. For accurate stability analysis, however, it is essential that the external dynamics are representative of the system behaviour. Therefore, the system of eq. (16) is modified to include the effect of actuator dynamics. Subsequently, the INDI controller is updated, ultimately providing commands that are feasible within the actuator bandwidth [13]. As will be demonstrated at the end of this adaptation, the adapted controller is effectively the same controller from eq. (32), albeit delayed with a first order lag.

The first step in this process is to augment the state vector with the rotor speeds, ω to account for the first order lag due to actuator dynamics, leading to the extended state vector, x_a:

$$x_a = \begin{bmatrix} Z & V_z & h_1 & h_2 & p & q & r & \omega_1 & \omega_2 & \omega_3 \end{bmatrix}^T \tag{75}$$

Now that the rotor speeds ω have been reclassified as states, a new input vector must be defined for the extended state-space system. Given the first order rotor dynamics in eq. (65), the augmented input vector is taken as:

$$u_a = \begin{bmatrix} \omega_{c_1} & \omega_{c_2} & \omega_{c_3} \end{bmatrix}^T \tag{76}$$

Given the augmented state vector, $\boldsymbol{x_a}$ and the new input vector $\boldsymbol{u_a}$, the state space-system can be rewritten as:

$$\begin{bmatrix} \dot{Z} \\ \dot{V}_z \\ \dot{h}_1 \\ \dot{h}_2 \\ \dot{p} \\ \dot{q} \\ \dot{r} \\ \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix} = \begin{bmatrix} V_z \\ g + F^I_{a,z} - \frac{n^I_z \overline{\kappa}}{m}\left(\omega_1^2 + \omega_2^2 + \omega_3^2\right) \\ h_2 r - h_3 q + \lambda_1 \\ h_3 p - h_1 r + \lambda_2 \\ A_x qr + \frac{1}{I_x}\left(M^B_{G_x} + M^B_{a_x}\right) + G_p\left(\omega_1^2 - \omega_2^2 - \omega_3^2\right) \\ A_y pr + \frac{1}{I_y}\left(M^B_{G_y} + M^B_{a_y}\right) + G_q\left(\omega_1^2 + \omega_2^2 - \omega_3^2\right) \\ A_z pq + \frac{1}{I_z}\left(M^B_{G_z} + M^B_{a_z} - \gamma r\right) + G_r\left(\omega_1^2 - \omega_2^2 + \omega_3^2\right) \\ -\omega_1 \\ -\omega_2 \\ -\omega_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \omega_B & 0 & 0 \\ 0 & \omega_B & 0 \\ 0 & 0 & \omega_B \end{bmatrix} \begin{bmatrix} \omega_{c_1} \\ \omega_{c_2} \\ \omega_{c_3} \end{bmatrix} \tag{77}$$

Due to the extended state vector, the state transformation $\boldsymbol{z} = \boldsymbol{\mathcal{T}(x)}$ is no longer valid. The second step is therefore to adapt the state transformation to suit the new system, which requires analysis of the output dynamics. What can readily be observed is that through the addition of the actuator dynamics, the relative degree for each of the outputs has increased by one. In the ρ-th output derivative $\boldsymbol{y}^{(\rho)}$ the inputs $\boldsymbol{u_a}$ do not appear, appearing instead in the output derivative $\boldsymbol{y}^{(\rho+1)}$. The output time derivatives of order $\rho+1$, $\boldsymbol{y}^{(\rho+1)}$ are:

$$\boldsymbol{y}^{(\rho+1)} = \begin{bmatrix} \ddot{V}_z & \dddot{h}_1 & \dddot{h}_2 \end{bmatrix}^T \tag{78}$$

Making the assumption that the state dependent terms are negligible in eq. (70), the output dynamics can now be written as:

$$\boldsymbol{y}^{(\rho+1)} = 2\omega_B \mathrm{diag}(\boldsymbol{\omega}) \begin{bmatrix} \frac{n^I_z \kappa}{m} & \frac{n^I_z \kappa}{m} & \frac{n^I_z \kappa}{m} \\ h_2 G_r - h_3 G_q & -h_2 G_r - h_3 G_q & h_2 G_r + h_3 G_q \\ h_3 G_p - h_1 G_r & -h_3 G_p + h_1 G_r & -h_3 G_p - h_1 G_r \end{bmatrix} \begin{bmatrix} \omega_{c_1} - \omega_1 \\ \omega_{c_2} - \omega_2 \\ \omega_{c_3} - \omega_3 \end{bmatrix} + \begin{bmatrix} \dot{\delta}_1(z, \Delta t) \\ \dot{\delta}_2(z, \Delta t) \\ \dot{\delta}_3(z, \Delta t) \end{bmatrix} \tag{79}$$

Considering that the external state vector consists of a chain of output time derivatives, the external state vector is extended with the output derivatives $\boldsymbol{y}^{(\rho)}$, leading to the augmented external state vector $\boldsymbol{\zeta}$ in eq. (80):

$$\begin{aligned} \boldsymbol{\zeta} &= \begin{bmatrix} y_1, & \dot{y}_1 & \ddot{y}_1 & y_2, & \dot{y}_2 & \ddot{y}_2 & y_3, & \dot{y}_3 & \ddot{y}_3 \end{bmatrix}^T \\ &= \begin{bmatrix} Z - Z_{ref}, & V_z - V_{z_{ref}}, & a_z - a_{z_{ref}} & h_1 - n^B_x, & \dot{h}_1, & \ddot{h}_1 & h_2 - n^B_y, & \dot{h}_2, & \ddot{h}_2 \end{bmatrix}^T \end{aligned} \tag{80}$$

The external dynamics of the augmented system can then be rewritten using this new representation. Extending the external dynamics of eq. (48) with the dynamics of the output derivatives of order $(\rho+1)$ leads to the external dynamics in eq. (81) [9].

$$\dot{\boldsymbol{\zeta}} = \boldsymbol{A_a}\boldsymbol{\zeta} + \boldsymbol{B_a}\left(\frac{\partial[\boldsymbol{\alpha}(x) + \boldsymbol{\mathcal{B}}(x)\boldsymbol{u}]}{\partial x}\dot{\boldsymbol{x}} + \boldsymbol{\mathcal{B}_a}(\boldsymbol{x_a})\omega_B\left(\boldsymbol{\omega_c} - \boldsymbol{\omega_0}\right)\right) \tag{81}$$

In this equation, the matrices $\boldsymbol{A_a}$ and $\boldsymbol{B_a}$ bring the dynamics of the external states into the canonical form.

The nonlinear control effectiveness matrix is updated with the rotor dynamics to become:

$$\boldsymbol{\mathcal{B}_a}(\boldsymbol{x_a}) = 2\mathrm{diag}(\boldsymbol{\omega})\boldsymbol{\mathcal{B}}(\boldsymbol{x}) \tag{82}$$

Given the new output dynamics from eq. (81), a new control law can then be designed that compensates for the actuator dynamics and computes feasible control increments given the actuator bandwidth [13]. Assuming that the perturbation term $\dot{\boldsymbol{\delta}}(z, \Delta t)$ is negligible for a sufficiently high sampling rate, it was shown that the Laplace domain dynamics of the output derivative $y_i^{(\rho+1)}$ are provided by eq. (74). Taking the inverse Laplace transform of this equation, the time domain output dynamics of $y_i^{(\rho+1)}$ become [13]:

$$y_i^{(\rho+1)} + (k_1 + \omega_B)\, y_i^{(\rho)} + (k_1\omega_B + k_0)\, y_i^{(1)} + k_0\omega_B y_i^{(0)} = 0 \tag{83}$$

By solving this equation for $\boldsymbol{y}^{(\rho+1)}$, a virtual control law $\boldsymbol{\nu_a}$ can be designed for the extended states which reflects the actual output dynamics of the system and prevents commands that are infeasible within the actuator bandwidth [13]. This virtual control $\boldsymbol{\nu_a}$ is seen below:

$$\nu_{a_i} \quad = -\,(k_1 + \omega_B)\, y^{(\rho)} - (k_1\omega_B - k_0)\, \boldsymbol{y}^{(1)} - k_0\omega_B \boldsymbol{y}^{(0)} \tag{84}$$

Given that the external states are equal to the output derivatives $y_i^{(j)}$, the virtual control law ν_{a_i} for the output y_i can be defined as:

$$\boldsymbol{\nu_a} = -\boldsymbol{K_a}\boldsymbol{\zeta} \tag{85}$$

In this equation, $\boldsymbol{K_a}$ is the augmented gain matrix, which can be derived from the virtual control law in eq. (84). The augmented gain matrix then becomes:

$$\boldsymbol{K_a} = \begin{bmatrix} k_0\omega_B & k_0 + k_1\omega_B & k_1 + \omega_B & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & k_0\omega_B & k_0 + k_1\omega_B & k_1 + \omega_B & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & k_0\omega_B & k_0 + k_1\omega_B & k_1 + \omega_B \end{bmatrix} \tag{86}$$

Given the augmented virtual control law, $\boldsymbol{\nu_a}$, and the augmented nonlinear control effectiveness matrix, $\boldsymbol{\mathcal{B}_a(x_a)}$, a control law can be designed that inverts the dynamics of the extended external states and leads to the desired dynamics of eq. (84). This controller can be chosen as [13]:

$$\begin{aligned} \boldsymbol{\omega_c} \quad &= (\boldsymbol{\mathcal{B}_a(x_a)}\omega_B)^{-1}\left(-\,(k_1 + \omega_B)\,\boldsymbol{y}^{(\rho)} - (k_1\omega_B - k_0)\,\boldsymbol{y}^{(1)} - k_0\omega_B\boldsymbol{y}^{(0)}\right) + \boldsymbol{\omega_0} \\ &= (\boldsymbol{\mathcal{B}_a(x_a)}\omega_B)^{-1}\left(-\boldsymbol{K_a}\boldsymbol{\zeta}\right) + \boldsymbol{\omega_0} \end{aligned} \tag{87}$$

If the control law from eq. (87) is applied to the external dynamics as seen in eq. (81), the external dynamics of the controlled system can be rewritten to become eq. (88):[9]

$$\dot{\boldsymbol{\zeta}} = (\boldsymbol{A_a} - \boldsymbol{B_a}\boldsymbol{K_a})\,\boldsymbol{\zeta} + \boldsymbol{B_a}\dot{\boldsymbol{\delta}}(z, \Delta t) \tag{88}$$

What has effectively been achieved is that the external states $\boldsymbol{\zeta}$ now reflect the actual controlled output dynamics of the quadrotor, given the bandwidth of the rotors. The new controller is effectively the same INDI controller design of section III.D, but now delayed by the first order actuator dynamics [13]. This can be demonstrated by application of the control law in eq. (87) with ideal actuators with infinite bandwidth, as shown by Steffensen [13]. With these ideal actuators, this control law reduces to the original INDI controller eq. (32) in section III.C. This can be shown by taking the limit of $\boldsymbol{\omega_c}$ as the actuator bandwidth $\omega_B \rightarrow \infty$ [13]:

$$\begin{aligned} \lim_{\omega_B\to\infty} \boldsymbol{\omega_c} \quad &= \lim_{\omega_B\to\infty}\left((\boldsymbol{\mathcal{B}_a(x_a)}\omega_B)^{-1}\left(-\,(k_1 + \omega_B)\,\boldsymbol{y}^{(\rho)} - (k_1\omega_B - k_0)\,\boldsymbol{y}^{(1)} - k_0\omega_B\boldsymbol{y}^{(0)}\right) + \boldsymbol{\omega_0}\right) \\ &= (2\mathrm{diag}(\boldsymbol{\omega})\boldsymbol{\mathcal{B}(x)})^{-1}\left(-k_0\boldsymbol{y}^{(0)} - k_1\boldsymbol{y}^{(1)} - \boldsymbol{y}^{(\rho)}\right) + \boldsymbol{\omega_0} \\ &= (2\mathrm{diag}(\boldsymbol{\omega})\boldsymbol{\mathcal{B}(x)})^{-1}\left(-\boldsymbol{K}\boldsymbol{\xi} - \boldsymbol{y}^{(\rho)}\right) + \boldsymbol{\omega_0} \end{aligned} \tag{89}$$

Taking $\boldsymbol{\omega_0}$ to the left-hand side and multiplying both sides with $2\mathrm{diag}(\boldsymbol{\omega_0})$, leads to:

$$\begin{aligned} 2\mathrm{diag}(\boldsymbol{\omega_0})(\boldsymbol{\omega_c} - \boldsymbol{\omega_0}) &= \boldsymbol{\mathcal{B}(x)}^{-1}\left(-\boldsymbol{K}\boldsymbol{\xi} - \boldsymbol{y}^{(\rho)}\right) \\ &= \boldsymbol{\Delta u} \end{aligned} \tag{90}$$

The left hand side of this equation is equivalent to $\boldsymbol{\Delta u}$ as shown in eq. (67), whereas the right hand side is the original ideal control law for $\boldsymbol{\Delta u}$ without the influence of the actuator delay. This demonstrates that the augmented control law from eq. (87) is simply the original INDI control law eq. (32), delayed by the first order actuator dynamics [13].

To complete the augmented state transformation for the actuator compensated controller, it is necessary to assess the existence of uncontrollable internal states. As mentioned in section III.C, the number of uncontrollable internal state equals the number of states in $\boldsymbol{x_a}$ minus the total sum of the relative degrees of the transformed system. For the extended external state vector, the relative degree of each of the outputs, $\rho_i = 3$, leading to a total relative degree of $\overline{\rho} = 9$. With the existence of $n = 10$ states in $\boldsymbol{x_a}$, 1 uncontrollable state remains to complete the state transformation. With the extension of the state vector, the original internal state selection, η, is still independent of the inputs, as the condition in eq. (36) still holds for the extended system. Given the external dynamics from eq. (88) and the internal dynamics eq. (63), the dynamics of the extended state transformation become:

$$\begin{aligned}
\dot{\boldsymbol{\eta}} &= \boldsymbol{f}_0(\boldsymbol{\eta}, \boldsymbol{\xi}) \\
\dot{\boldsymbol{\zeta}} &= (\boldsymbol{A}_a - \boldsymbol{B_a K_a})\boldsymbol{\zeta} + \boldsymbol{B}_a \dot{\boldsymbol{\delta}}(z, \Delta t) \\
\boldsymbol{y} &= \boldsymbol{C}_a \boldsymbol{\zeta}
\end{aligned} \tag{91}$$

C. Attainable Virtual Control Set

The effect of actuator dynamics on the dynamics of the quadrotor has now been derived, but another actuator effect has a strong influence on the quadrotor dynamics. Due to rotor speed limits of the remaining actuators, not all virtual controls $\boldsymbol{v_a}$ can be realised by the quadrotor. This can lead to tracking error and can ultimately lead to loss-of-control, as demonstrated in high-speed flight tests by Sun et al. [5][6].

Upon rotor loss, the maximum thrust is reduced by a fourth, reducing the maximum attainable acceleration. But more importantly, the three remaining functional rotors are placed asymmetrically with respect to the center of gravity of the quadrotor, greatly reducing the set of attainable control torques compared to a nominal quadrotor as previously demonstrated by Sun [21].

To illustrate the effect of rotor loss on the controllability of the quadrotor, the convex hull of all attainable virtual control realisations is derived. This Attainable Virtual Control Set (AVCS) [22] encapsulates all of the feasible control realisations for thrust and attitude control and signifies the control authority of the actuators for the given system. To make the results more intuitive for interpretation, the thrust is used in the determination of the AVCS, instead of $\ddot{y}_1 = a_z$. The attitude control realisations are represented by the output derivatives $\ddot{y}_2 = \ddot{h}_1$ and $\ddot{y}_3 = \ddot{h}_2$, the second order derivatives of the horizontal primary axis reference components. These variables can be thought of as the angular accelerations of the thrust vector.

For a hover condition upon single rotor failure, with perfect angular tracking of n^B, with $\boldsymbol{h} = \boldsymbol{n}^B = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}$, the convex hull of the all the possible control realisations $\boldsymbol{v_R}$, is determined. The control realisation $\boldsymbol{v_R}$ is determined using:

$$\begin{aligned}
\boldsymbol{v_R} &= \boldsymbol{\mathcal{B}_R u_R} \\
&= \begin{bmatrix} -\kappa & -\kappa & -\kappa \\ h_3 G_p - h_1 G_r & -h_3 G_p + h_1 G_r & -h_3 G_p - h_1 G_r \\ h_1 G_q - h_2 G_p & h_1 G_q + h_2 G_p & -h_1 G_q + h_2 G_p \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \end{bmatrix}
\end{aligned} \tag{92}$$

The AVCS then signifies the set $\boldsymbol{S_R}$, which contains all the vectors $\boldsymbol{v_R}$ that are feasible within the rotor speed constraints. The set $\boldsymbol{S_R}$ is described by:

$$\boldsymbol{S_R} = \left\{ \boldsymbol{v_R} \in \mathbb{R}^3 \quad | \quad \omega_{min} \leq \omega_i \leq \omega_{max} \right\} \tag{93}$$

The computation of the AVCS is done numerically using a mesh of all possible combinations of rotor speeds $\boldsymbol{\omega}$, with a resolution of 10 rad/s. This mesh is transformed using eq. (92) into a mesh of control realisations that signifies the set $\boldsymbol{S_R}$. Subsequently, the convex hull of this set is determined, leading to the AVCS volume in fig. 5.

In isolation, the volume depicted in fig. 6 is not very informative. Therefore, the attainable attitude control realisations $\ddot{h}_1$ and $\ddot{h}_2$ are depicted for various fixed thrust magnitudes in fig. 6(a). In physical terms, this represents the set of angular acceleration of the thrust vector that can be achieved given the rotor speed limits, for a given thrust setting.

What can be seen in eq. (93), is that the AVCS at any thrust level has an irregular shape with an offset from 0, leading to a coupling effect between the control realisations for $\ddot{h}_1$, $\ddot{h}_2$ and T. Due to this coupling, it is possible that uncommanded attitude control is provided to both of the axes. Additionally, the set of attainable angular controls

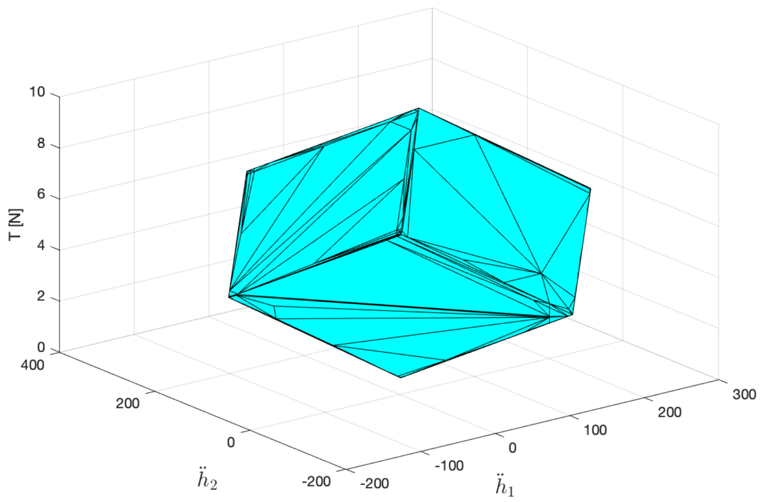

Fig. 5 The convex hull of the attainable virtual control set (AVCS).

varies strongly with the desired thrust level and vice versa. These two factors combined lead to a strong coupling between the three outputs and may lead to unexpected discrepancies between the commanded virtual control and the control realisation. A high thrust command might for example lead to uncommanded angular control realisations and subsequently destabilise the quadrotor. Similarly, certain angular control commands may lead to an insufficient thrust realisation.

The limitation of the angular control authority can be somewhat alleviated by generating a constant precession torque, as described by Sun et al. [8]. This can be achieved by selecting nonzero values for the body frame primary axis components, n_x^B and n_y^B, which leads a relaxed hover equilibrium with constant nonzero pitch and roll rates. This can be derived from the kinematics of $\boldsymbol{n}^B$ in eq. (94). Since the vector $\boldsymbol{n}^B$ is constant, its derivative has to be equal to 0. For nonzero selections of n_x^B and n_y^B and a constant yaw rate, constant nonzero pitch and yaw rate is required to maintain the equilibrium[4][5][8]:

$$\dot{\boldsymbol{n}}^B = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \begin{bmatrix} n_x^B \\ n_y^B \\ n_z^B \end{bmatrix} = \boldsymbol{0} \tag{94}$$

Nonzero pitch and yaw rate, combined with the constant yaw rate will generate a torque with [8]:

$$\boldsymbol{M}_{prec} = \boldsymbol{\Omega}^B \times \boldsymbol{I}_v \boldsymbol{\Omega} \tag{95}$$

This subsequently affects the attitude dynamics of $\ddot{\boldsymbol{h}}$, through the kinematics as seen below:

$$\ddot{\boldsymbol{h}}_{prec} = \boldsymbol{I}_v^{-1} \left(\boldsymbol{\Omega}^B \times \boldsymbol{I}_v \boldsymbol{\Omega} \right) \times \boldsymbol{h} \tag{96}$$

Given this effect of the precession torque on $\ddot{\boldsymbol{h}}$, the control realisations can be formulated as eq. (97). In this determination, it is once again assumed that the attitude tracking error is 0, so $\boldsymbol{h} = \boldsymbol{n}^B$.

$$\boldsymbol{\nu_R} = \begin{bmatrix} -n_z^I \kappa & -n_z^I \kappa & -n_z^I \kappa \\ h_3 G_p - h_1 G_r & -h_3 G_p + h_1 G_r & -h_3 G_p - h_1 G_r \\ h_1 G_q - h_2 G_p & h_1 G_q + h_2 G_p & -h_1 G_q + h_2 G_p \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^3 \end{bmatrix} + \boldsymbol{I}_v^{-1} \left(\boldsymbol{\Omega}^B \times \boldsymbol{I}_v \boldsymbol{\Omega} \right) \times \boldsymbol{h} \tag{97}$$

A primary axis vector of $\boldsymbol{n}^B = [\begin{matrix} 0.2 & 0.2 & -0.96 \end{matrix}]^T$ was found to reduce the required control effort by Mueller et al [4] and was also employed by Sun et al.[5][8]. The attainable attitude control realisations with and without precession

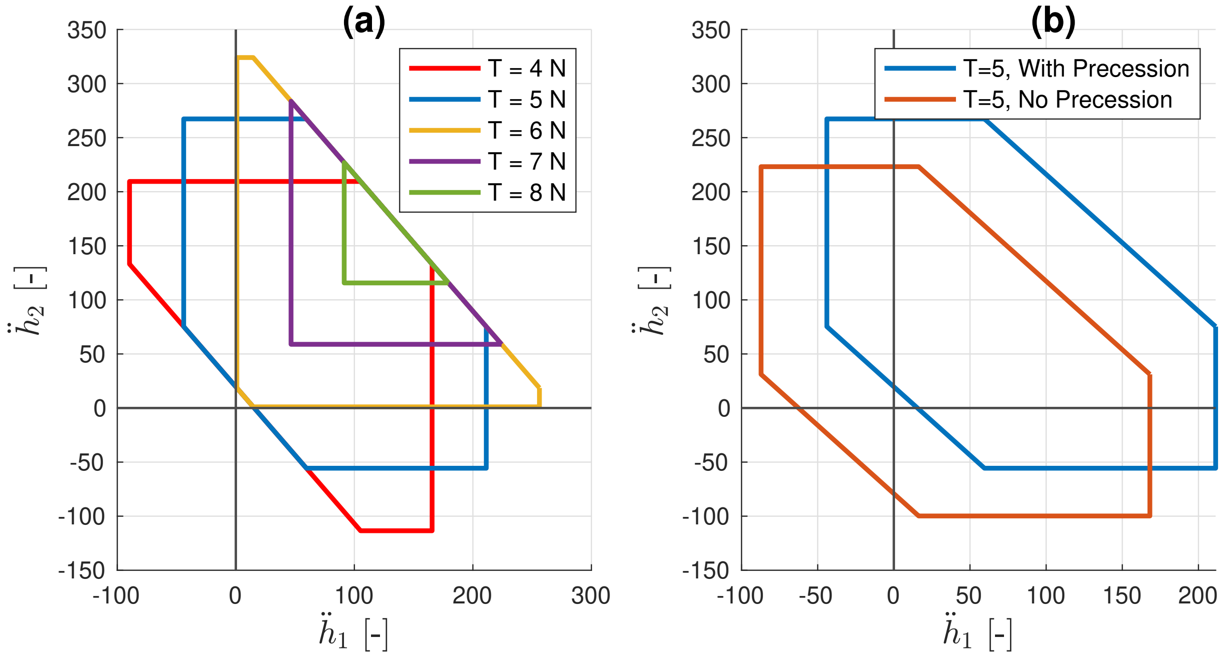

Fig. 6 (a) Depicts the AVCS of $\ddot{h}_1$ and $\ddot{h}_2$ at specified thrust levels T. (b) Depicts the AVCS of $\ddot{h}_1$ and $\ddot{h}_2$ at $T = 5$ with and without precession.

are compared at a thrust of 5 N in fig. 6(b). What is visible is that this precession torque alleviates the imbalance in the attitude authority and potentially decreases the likelihood of saturation. However, the shape of the AVCS is still asymmetric and the possibility of generating uncommanded control realisations for both $\ddot{h}_1$ and $\ddot{h}_2$ remains.

D. Effect of Actuator Saturation

It is visible that rotor loss imposes limitation on the attainable control set of the quadrotor. Sun et al. [5][6] have hypothesised that these actuator limitations are the main cause behind the loss of control occurrences in high-speed flight tests using a SRF and DRF Parrot Bebop 2 quadrotor [5][6]. It was shown that prior to loss of control, the frequency of rotor saturations grew, culminating in a crash [5][6]. To prove this fact and possibly predict loss of control situations, the effect of actuator saturation has to be quantified in the error dynamics.

To begin, the actual commanded input $\boldsymbol{\omega_c}$ is modified to incorporate the effect of actuator saturation in eq. (98). It is assumed that both the rotor speed and the rotor speed command are subject to saturation with $\boldsymbol{\omega} \in [\omega_{min}, \omega_{max}]$ and $\boldsymbol{\omega_c} \in [\omega_{min}, \omega_{max}]$. The hedged rotor speed, $\boldsymbol{\omega_h}$, is introduced, which signifies the difference between the rotor speed computed by the INDI controller, compared to the command that is sent to the actuator. Due to command saturation, this command differs from the desired control input. This concept of hedged rotor speed is based on the concept of pseudo-control hedging as first defined by Johnson and Calise [23].

$$\boldsymbol{\omega_c} = (\mathcal{B}_{\boldsymbol{a}}(\boldsymbol{x_a})\omega_B)^{-1}(\boldsymbol{\nu_c}) + \boldsymbol{\omega_0} - \boldsymbol{\omega_h} \tag{98}$$

The hedged rotor speed command therefore signifies the difference between the rotor speeds calculated by the INDI controller minus the actual command after applying the rotor speed constraints, as seen below:

$$\boldsymbol{\omega_h} = (\mathcal{B}_{\boldsymbol{a}}(\boldsymbol{x_a})\omega_B)^{-1}(\boldsymbol{\nu_c}) + \boldsymbol{\omega_0} - \boldsymbol{\omega_c} \tag{99}$$

The first order actuator dynamics from Steffensen [13] can then be modified by addition of the hedged input value $\boldsymbol{\omega_h}$, leading to eq. (100).

$$\dot{\omega} = \omega_B(\boldsymbol{\omega_c} - \boldsymbol{\omega_0} - \omega_h) \tag{100}$$

The dynamics of the output derivative $\boldsymbol{y}^{(\rho+1)}$ in eq. (70) [13][11] are adjusted with the adjusted actuator dynamics from eq. (100), leading to the following dynamics:

$$\boldsymbol{y}^{(\rho+1)} = \boldsymbol{\mathcal{B}_a}(\boldsymbol{x_a})\omega_B\left(\boldsymbol{\omega_c} - \boldsymbol{\omega_0} - \omega_h\right) \tag{101}$$

When applying the actuator compensated control law from eq. (87) to the dynamics in eq. (101), the output dynamics of $\boldsymbol{y}^{(\rho+1)}$ become:

$$\begin{aligned} \boldsymbol{y}^{(\rho+1)} &= \boldsymbol{\nu_a} - \boldsymbol{\mathcal{B}_a}(\boldsymbol{x_a})\omega_B\boldsymbol{\omega_h} \\ &= -\boldsymbol{K_a}\boldsymbol{\xi} - \boldsymbol{\nu_h} \end{aligned} \tag{102}$$

In the output dynamics of eq. (102), the hedged virtual control, $\boldsymbol{\nu_h}$ is introduced, which is equal to [23][10]:

$$\boldsymbol{\nu_h} = \boldsymbol{\mathcal{B}_a}(\boldsymbol{x_a})\omega_B\boldsymbol{\omega_h} \tag{103}$$

This variable can be used to quantify the tracking performance when perturbed by actuator saturation, when compared to the commanded virtual control $\boldsymbol{\nu_a}$. Given the tracking error and measured disturbances acting on $y^{(\rho+1)}$, the virtual control signal $\boldsymbol{\nu_a}$ signifies the required virtual control to provide the desired error dynamics. The hedged virtual control signal $\boldsymbol{\nu_h}$ then signifies the inability of the actuators to realise the desired error dynamics for the given trajectory [10]. The larger the magnitude of the hedged virtual control signal, the larger the discrepancy between the desired error dynamics and the actual error dynamics, which could potentially lead to a loss of control.

The actual control that is exerted on the system can be quantified by the control realisation, $\boldsymbol{\nu_r}$.

$$\boldsymbol{\nu_r} = \boldsymbol{\nu_a} - \boldsymbol{\nu_h} \tag{104}$$

In fig. 7, the hedged virtual control is visualised for a fictional control system. The top subfigure shows the virtual control in blue, with the orange representing the control realisation. This control realisation is limited by the occurrence of saturation at $\boldsymbol{\nu_r} = 1.5$. The hedged virtual control $\boldsymbol{\nu_h}$ is depicted in the bottom figure, which clearly depicts how the hedged virtual control signifies the discrepancy between the control command and the control realisation. This hedged virtual control, $\boldsymbol{\nu_h}$, therefore signifies an uncommanded perturbation on the controlled system, leading to tracking error and potentially to instability.

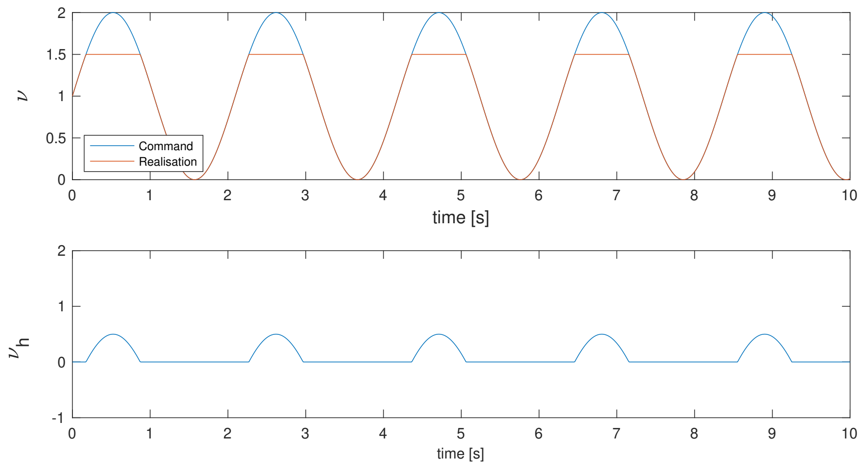

Fig. 7 A fictional visual representation of the hedged virtual control signal in case of infeasible virtual control signal due to actuator saturation. The upper figure depicts the commanded virtual control, compared to the control realisation. The bottom figure shows the hedged virtual control associated with this control realisation.

Another scenario may also lead to discrepancies between control command and realisation. As mentioned, the limited volume of the AVCS can lead to coupling between the control realisations of the different outputs. For example, a very high thrust command can lead to uncommanded attitude control realisations. This occurs when an attitude command is given that lies outside of the AVCS in fig. 6 This situation is visually represented for a fictional system in fig. 8, where it can be seen that the control realisation is markedly different from the command for an unknown reason.

The hedged virtual control similarly represents the discrepancy and describes the undesireable perturbation that is added to the system due to actuator effects.

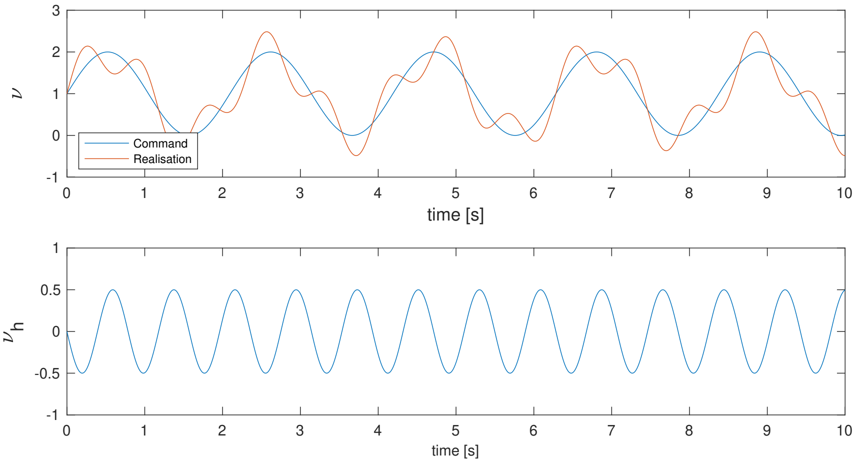

Fig. 8 A fictional visual representation of the hedged virtual control signal when uncommanded control realisations occur. The upper figure depicts the commanded virtual control, compared to the control realisation. The bottom figure shows the hedged virtual control associated with this control realisation.

By applying the perturbed output dynamics to the external state dynamics from eq. (88), the external dynamics can be rewritten as eq. (105): [9]

$$\dot{\zeta} = (\boldsymbol{A_a} - \boldsymbol{B_a}\boldsymbol{K_a})\,\zeta - \boldsymbol{B_a}\boldsymbol{\nu_h} + \dot{\delta}(z, \Delta t) \tag{105}$$

The external state dynamics of eq. (105) finally show the output dynamics of the SRF quadrotor when the actuator dynamics and saturation effects are considered. The first term on the right-hand side of eq. (105) are the system dynamics under the influence of actuator lag. Now, with the addition of the hedged virtual control signal, the perturbation as a result of rotor saturation can be quantified, as seen in the second term on the right-hand side of eq. (105). These dynamics can subsequently be employed to assess the stability of the quadrotor under the influence of actuator effects.

What should be noted is the fact that saturation of the command can have a pronounced effect on the stability, given the actuator dynamics of eq. (65). This is because saturation of the rotor speed commands leads to lower $\dot{\omega}$ than needed to achieve the desired output dynamics.

V. Stability Analysis

In the previous section the effect of the hedged virtual control signal on the external dynamics was derived. This has shown that the occurrence of actuator saturation adds a perturbation to the controlled system, leading to undesirable discrepancies between the actual error dynamics and its the expected behaviour.

Theoretically, this knowledge could be used to predict the hedged virtual control signal and design trajectories that prevent degradation of tracking performance. Methods such as pseudo-control hedging (PCH) can be used to scale down the reference trajectory to such an extent that the quadrotor can remain stable along its trajectory despite rotor limitations [23] [24]. Such algorithms have been successfully applied to aircraft [25][26] and quadrotor UAVs [27][11] previously. Pseudo-control hedging methods could be combined with yaw-independent trajectory guidance methods for quadrotor UAVs, such as those suggested by Hehn[16] and Mueller [17][18].

However, a combination of two characteristics of the fault-tolerant reduced attitude control makes this prohibitively complex. As can be seen in fig. 6, the attainable moment set of the quadrotor is asymmetric and varies strongly with the required thrust. Additionally, the quadrotor's continuous rotation complicates the generation of feasible, constrained trajectories, as the orientation of the body-frame with respect to the inertial frame varies constantly. Since the performance constraints are defined in the body frame, whereas trajectory guidance occurs in the inertial frame, feasible trajectory planning is exceedingly complex.

Therefore, the most feasible option to prevent the occurrence of loss-of-control is to use this knowledge to provide early warning of impending loss of control. As such, the trajectory can be aborted prior to failure.

A. Direct Method for Stability Analysis

Before loss-of-control can be predicted, the stability of the quadrotor under these saturation perturbations has to be defined. Wang [9] provides a potent method for the assessment of the stability of the external states of an INDI controller using Lyapunov's direct method [9]. First, it is necessary to define a Lyapunov equation for the INDI controlled quadrotor. This equation, $V(\boldsymbol{\zeta})$, is initially defined for the unperturbed system that does not suffer from actuator saturation.

Given a Lyapunov function $V(\boldsymbol{\zeta})$ that is positive for all $\boldsymbol{\zeta} \neq \mathbf{0}$, the system is stable at its origin $\boldsymbol{\zeta} = \mathbf{0}$ if the derivative of the Lyapunov function $\dot{V}(\boldsymbol{\zeta}) < 0$ for all $\boldsymbol{\zeta} \neq \mathbf{0}$. So to prove the stability around the origin, the following conditions need to hold for the given Lyapunov function $V(\boldsymbol{\zeta})$:[19] [9]

$$\begin{aligned} &\textbf{(1)} \quad V(\boldsymbol{\zeta}) > 0, \quad \text{for all } \boldsymbol{\zeta} \neq \mathbf{0} \\ &\textbf{(2)} \quad \dot{V}(\boldsymbol{\zeta}) < 0, \quad \text{for all } \boldsymbol{\zeta} \neq \mathbf{0} \end{aligned} \tag{106}$$

First, a candidate Lyapunov function is to be defined for the external dynamics in eq. (105) For a state vector such as $\boldsymbol{\zeta}$, a Lyapunov function in the form of eq. (107) can be defined, with the symmetric matrix $\boldsymbol{P}$ being a crucial part of the stability proof as explained subsequently [19].

$$\boldsymbol{V}(\boldsymbol{\zeta}) = \boldsymbol{\zeta}^T \boldsymbol{P} \boldsymbol{\zeta} \tag{107}$$

The matrix $\boldsymbol{P}$ is the solution eq. (108) and is strictly positive and symmetric, such that $\boldsymbol{P} > \mathbf{0}$ and $\boldsymbol{P}^T = \boldsymbol{P}$ [9][19].

$$\boldsymbol{P}(\boldsymbol{A}_a - \boldsymbol{B}_a \boldsymbol{K}_a) + (\boldsymbol{A}_a - \boldsymbol{B}_a \boldsymbol{K}_a)^T \boldsymbol{P} = -\boldsymbol{I} \tag{108}$$

The existence of a positive, symmetric matrix $\boldsymbol{P}$ that is a solution to eq. (108) proves that the unperturbed system is stable at its origin $\boldsymbol{\zeta} = \mathbf{0}$ [9][19].

If $\boldsymbol{P}$ is positive, the first condition for $V(\boldsymbol{\zeta})$ in eq. (106) is met, as a positive matrix $\boldsymbol{P}$ ensures that $V(\boldsymbol{\zeta})$ is positive for all $\boldsymbol{\zeta} \neq \mathbf{0}$ [19]. It can be shown that for all $\boldsymbol{\zeta}$, $V(\boldsymbol{\zeta})$ is bounded by two positive definite functions given in eq. (109), in which λ_{min} and λ_{max} signify the minimum and maximum eigenvalues of $\boldsymbol{P}$, respectively. Since $\boldsymbol{P}$ is strictly positive, these eigenvalues will also be positive, ensuring that $V(\boldsymbol{\zeta})$ is positive semi-definite for all $\boldsymbol{\xi}$ and thus a valid Lyapunov function [9] [19].

$$0 \leq \lambda_{min} ||\boldsymbol{\zeta}||^2 \leq V(\boldsymbol{\zeta}) \leq \lambda_{max} ||\boldsymbol{\zeta}||^2 \tag{109}$$

The proof of the second condition in eq. (106) requires the derivative of the Lyapunov function, $\dot{V}(\boldsymbol{\zeta})$. First, the stability of the system without the effect of rotor saturation is investigated. The dynamics of the unperturbed system can be seen below in eq. (110):

$$\dot{\boldsymbol{\zeta}} = (\boldsymbol{A_a} - \boldsymbol{B_a} \boldsymbol{K_a})\, \boldsymbol{\zeta} \tag{110}$$

Given these unperturbed state dynamics in the absence of rotor saturation, the derivative of the Lyapunov function in eq. (107) is derived in eq. (111) [9][19] .

$$\begin{aligned} \dot{\boldsymbol{V}}(\boldsymbol{\zeta}) &= \frac{\partial V(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}} \dot{\boldsymbol{\zeta}} \\ &= \boldsymbol{\zeta^T} \boldsymbol{P} \dot{\boldsymbol{\zeta}} + \dot{\boldsymbol{\zeta}}^{\boldsymbol{T}} \boldsymbol{P} \boldsymbol{\zeta} \\ &= \boldsymbol{\zeta}^T \boldsymbol{P}(\boldsymbol{A}_a - \boldsymbol{B}_a \boldsymbol{K}_a)\boldsymbol{\zeta} + \boldsymbol{\zeta}^T (\boldsymbol{A}_a - \boldsymbol{B}_a \boldsymbol{K}_a)^T \boldsymbol{P} \boldsymbol{\zeta} \\ &= \boldsymbol{\zeta}^T \left(\boldsymbol{P}(\boldsymbol{A}_a - \boldsymbol{B}_a \boldsymbol{K}_a) + (\boldsymbol{A}_a - \boldsymbol{B}_a \boldsymbol{K}_a)^T \boldsymbol{P} \right) \boldsymbol{\zeta} \\ &= -\boldsymbol{\zeta}^T \boldsymbol{\zeta} \\ &= -||\boldsymbol{\zeta}||^2 \end{aligned} \tag{111}$$

Due to the definition of the matrix $\boldsymbol{P}$ in eq. (108), the unperturbed Lyapunov derivative reduces to $\dot{\boldsymbol{V}}(\boldsymbol{\zeta}) = -||\boldsymbol{\zeta}||^2$. As a consequence, it is proven that $\dot{V}(\boldsymbol{\zeta}) < 0$ for all $\boldsymbol{\zeta} \neq \mathbf{0}$. Therefore, if the matrix $\boldsymbol{P}$ from eq. (108) exists, the conditions from eq. (106) are met and the unperturbed system from eq. (110) is stable at its origin for all values of $\boldsymbol{\zeta}$ [9][19]. Now, the perturbation due to actuator saturation can be introduced to the external dynamics to assess its effect on the system stability. The external dynamics from eq. (105) are simplified by assuming that $\dot{\boldsymbol{\delta}}(z, \Delta t)$ is negligble, which leads to:

$$\dot{\boldsymbol{\zeta}} = (\boldsymbol{A_a} - \boldsymbol{B_a}\boldsymbol{K_a})\,\boldsymbol{\zeta} - \boldsymbol{B_a}\boldsymbol{\nu_h} \tag{112}$$

Given these perturbed external dynamics, the perturbed derivative of the Lyapunov equation can be determined using the same method as in eq. (111). This leads to the perturbed derivative of the Lyapunov function below [9]:

$$\begin{aligned}\dot{V}(\boldsymbol{\zeta}) &= \frac{\partial V(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\dot{\boldsymbol{\zeta}} \\ &= \boldsymbol{\zeta}^T\left(\boldsymbol{P}(\boldsymbol{A}_a - \boldsymbol{B}_a\boldsymbol{K}_a) + (\boldsymbol{A}_a - \boldsymbol{B}_a\boldsymbol{K}_a)^T\boldsymbol{P}\right)\boldsymbol{\zeta} - \boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B}_a\boldsymbol{\nu_h} - \boldsymbol{P}(\boldsymbol{B}_a\boldsymbol{\nu_h})^T\boldsymbol{\zeta} \\ &= -||\boldsymbol{\zeta}||^2 - 2\boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B}_a\boldsymbol{\nu_h}\end{aligned} \tag{113}$$

The simplification in the final line of eq. (113) can be made due to the fact that the matrix $\boldsymbol{P}$ is symmetric. It can be seen in eq. (113) that the stability of the perturbed external states has an additional dependence on the hedged virtual control, $\boldsymbol{\nu_h}$. If large discrepancies between command and control realisation occur due to saturation, instability can occur if this perturbation causes $\dot{V}(\boldsymbol{\zeta})$ to become larger than 0. Whilst a positive $\dot{V}(\boldsymbol{\zeta})$ does not directly lead to a loss of control, it is indicative of tracking error growth.
The effect of the perturbation induced by rotor saturation can be seen in eq. (114).

$$\dot{V}(\boldsymbol{\zeta})\begin{cases}\leq 0, & \text{if: } ||\boldsymbol{\zeta}||^2 \geq -2\boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B}_a\boldsymbol{\nu_h} \\ > 0, & \text{if: } ||\boldsymbol{\zeta}||^2 < -2\boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B}_a\boldsymbol{\nu_h}\end{cases} \tag{114}$$

This stability condition shows that given the magnitude of $\boldsymbol{\zeta}$, there are values for $\boldsymbol{\nu_h}$ that can cause $\dot{V}(\boldsymbol{\xi})$ to become positive [9]. In such a case, the external states are temporarily unstable, leading to error growth until the stability condition in eq. (114) is met [9][19]. In physical terms, this means that insufficient control input $\boldsymbol{u_a}$ is being generated to stabilize the system around $\boldsymbol{\zeta} = \boldsymbol{0}$ for the desired trajectory. Because this perturbation is caused by rotors that saturate at their speed limits, it is unlikely that the quadrotor will be able to generate the required control realisation upon continuation of this trajectory. Therefore, if the perturbations lead to instability as defined by eq. (114), it is likely that continuation of the trajectory will lead to error growth and ultimately loss of control. It should be noted that this analysis only accounts for the effect of actuator saturation and does not include the effect of the ignored perturbation terms contained in $\boldsymbol{\delta}(\boldsymbol{\Delta z}, \Delta t)$, which can provide an additional destabilising effect.

B. Loss of Control Prediction

The destabilising effect of actuators has now been quantified using Lyapunov's direct method. However, whilst $\dot{V}(\boldsymbol{\zeta})$ can be used to identify the occurrence of instability, its predictive quality is limited. Its value does not intuitively quantify the margin between stability and instability.

To quantify the margin between stability and instability the stability coefficient, C_s, is derived. This coefficient is a ratio between the two terms in eq. (114) and is used to quantify the destabilising effect of rotor saturation in an intuitive manner. This stability coefficient is first determined by rewriting the stability condition in eq. (114), leading to eq. (115). To prevent numerical difficulties due to divisions by zero, the numerator and the denominator are filtered using a low-pass filter.

$$C_s = \frac{(-2\boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B_a}\boldsymbol{\nu_h})_f}{\|\boldsymbol{\zeta}\|_f^2} \tag{115}$$

This stability coefficient provides information about the sign of the Lyapunov derivative $\dot{V}$, while additionally providing an indication of the ratio between the stabilising and destabilising terms in eq. (114). C_s can be interpreted as a dimensionless coefficient that can be used to derive a filtered estimate of $\dot{V}(\boldsymbol{\zeta})$. This can be seen by rewriting eq. (113) and eq. (115) into:

$$\dot{V}(\boldsymbol{\zeta}) \approx (C_s - 1)\|\boldsymbol{\zeta}\|^2 \tag{116}$$

If $C_s = 0$, no perturbation occurs and the system is equal to the unperturbed system and therefore stable. When $0 < C_s < 1$, the system is theoretically stable around trajectory at that specific instant, as $\dot{V}(\boldsymbol{\zeta}) < 0$. A nonzero C_s does however indicate that the perturbation due to rotor saturation is affecting the tracking performance and stability, whilst its value provides an indication of the relative severity of the destabilising effect of actuator saturation. As such, it can be used to predict the onset of a loss of tracking performance and provide early warning of degraded control performance.

When $C_s = 1$ the two terms of eq. (114) are equal, which means that $\dot{V}(\boldsymbol{\zeta}) = 0$ and the system is marginally stable. At this point, the quadrotor is hypothetically vulnerable to instability if the rotor saturation continues to occur. At the moment that $C_s > 1$, the Lyapunov derivative $\dot{V} > 0$ and the system is unstable at that point in time for as long as the perturbation remains too large, leading to error growth and possibly a loss of control. It should be noted that a short instance of $C_s > 1$ does not directly imply loss of control is occurring, however, the temporary instability can lead to error growth, which requires larger virtual controls to compensate. If the actuators are already at their physical limit, it is likely that the actuators are unable to provide the required stabilising control action.

With eq. (115), a quantification of the destabilising effect of actuator saturation is available. However, this full state stability analysis does not provide an indication of the stability of the individual outputs. It is, for example, theoretically possible that the attitude is still stable when altitude control is lost.

The state transformation $\boldsymbol{\mathcal{T}_a(x_a)}$ transforms the system into three canonical form third order integrator chains for each of the outputs y_i. Although these systems are theoretically uncoupled [19], coupling between the systems exists due to the restricted AVCS. However, through the computation of $\boldsymbol{\nu_h}$, the AVCS limitations have been incorporated into the error dynamics. Therefore, the system in eq. (105) can be decoupled into three separate systems for each of the outputs as seen in eq. (117) [19].

$$\boldsymbol{\zeta} = \begin{bmatrix} \zeta_{r_1} \\ \zeta_{r_2} \\ \zeta_{r_3} \end{bmatrix} \tag{117}$$

The state vectors of the reduced state vectors can be seen below in eq. (118):

$$\zeta_{r_1} = \begin{bmatrix} Z - Z_{ref} \\ V_z - V_{z_{ref}} \\ a_z \end{bmatrix}, \zeta_{r_2} = \begin{bmatrix} h_1 - n_x^B \\ \dot{h}_1 \\ \ddot{h}_1 \end{bmatrix}, \quad \zeta_{r_3} = \begin{bmatrix} h_2 - n_y^B \\ \dot{h}_2 \\ \ddot{h}_2 \end{bmatrix} \tag{118}$$

The external dynamics of eq. (105) can now be decoupled into the three separate output integrator chains, as seen below in eq. (119)

$$\begin{aligned} \dot{\zeta}_{r_1} &= (\boldsymbol{A_r} - \boldsymbol{B_r K_z})\, \zeta_z - \boldsymbol{B_r} \nu_{h_1} \\ \dot{\zeta}_{r_2} &= (\boldsymbol{A_r} - \boldsymbol{B_r K_x})\, \zeta_x - \boldsymbol{B_r} \nu_{h_2} \\ \dot{\zeta}_{r_2} &= \left(\boldsymbol{A_r} - \boldsymbol{B_r K_y}\right) \zeta_x - \boldsymbol{B_r} \nu_{h_3} \end{aligned} \tag{119}$$

In eq. (119), the state and input matrices, $\boldsymbol{A_r}$ and $\boldsymbol{B_r}$, are given by eq. (120). The gain matrix $\boldsymbol{K_{r_i}}$ can be derived directly from the matrix $\boldsymbol{K_a}$ in eq. (86).

$$\boldsymbol{A_r} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \boldsymbol{B_r} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{120}$$

Using the same stability analysis methodology, the stability of the separated systems can be derived using eq. (121). First a Lyapunov function V_{r_i} is defined for each integrator chain for output y_i. This Lyapunov function is given below in eq. (121), with the symmetric matrix $\boldsymbol{P}_{r_i}$ the solution of the Lyapunov equation in eq. (122)

$$\boldsymbol{V}_{r_i}(\zeta_{r_i}) = \zeta_{r_i}^T \boldsymbol{P}_{r_i} \zeta_{r_i} \tag{121}$$

$$\boldsymbol{P}_{r_i}(\boldsymbol{A}_r - \boldsymbol{B}_r \boldsymbol{K}_{r_i}) + (\boldsymbol{A}_r - \boldsymbol{B}_r \boldsymbol{K}_{r_i})^T \boldsymbol{P}_{r_i} = -\boldsymbol{I} \tag{122}$$

The existence of the matrix $\boldsymbol{P}_{r_i} = \boldsymbol{P}_{r_i}^T > 0$ proves the stability of the unperturbed dynamics $\dot{\boldsymbol{\zeta}}_{r_i}$, as established earlier in section V.A. Given the Lyapunov function V_{r_i} and the matrix $\boldsymbol{P}_{\boldsymbol{r_1}}$, the stability of the perturbed system is provided by eq. (123):

$$\begin{aligned} \dot{V}_{r_i}(\zeta)_{r_i} &= \zeta_{r_i}^T \left(\boldsymbol{P}_{r_i}(\boldsymbol{A}_r - \boldsymbol{B}_r \boldsymbol{K}_{r_i}) - (\boldsymbol{A}_r - \boldsymbol{B}_r \boldsymbol{K}_{r_i})^T \boldsymbol{P}_{r_i} \right) \zeta_{r_i} - 2\zeta_{r_i}^T \boldsymbol{P}_{\boldsymbol{r_i}} \boldsymbol{B}_{\boldsymbol{r}} \boldsymbol{\nu}_{\boldsymbol{h_i}} \\ &= -||\zeta_{r_i}||^2 - 2\zeta_{r_i}^T \boldsymbol{P}_{\boldsymbol{r_i}} \boldsymbol{B}_{\boldsymbol{r}} \boldsymbol{\nu}_{\boldsymbol{h_i}} \end{aligned} \tag{123}$$

Rewriting the Lyapunov derivative in eq. (123), a stability coefficient can be derived for each of the individual outputs. This can be seen in eq. (124)

$$C_{s_i} = \frac{(-2\boldsymbol{P}_{\boldsymbol{r_i}} \boldsymbol{B}_{\boldsymbol{r}} \boldsymbol{\nu}_{\boldsymbol{h_i}})_f}{||\boldsymbol{\zeta}_{\boldsymbol{r_i}}||_f^2} \tag{124}$$

With the stability coefficient for each of the three outputs, C_{s_i}, it can be more accurately assessed whether the system is stable. Similarly as for the full system, $C_{s_i} > 1$ indicates that the perturbation due to actuator saturation leads to instability. Because the stability assessment is now separated for each of the outputs, it is easier to predict the failure mode of the quadrotor.

VI. Results

To validate the controller design and loss of control prediction methodology, simulations are conducted using a 6-degree-of-freedom model of the Parrot Bebop 2 quadrotor UAV, with the prescribed dynamics of section II . [5] A high-fidelity aerodynamic model of a Bebop 2 quadrotor experiencing single rotor failure is used that models the complex aerodynamic forces and moments, including rotor interference and high-speed effects[15] with validity up to airspeeds of 16 m/s. The model is developed in a Matlab/Simulink environment, the parameters that were used are provided in table 2.[6]

parameter	**m**	$\mathbf{I}_{xx}$	$\mathbf{I}_{yy}$	$\mathbf{I}_{zz}$	**b**	**l**
value	0.51 kg	0.0019 kg/m^2	0.0019 kg/m^2	0.0033 kg/m^2	0.115 m	0.875m
parameter	$\boldsymbol{\omega_{min}}$	$\boldsymbol{\omega_{max}}$	$\boldsymbol{\omega_B}$	$\boldsymbol{\kappa}$	$\boldsymbol{\sigma}$	$\mathbf{I}_p$
value	300 rad/s	1200 rad/s	50 rad/s	$1.9{\cdot}10^{-6}$	$1.9{\cdot}10^{-8}$	$7.1{\cdot}10^{-6}$

Table 2 Parameter values of the Parrot Bebop 2 quadrotor UAV[6][7].

To assess the validity of the loss-of-control prediction methodology, simulations were conducted to induce loss of control. Sun et al [5][6] have previously conducted high-speed flight tests to identify the maximum flight speed after single rotor failure. In these experiments, actuator saturation and actuator delay were identified as the most likely cause of failure [5][6].

To replicate this loss of control due to overspeed, a continuously accelerating trajectory in the inertial x-direction is tracked at an altitude of 2 m until a crash occurs, with the reference signal given in eq. (125). The body frame primary axis vector is set to $\boldsymbol{n^B} = [\ 0.2 \quad 0.2 \quad -0.96\ ]^T$ to provide a stabilising precession moment [8], as was shown in detail in section IV.D.

$$\boldsymbol{x}_{des} = \left[\ \tfrac{0.3t^2}{2} \quad 0 \quad -2\ \right]^T \tag{125}$$

In fig. 9, the position and velocity tracking performance is depicted between 15 and 45 seconds along this trajectory. In this figure, the figures in the left column depict the position tracking, whereas the right column depicts the velocity tracking. The reduced attitude tracking is represented in fig. 10, in which the left column depicts the $x-$, $y-$ and $z-$components of the inertial reduced attitude, $\boldsymbol{n^I}$, whereas the right column depicts the tracking error of the components of the reduced attitude in the body frame, $\boldsymbol{h} - \boldsymbol{n^B}$.

In fig. 9 and fig. 10, three different moments of interest are visible, which indicate various stages of the occurrence of loss of control. Firstly, the first dotted line at t=35 s marks the instant whereby all of the depicted measurements start to diverge. Whilst slight position and attitude tracking error was already visible, all of the tracking errors diverge simultaneously from this point onward, indicating instability. Subsequently, the red vertical line marks the second moment of interest, when the quadrotor altitude exceeds the minimum altitude of 2 m at t=39.6 s at a velocity of 11 m/s, which in reality would constitute impact with the ground. At this point, it is also visible that the position tracking in the x- and y- direction has similarly degraded and position control is lost. Finally, ignoring this virtual altitude limit, the dotted line at t=44 s marks the occurrence of an upset at approximately 12 m/s, when the inertial primary axis component $n_z^I > 0$ and the quadrotor is oriented upside down. At this point, the quadrotor becomes fully uncontrollable due to the existence of a singularity in the control effectiveness matrix and would require an upset recovery controller to recover [21]. Given the high velocity, the chances of recovery would depend strongly on the available altitude and lateral clearance [21]. Therefore this instant could be classified as a loss of control.

The inertial frame reduced attitude tracking in fig. 10, reveals a strong increase in the horizontal components at t=35 s, likely a result of the inability of the quadrotor to compensate for the high-speed aerodynamic forces. With increasing flight speed, these disturbance forces grow, reducing the forward acceleration and introducing a lateral force [5][15]. This subsequently leads to larger acceleration commands from the outer loop, leading to greater tilt commands for $\boldsymbol{n^I}$. As visible in fig. 10 the quadrotor is unable to realise the attitude commands, as seen by the large oscillations of $\boldsymbol{h}$. This can similarly be explained by increasing high speed aerodynamic disturbance moments affecting the attitude control.

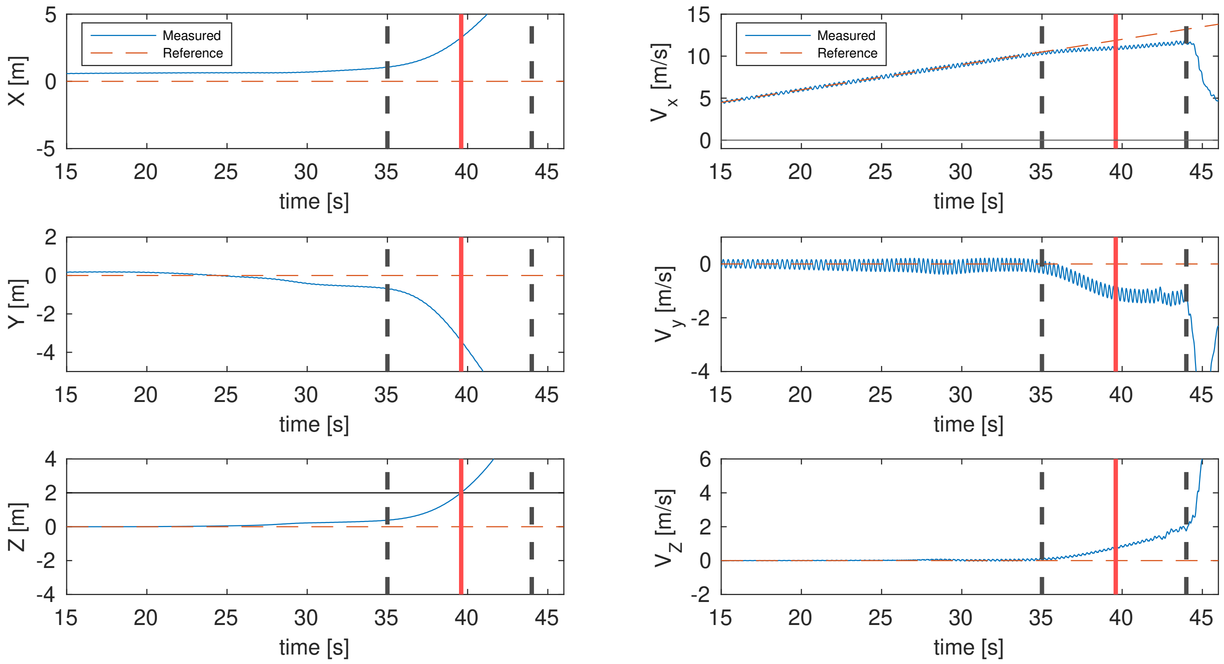

Fig. 9 The tracking performance for position and velocity over the accelerating trajectory. The left column depicts the position tracking errors, whereas the right column depicts the inertial velocity.

Upon such large aerodynamic forces and moments, greater thrust and control moments are required, increasing the likelihood of saturation. It is therefore expected that the sudden degradation of control performance at high speed is the result of actuator saturation. To confirm this hypothesis, the commanded and measured rotor speeds are inspected in fig. 11. For readability, the rotor speeds are depicted for at two different intervals, whereby the first interval is given by $t = [15, 20]$, while the second interval is given by $t = [30, 35]$. Comparing the two intervals for each individual rotor, the frequency and duration of saturation is clearly increased in the second interval. Rotor 3 even remains constant at ω_{max} for the entirety of the second interval.

In fig. 12 the virtual control, $\boldsymbol{v_a}$, is compared to the control realisation, $\boldsymbol{v_r}$, for the same intervals. This clearly depicts the effect of this rotor saturation on the control realisation. Whilst the command and realisation are very similar in the first interval, the second interval shows a large discrepancy between these variables. This indicates that the actuators cannot realise the required virtual control, leading to the observed tracking error divergence. For v_2 in the second interval, it can additionally be observed that the control realisation exhibits an erratic pattern compared to its

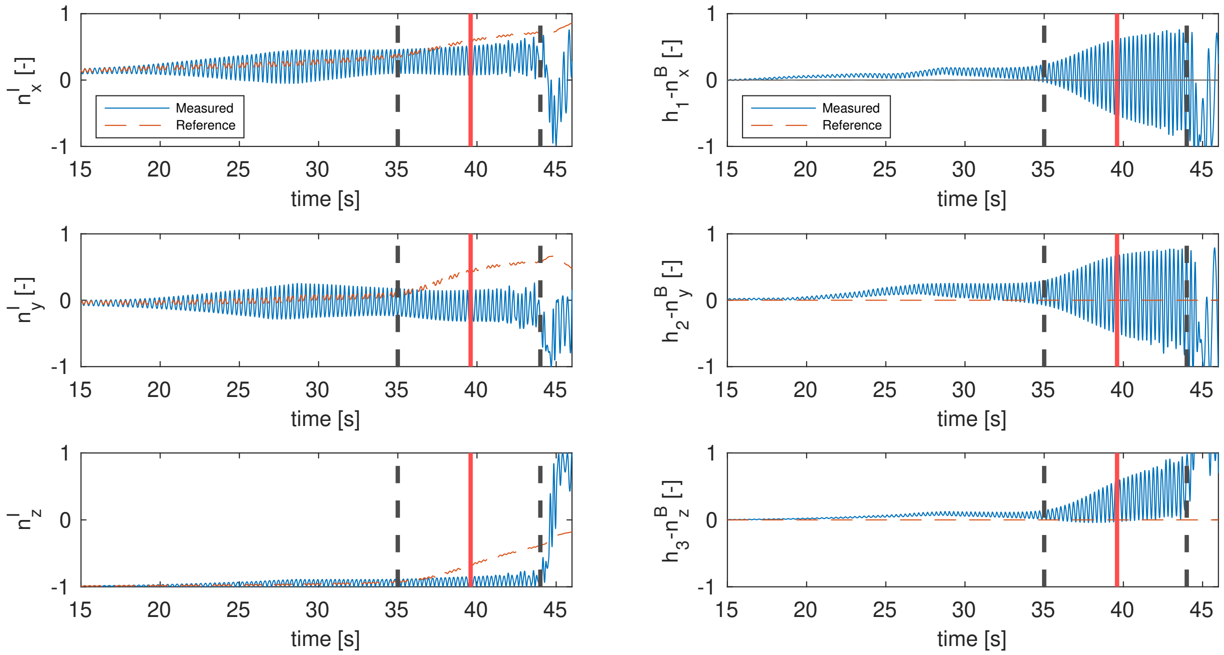

Fig. 10 The tracking performance of the attitude, represented by the inertial frame primary axis in the left column and the body frame primary axis tracking error in the right column.

command, demonstrating that actuator saturation can lead to unpredictable control realisations. Finally, it is observable that the virtual control starts to increase drastically as a result of the growing tracking error which the actuators are unable to compensate, leading to further growth of the tracking error.

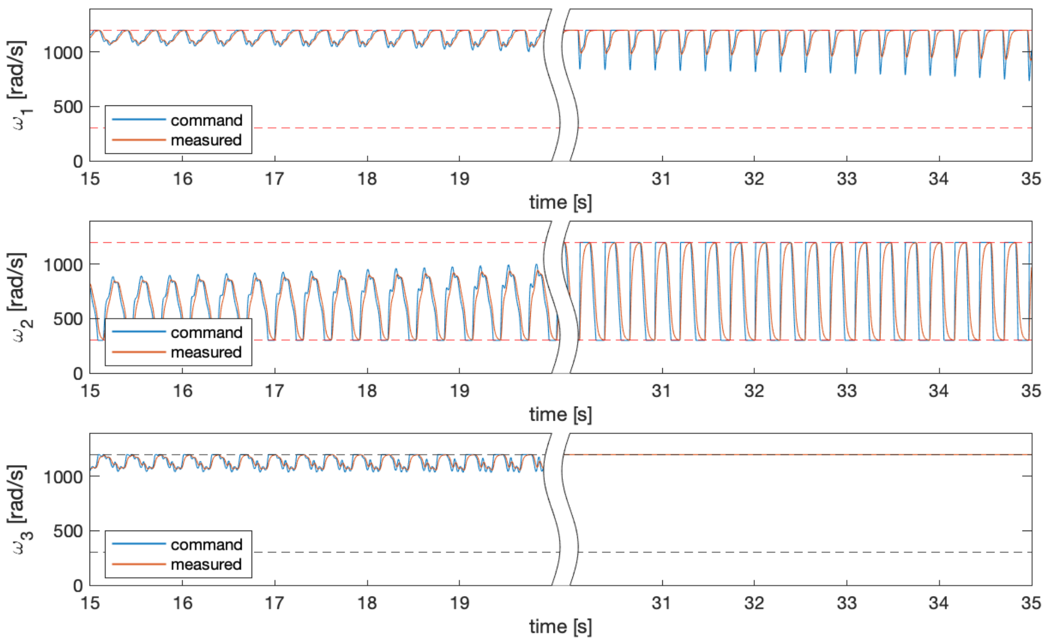

Fig. 11 The commanded and actual rotor speeds at two different intervals along the trajectory.

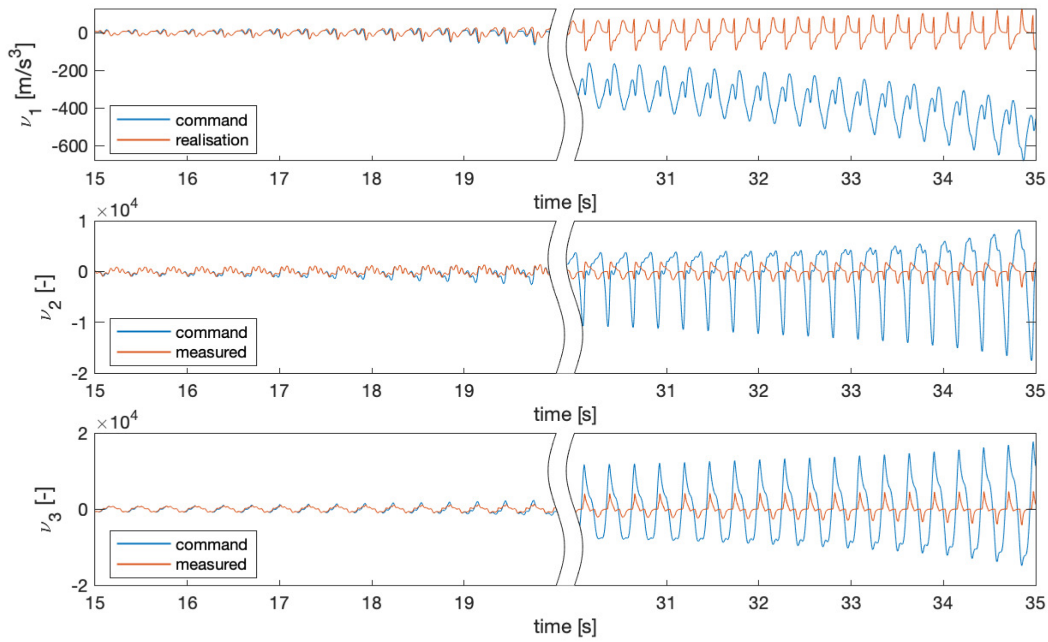

Fig. 12 The commanded virtual control ν_a compared to the control realisation ν_r, for y_1, y_2 and (y_3.

Given the clearly observable discrepancy between command and control realisation, the occurrence of actuator saturation influences the tracking performance of the quadrotor. It is observed that the trajectory tracking performance degrades strongly as soon as this discrepancy occurs. Therefore, the destabilising effect of actuator saturation should be visible in the stability analysis of the quadrotor around its trajectory. For that purpose, the stability coefficients C_{s_i} are determined using eq. (124) and presented in the left column of fig. 13 for each of the outputs, $\boldsymbol{y} = \begin{bmatrix} Z - Z_{ref} & h_1 - n_x^B & h_2 - n_y^B \end{bmatrix}^T$. The stability coefficients are low-pass filtered with a bandwidth of 16 rad/s, corresponding to the average rotational rate to avoid high outliers due to zero crossings in the denominator of C_{s_i}.

It is visible that the stability coefficients show a gradual increase for all of the outputs starting at approximately t = 15 s, when the frequency and duration of rotor saturation starts to increase. This indicates that actuator saturation is affecting the tracking performance of the quadrotor, even though the rotor saturation alone does not completely destabilise the system. At t=25 s, it is observable that the coefficients exceed 1, which indicates that effect of actuator saturation alone is sufficiently strong to cause divergence of the tracking error. This time roughly corresponds to the gradual increase in tracking error observed for the position and attitude control. From this point onwards, the stability coefficients remain above one, which shows that error divergence is occurring as a result of actuator saturation, corresponding with the observations made in the previous section. This demonstrates the predictive value of the stability coefficient C_s, which not only indicates the moment of error divergence, but also provides an indication of degrading stability prior to its occurrence.

As an additional verification of the accuracy of the prediction, the correlation between the virtual control command $\boldsymbol{\nu_a}$ and the realisation $\boldsymbol{\nu_r}$ is investigated. If the correlation between these signals is strong, the discrepancy between command and realisation is limited. For a high correlation between this signals, it is likely that the system can still compensate for momentary saturation. Once the correlation degrades, it is likely that saturation has a longer lasting effect and the required virtual control can no longer be realised by the actuator. The correlation between the virtual control command $\boldsymbol{\nu_a}$ and the realisation $\boldsymbol{\nu_r}$ is quantified using eq. (126). [28] A moving window of n_s = 1000 samples is used, which is roughly equal to 2 s for the INDI control design of section IV.B.

$$\mathcal{R}_{ar} = \frac{\sum_{i=1}^{n_s}(\boldsymbol{\nu_a}(\boldsymbol{i}) - \overline{\boldsymbol{\nu}}_{\boldsymbol{a}}) \sum_{i=1}^{n_s}(\boldsymbol{\nu_r}(\boldsymbol{i}) - \overline{\boldsymbol{\nu}}_{\boldsymbol{r}})}{\sqrt{\sum_{i=1}^{n_s}(\boldsymbol{\nu_a}(\boldsymbol{i}) - \overline{\boldsymbol{\nu}}_{\boldsymbol{a}})^2}\sqrt{\sum_{i=1}^{n_s}(\boldsymbol{\nu_r}(\boldsymbol{i}) - \overline{\boldsymbol{\nu}}_{\boldsymbol{r}})^2}} \tag{126}$$

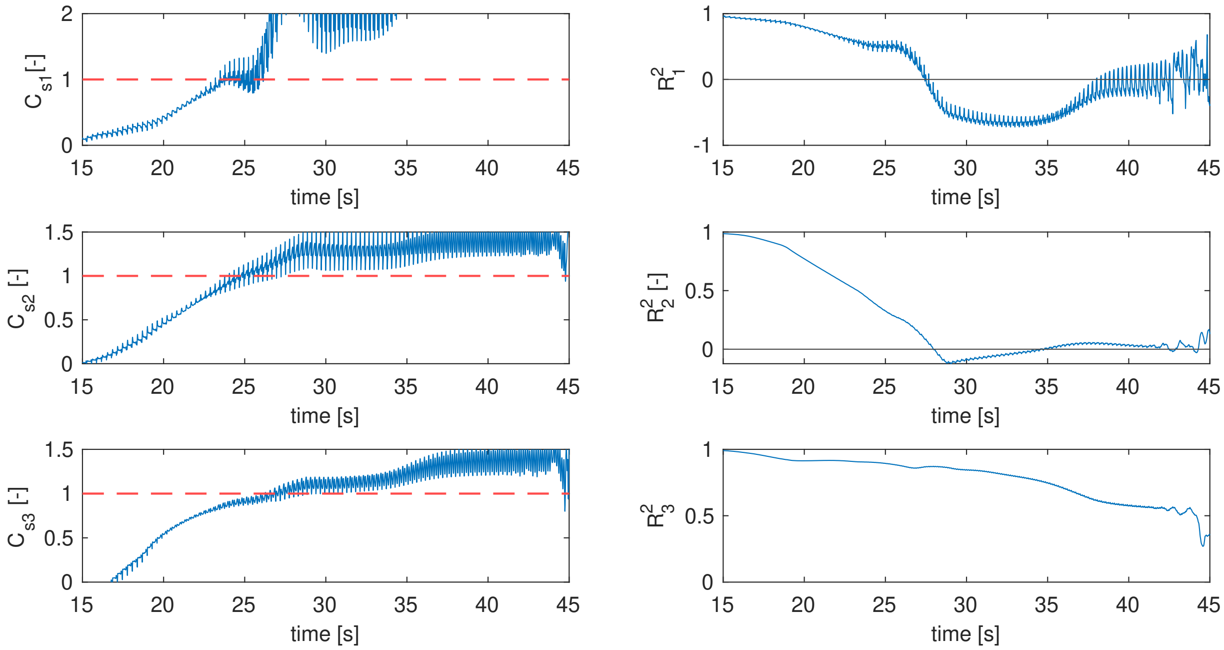

Fig. 13 Estimated stability parameters of the quadrotor around its trajectory. The left column shows the stability coefficients for each of the three inner-loop outputs. The red dotted line indicates the upper limit for stability at $C_{s_i} = 1$. The right column depicts the correlation of ν_a and ν_r for each of the outputs.

The right column of fig. 13 depicts the correlation $\mathcal{R}_{ar_i}$ for each of the different output channels. This shows that for t = [15,20], the correlation between the command and the realisation strongly drops for for y_1 and y_2, whilst y_3 shows a smaller but noticeable decrease. Similarly to the pattern observed for C_s, this coincides with the time when the tracking performance noticeably degrades. As can also be observed in fig. 12 it coincides with the moment when the control realisation no longer resembles the command. In this case it can be assumed that the quadrotor is no longer capable of providing the control required to stabilise itself along its desired trajectory. Neither can it compensate for the temporary instability due to rotor saturation. Therefore, a combination of $C_{s_i} > 1$ and a low correlation $\mathcal{R}_{ar_i}$ is indicative of an impending loss of control scenario if the trajectory is continued. This correlation signal can therefore be used as an onboard verification when C_s predicts the occurrence of saturation induced instability. This is particularly useful in case outliers occur for C_s due to either improper filter bandwidth selection or division by a very low denominator in eq. (124).

As a final verification, the measured internal state, η is presented in fig. 14. This shows that the internal state remained stable over the course of the trajectory, up until the occurrence of the upset. The instability of the internal dynamics could potentially explain the sudden occurrence of the attitude upset. Therefore it is recommended to perform more detailed research into this topic to add to the safety of quadrotors after rotor failure.

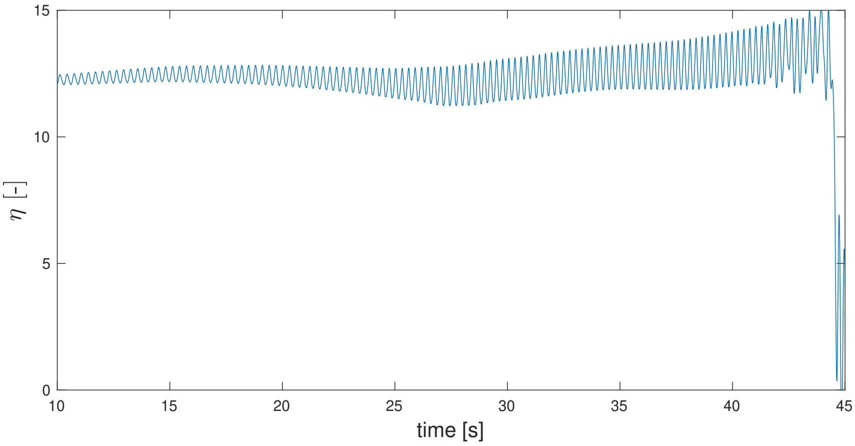

Fig. 14 The measurement of the internal state η over the trajectory.

VII. Conclusion

Over the course of this research the impact of actuator effects on the stability of a quadrotor after single rotor failure has been analysed. Whilst the applied INDI controller has been proven to be theoretically stable under the assumption of affine control, it has been shown that the actuator dynamics and rotor saturation affect the tracking performance and stability.

The system dynamics have been adapted to include the delay due to the actuator dynamics, leading up to the design of an augmented INDI controller that accounts for the actuator lag. The effects of actuator saturation have been analysed and quantified using the hedged virtual control, a variable that signifies the discrepancy between the commanded and realised virtual control. Through the use of these methods, the impact of actuator effects on the tracking performance and stability of a damaged quadrotor can be quantified in real-time, given its trajectory.

By application of Lyapunov's direct method, a stability metric has been defined that provides a measure of the stability of the quadrotor around its trajectory, based on the augmented system dynamics and the hedged virtual control signal. These stability coefficients provide an indication of the destabilising effect of actuator saturation in an intuitive manner, which can be used to prevent error growth and instability prior to its occurrence. Whilst a stability coefficient short occurrence of $C_s > 1$ in isolation does not imply immediate loss of control, continued intervals with a stability coefficient$C_s > 1$ do indicate that the actuators are unable to provide the control realisation required to track the trajectory. This will lead to error growth and can ultimately lead to a loss of control situation. This characteristic can be used to predict the degradation of the control performance and provide early warning of impending loss of control.

In simulation, the stability metric was shown to be capable of predicting loss-of-control prior to its occurrence. Prior to instability, the stability coefficient similarly provides an intuitive indication of the margin between stability and instability. This property is particularly useful for signalling the degradation of control performance.

To verify the validity of loss of control warnings based on the stability coefficient, the correlation coefficient between the commanded virtual control and the control realisation can be used. A loss of control prediction should only be issued if this value is below a predetermined threshold.

VIII. Discussion

Whilst this research provides a promising theoretical method to predict loss-of-control for quadrotors experiencing single rotor failure, further work is required to validate this methodology.

First of all, the actuator dynamics are currently assumed to be modelled accurately by a first order lag model. Whilst this assumption has been made in prior research on quadrotors[11][12][7], real-life validation of the methodology is required to assess its accuracy. Additionally, more detailed research into rotor speed control would add to the fidelity of the loss-of-control prediction. Direct control over the rotational acceleration of the rotors could potentially offer improved performance.

Secondly, a constant rotor thrust coefficient is assumed in the prediction methodology. In reality, nonlinear effects such as blade flapping, rotor interference and the rotor angle of attack affect the thrust coefficient of the individual rotors

[1][15][14]. Whilst these effects were modelled in simulation, using a high-fidelity aerodynamic model [15], its effects are not incorporated in the stability analysis and controller design. Assessment of the effect of the rotor thrust coefficient variation on the stability of the quadrotor could potentially add accuracy to the prediction method.

Thirdly, the current methodology requires accurate knowledge of the quadrotor states. In earlier research with the Bebop 2 quadrotor, state-estimation using Optitrack motion capturing was used to provide an accurate estimation of the ground truth [5][6]. Analysis of the effect of measurement error was beyond scope for this research, so the validity of this loss-of-control prediction methodology under real-world conditions without accurate state knowledge is unknown. It is likely that accurate ground truth knowledge is required to provide accurate prediction of stability.

Additionally, it is recommended to further investigate the stability of the internal dynamics over a broader range of $\boldsymbol{\zeta}$. Whilst it can be shown that the internal dynamics largely remain stable as long as the external states are stabilised around the origin, the exact domain of attraction is unknown. Therefore, unexpected instability and loss-of-control may occur as a result of unexpectedly unstable internal dynamics.

Finally, it may be worthwhile to investigate utilising the stability analysis methodology to design trajectory planning methods or adaptive outer-loop controllers that can provide feasible trajectories for the INDI inner-loop controller. Whilst this was originally the goal of the research, this was deemed infeasible within the scope, due to the nonlinearity of the trajectory constraints.

Appendix: Internal Dynamics Derivation

The internal dynamics from eq. (127) can be rewritten in terms of the internal and external states to complete the state transformation $\boldsymbol{\mathcal{T}(x)}$.

$$\dot{\eta} = \dot{V}_z + \mu_1 \dot{p} + \mu_2 \dot{q} + \mu_3 \dot{r} \tag{127}$$

The first step in this process is to rewrite the dynamics of $\dot{V}_z$, $\dot{p}$, $\dot{q}$ and $\dot{r}$ in terms of state dependent terms only. For this purpose, the system in eq. (16) is rewritten and all of the input-dependent terms $\boldsymbol{G(x)u}$ are ignored, since the internal dynamics are proven to be independent from the input. Additionally, external forces and moments are taken as external, unmodelled disturbances and are therefore ignored. This method is similar to the method of Sun et al for the internal dynamics of the SRF case. [6]

$$\begin{aligned} \dot{V}_z &= g \\ \dot{p} &= A_x qr \\ \dot{q} &= A_y pr \\ \dot{r} &= A_z pq - \gamma r \end{aligned} \tag{128}$$

The second step is to rewrite the variables in the internal dynamics that are written in terms of the states *bmx* as variables in terms of the transformed states.

For $\dot{V}_Z$ this can be done quite simply, as seen in eq. (129)

$$\dot{V}_z = \xi_2 + V_{z_{ref}} \tag{129}$$

For $\dot{p}$, $\dot{q}$ and $\dot{r}$ this is more complex and requires several steps. First, p, q and r are defined partially as functions of the transformed states:

$$p = \frac{h_1 r + \dot{h}_2}{h_3} = \frac{(\xi_3 + n_x^B) r + \xi_6}{h_3} \tag{130}$$

$$q = \frac{h_2 r - \dot{h}_1}{h_3} = \frac{(\xi_5 + n_y^B) r - \xi_4}{h_3} \tag{131}$$

$$r = \frac{1}{\mu_3} (\eta - V_z - \mu_1 p - \mu_2 q) \tag{132}$$

Substituting eq. (134) and eq. (135) into eq. (132) and solving for r, leads to the following definition of r in terms of the state transformation $\boldsymbol{\mathcal{T}(x)}$ in eq. (133)

$$r = \frac{h_3\eta - h_3\xi_2 - \mu_1\xi_6 + \mu_2\xi_4}{\mu_1\left(\xi_3 + n_x^B\right) + \mu_2\left(\xi_5 + n_y^B\right) + \mu_3 h_3} = \frac{\alpha}{\beta} \tag{133}$$

Substituting the definition of r into eq. (134) and eq. (135), leads to the following definitions for p and q:

$$p = \frac{\alpha^2(\xi_3 + n_x^B) + \alpha\xi_6}{\alpha h_3} \tag{134}$$

$$q = \frac{\alpha^2(\xi_5 + n_y^B) - \alpha\xi_4}{\alpha h_3} \tag{135}$$

Rewriting eq. (127) using eq. (128) leads to the following reformulation of the internal dynamics in eq. (136):

$$\dot{\eta} = g - V_{z_{ref}} + \mu_1 A_x qr + \mu_2 A_y pr + \mu_3\left(A_z pq - \gamma r\right) \tag{136}$$

Substituting the definitions for p,q and r into the equations for $\dot{p}$, $\dot{q}$ and $\dot{r}$ in eq. (136) leads to:

$$\begin{aligned} \dot{\eta} = g - \frac{\mu_3\gamma\beta}{I_z\alpha} \quad &+ \frac{\mu_1 A_x\beta\left(\beta\left(\xi_5 + n_y^B\right) + \alpha\xi_4\right) + \mu_2 A_y\beta\left(\beta\left(\xi_3 + n_x^B\right) + \alpha\xi_6\right)}{h_3 a^2} \\ &+ \frac{\mu_3 A_z\left(\alpha\xi_6 + \beta\left(\xi_3 + n_x^B\right)\left(\beta\left(\xi_5 + n_y^B\right)\right) - \alpha\xi_4\right)}{h_3^2\alpha^2} \end{aligned} \tag{137}$$

References

[1] Mahony, R., Kumar, V., and Corke, P., "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE Robotics and Automation Magazine*, Vol. 19, No. 3, 2012, pp. 20–32. https://doi.org/10.1109/MRA.2012.2206474.

[2] Lanzon, A., Freddi, A., and Longhi, S., "Flight control of a quadrotor vehicle subsequent to a rotor failure," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 2, 2014, pp. 580–591. https://doi.org/10.2514/1.59869.

[3] Mueller, M. W., and D'Andrea, R., "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers," *Proceedings - IEEE International Conference on Robotics and Automation*, 2014. https://doi.org/10.1109/ICRA.2014.6906588.

[4] Mueller, M. W., and D'Andrea, R., "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles," *International Journal of Robotics Research*, Vol. 35, No. 8, 2016, pp. 873–889. https://doi.org/10.1177/0278364915596233.

[5] Sun, S., Sijbers, L., Wang, X., and De Visser, C., "High-Speed Flight of Quadrotor Despite Loss of Single Rotor," *IEEE Robotics and Automation Letters*, Vol. 3, No. 4, 2018, pp. 3201–3207. https://doi.org/10.1109/LRA.2018.2851028.

[6] Sun, S., Wang, X., Chu, Q., and de Visser, C., "Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors," *IEEE Transactions on Robotics*, Vol. 37, No. 1, 2020, pp. 116–130. https://doi.org/10.1109/TRO.2020.3010626.

[7] Lu, P., and Van Kampen, E. J., "Active fault-tolerant control for quadrotors subjected to a complete rotor failure," *IEEE International Conference on Intelligent Robots and Systems*, Vol. 2015-Decem, 2015, pp. 4698–4703. https://doi.org/10.1109/IROS.2015.7354046.

[8] Sun, S., Baert, M., Van Schijndel, B. S., and De Visser, C., "Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances," *Proceedings - IEEE International Conference on Robotics and Automation*, 2020. https://doi.org/10.1109/ICRA40945.2020.9197239.

[9] Wang, X., van Kampen, E. J., Chu, Q., and Lu, P., "Stability analysis for incremental nonlinear dynamic inversion control," *Journal of Guidance, Control, and Dynamics*, Vol. 42, No. 5, 2019, pp. 1116–1129. https://doi.org/10.2514/1.G003791.

[10] Raffler, T., Wang, J., and Holzapfel, F., "Path generation and control for unmanned multirotor vehicles using nonlinear dynamic inversion and pseudo control hedging," *IFAC Proceedings Volumes (IFAC-PapersOnline)*, Vol. 19, No. PART 1, 2013, pp. 194–199. https://doi.org/10.3182/20130902-5-de-2040.00132.

[11] Smeur, E. J., Chu, Q., and De Croon, G. C., "Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 3, 2016, pp. 450–461. https://doi.org/10.2514/1.G001490.

[12] Smeur, E., Höppener, D., and De Wagter, C., "Prioritized Control Allocation for Quadrotors Subject to Saturation," *IMAV 2017: International Micro Air Vehicle Conference and Competition 2017*, , No. September, 2017, pp. 37–43. URL http://www.imavs.org/papers/2017/51_imav2017_proceedings.pdf.

[13] Steffensen, R., Steinert, A., and Smeur, E. J. J., "Nonlinear Dynamic Inversion with Actuator Dynamics: An Incremental Control Perspective," *Journal of Guidance, Control, and Dynamics*, Vol. 46, No. 4, 2023, pp. 709–717. https://doi.org/10.2514/1.g007079.

[14] Sun, S., De Visser, C. C., and Chu, Q., "Quadrotor gray-box model identification from high-speed flight data," *Journal of Aircraft*, Vol. 56, No. 2, 2019, pp. 645–661. https://doi.org/10.2514/1.C035135.

[15] Sun, S., and de Visser, C., "Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime," *IEEE Robotics and Automation Letters*, Vol. 4, No. 4, 2019, pp. 3868–3875. https://doi.org/10.1109/lra.2019.2928758.

[16] Hehn, M., and Dandrea, R., "Real-Time Trajectory Generation for Quadrocopters," *IEEE Transactions on Robotics*, Vol. 31, No. 4, 2015, pp. 877–892. https://doi.org/10.1109/TRO.2015.2432611.

[17] Mueller, M. W., Hehn, M., and Dandrea, R., "A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation," *IEEE Transactions on Robotics*, Vol. 31, No. 6, 2015, pp. 1294–1310. https://doi.org/10.1109/TRO.2015.2479878.

[18] Mueller, M. W., and D'Andrea, R., "A model predictive controller for quadrocopter state interception," *2013 European Control Conference, ECC 2013*, 2013, pp. 1383–1389. https://doi.org/10.23919/ecc.2013.6669415.

[19] Khalil, H. K., *Nonlinear systems*, Prentice Hall, 2002.

[20] Yang, X., and Zhu, J. J., "Singular perturbation margin and generalised gain margin for nonlinear time-invariant systems," *International Journal of Control*, Vol. 89, No. 3, 2016, pp. 451–468. https://doi.org/10.1080/00207179.2015.1079738, URL https://doi.org/10.1080/00207179.2015.1079738.

[21] Sun, S., Baert, M., Van Schijndel, B. S., and De Visser, C., "Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances," *Proceedings - IEEE International Conference on Robotics and Automation*, 2020. https://doi.org/10.1109/ICRA40945.2020.9197239.

[22] Narasimhan, A., de Visser, C., and De Wagter, C., "Fault Tolerant Control of Multirotor UAV for Piloted Outdoor Flights," ????, pp. 1–30.

[23] Johnson, E., and Calise, A., "Pseudo-Control Hedging: A New Method For Adaptive Control," 2001.

[24] Zhang, J., and Holzapfel, F., "Saturation protection with pseudo control hedging: A control allocation perspective," *AIAA Scitech 2021 Forum*, , No. January, 2021, pp. 1–10. https://doi.org/10.2514/6.2021-0370.

[25] Matamoros, I., and de Visser, C. C., "Incremental nonlinear control allocation for a tailless aircraft with innovative control effectors," *AIAA Guidance, Navigation, and Control Conference, 2018*, , No. 210039, 2018, pp. 1–25. https://doi.org/10.2514/6.2018-1116.

[26] Lombaerts, T., Looye, G., Chu, P., and Mulder, J. A., "Pseudo control hedging and its application for safe flight envelope protection," *AIAA Guidance, Navigation, and Control Conference*, 2010. https://doi.org/10.2514/6.2010-8280.

[27] Zhang, J., and Holzapfel, F., "Saturation protection with pseudo control hedging: A control allocation perspective," *AIAA Scitech 2021 Forum*, , No. January, 2021, pp. 1–10. https://doi.org/10.2514/6.2021-0370.

[28] Klein, V., and Morelli, E. A., *Aircraft System Identification: Theory and Practice. American Institute of Aeronautics and Astronautics*, Vol. 13, 2006.

Part II

Closure

2
Conclusion

Following the research effort shown in the literature study and the scientific paper, the conclusions from the research effort are presented in this section. First, the subquestions are answered in Section 2.1, followed by the answer to the main research question and the concluding remarks. Finally, recommendations for future research are provided, based on the findings of this report.

2.1. Answers to the Sub-Questions

> **RQ1:** : How can a loss-of-control scenario be defined?
> - What are the existing definitions of a loss-of-control scenario for a quadrotor experiencing rotor failure?
> - Are these definitions sufficient, given observations in experiments and simulation?

The definition for a loss of control is not well-defined and often used alongside the definition of an upset. Sun defines an upset as a scenario whereby an uncommanded attitude deviation occurs, whereby the pitch or roll angle exceed $\frac{\pi}{2}$. This definition is well-defined and presents a scenario wherein altitude control is impossible for fixed-pitch quadrotors that cannot provide negative thrust. Both the windtunnel tests by Sun et al [3][4] and the simulations in the scientific paper, however, show that loss of position control may occur earlier than an attitude upset. It was observed that over an accelerating trajectory, tracking deficiencies in position tracking start to occur when insufficient control realisation can be generated due to actuator saturation. This tracking error can rapidly grow if not acted upon and subsequently lead to a crash.

It was demonstrated in the research paper that the stability under these actuator related perturbations can be quantified. This method is based on Lyapunov's direct method and can provide early warning of loss-of-control due to actuator saturation.

A loss-of-control situation can be classified as a situation whereby the actuators cannot provide the required control realisation to stabilise the quadrotor around its trajectory. A sudden disturbance or continuation of the same trajectory will ultimately lead to a crash. Such a situation is characterised by one or more stability coefficients $C_{s_i} > 1$ and a correlation coefficient $\mathcal{R}_\rangle < 1$ between the virtual control command ν_{c_i} and the control realisation ν_{r_i}.

> **RQ2:** : How does the rotor failure impact the stability of the quadrotor?
> - How can the theoretical stability of the quadrotor controller be determined?
> - What is the theoretical domain of attraction for the quadrotor fault-tolerant control system upon single rotor failure?
> - What factors influence the stability of the quadrotor upon rotor failure?

With reduced attitude control methods, quadrotors can be controlled despite a rotor failure. The INDI controller by Sun et al. [6] has been mathematically proven to be stable for the SRF case, as long as the thrust vector has an upward component in inertial space [6]. However, this stability analysis foregoes the effects of actuator dynamics and saturation. As demonstrated in the scientific paper in section IV, the attainable virtual control set is severely limited due to the rotor speed limits. Additionally, conventional INDI designs have ignored singular dynamics due to actuator lag. At high controller effort, these limitations can lead to saturation of the rotors that introduce unpredictable, destabilising perturbations. As detailed in section V of the scientific paper, this destabilising effect can be determined by introducing the hedged virtual control signal as a perturbation term in the external state equation. By application of Lyapunov's direct method, the stability of the external states can be assessed. A domain of attraction is difficult to quantify, due to the existence of highly nonlinear internal dynamics, combined with the asymmetry of the attainable virtual control set.

RQ3: What factors are the leading causes of Loss-of-Control?
- What external factors contribute to the the occurrence of a loss-of-control situation?
- How can these factors lead to a loss-of-control situation?
- What scenarios or trajectories place the quadrotor and its control system at risk of a loss of control?

As mentioned in the answer of RQ2, the main factor is the saturation of the rotors, which is a result of multiple other factors. First of all, external aerodynamic disturbances at high speed are a large cause of rotor saturation, which was also demonstrated by Sun et al.[3]. At high speed, the magnitude of these disturbances grow, requiring more opposite control input to stabilise the quadrotor along its trajectory. Secondly, whilst the actuator dynamics are inverted by the controller, they induce lag to the system. Due to this, larger virtual control is needed to achieve the desired error dynamics. Similarly, commanding large inertial acceleration vectors can lead to instability. Such acceleration profiles require large attitude and thrust realisations, which can lead to saturation. As shown in the scientific paper, saturation induces undesired perturbations that can destabilise the quadrotor.

RQ4: How can the occurrence of loss-of-control scenarios be predicted?
- Are there observable variables that are indicative of the loss-of-control performance?
- If so, are there observable patterns that indicate the onset of loss-of-control before its actual occurrence?
- Are these patterns visible for all of the envisioned loss-of-control scenarios of *RQ3*?

As is described in detail in the scientific paper, loss-of-control can be predicted by means of the stability margin. Whilst it is prohibitively complex to analytically predict the magnitude of the hedged virtual control signal and thus the perturbed stability, it is possible to measure this quantity. Using measured external states and known control parameters, the stability of the quadrotor can be assessed.

RQ5: How can loss-of-control be prevented?
- Can safe constraints be defined on the flight envelope?
- Can the onset of loss-of-control be predicted sufficiently early to prevent its occurrence?
- How can these constraints be incorporated into existing fault-tolerant-control algorithms for quadrotors experiencing single rotor failure?
- How does loss-of-control prevention impact the performance, compared to FTC without such flight envelope protection?

Because of the complexity of analytical predictions of the hedged control signal, it is currently beyond scope of this research to define a safe flight envelope. However, loss-of-control can effectively be prevented by selecting a safe threshold and reducing the desired velocity and acceleration.

2.2. Answer to Main Question

Main Question: How can Loss-of-Control occurrence theoretically be predicted using control theory for quadrotors experiencing single rotor failure?

Over the course of this research, the stability of quadrotors in the SRF case has been analysed and quantified. It was found that perturbations due to actuator effects can potentially destabilise the quadrotor along its desired trajectory. In the scenario of an overspeed, high-speed aerodynamic disturbances ultimately lead to infeasible virtual control commands and a subsequent loss of control.

Due to the nonlinearity of the control effectiveness for reduced attitude control, analytical determinations of a safe flight envelope are highly complex. However, using measured variables, known physical constants and control parameters, the discrepancy between the virtual control and the control realisation can be quantified in the form of the hedged virtual control signal. Combined with measured state variables, the stability of the quadrotor along its trajectory can be quantified by means of Lyapunov's direct method.

Using the stability coefficient as a metric of the relative magnitude of the actuator perturbations, it is possible to obtain an early warning for the onset of degraded control performance. As an additional measure against overly conservative predictions, the correlation between the virtual control and realisation can be monitored. When the stability coefficients and the correlation coefficient exceed their safe threshold, the stability around the trajectory is greatly diminished. Upsets and loss of control situations can ultimately be prevented by scaling down the acceleration command at the occurrence of a loss-of-control warning.

3 Recommendations

Whilst this research provides a promising theoretical method to predict loss-of-control for quadrotors experiencing single rotor failure, further work is required to validate this methodology.

First of all, the actuator dynamics are currently assumed to be modelled accurately by a first order lag model. Whilst this assumption has been made in prior research on quadrotors[7][8][5], real-life validation of the methodology is required to assess its accuracy. Additionally, more detailed research into rotor speed control would add to the fidelity of the loss-of-control prediction. Direct control over the rotational acceleration of the rotors could potentially offer improved performance.

Secondly, a constant rotor thrust coefficient is assumed in the prediction methodology. In reality, nonlinear effects such as blade flapping, rotor interference and the rotor angle of attack affect the thrust coefficient of the individual rotors [9][10][11]. Whilst these effects were modelled in simulation, using the high-fidelity aerodynamic model in simulation [10], its effects are not incorporated in the stability analysis and controller design. Assessment of the effect of the rotor thrust coefficient variation on the stability of the quadrotor could potentially add accuracy to the prediction method.

Thirdly, the current methodology requires accurate knowledge of the quadrotor states. In earlier research with the Bebop 2 quadrotor, state-estimation using Optitrack motion capturing was used to provide an accurate estimation of the ground truth [3][6]. Analysis of the effect of measurement error was beyond scope for this research, so the validity of this loss-of-control prediction methodology under real-world conditions without accurate state knowledge is unknown. It is likely that accurate ground truth knowledge is required to provide accurate prediction of stability.

Additionally, it is recommended to further investigate the stability of the internal dynamics over a broader range of ζ. Whilst it can be shown that the internal dynamics largely remain stable as long as the external states are stabilised around the origin, the exact domain of attraction is unknown. Therefore, unexpected instability and loss-of-control may occur as a result of unexpectedly unstable internal dynamics.

Finally, it may be worthwhile to investigate utilising the stability analysis methodology to design trajectory planning methods or adaptive outer-loop controllers that can provide feasible trajectories for the INDI inner-loop controller. Whilst this was originally the goal of the research, this was deemed infeasible within the scope, due to the nonlinearity of the trajectory constraints.

References

[1] Alexander Lanzon et al. "Flight control of a quadrotor vehicle subsequent to a rotor failure". In: *Journal of Guidance, Control, and Dynamics* 37.2 (2014), pp. 580–591. DOI: 10.2514/1.59869.

[2] Mark W. Mueller et al. "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles". In: *International Journal of Robotics Research* 35.8 (2016), pp. 873–889. DOI: 10.1177/0278364915596233.

[3] Sihao Sun et al. "High-Speed Flight of Quadrotor Despite Loss of Single Rotor". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3201–3207. DOI: 10.1109/LRA.2018.2851028.

[4] Sihao Sun et al. "Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances". In: *Proceedings - IEEE International Conference on Robotics and Automation*. 2020. DOI: 10.1109/ICRA40945.2020.9197239. arXiv: 2002.09425.

[5] Peng Lu et al. "Active fault-tolerant control for quadrotors subjected to a complete rotor failure". In: *IEEE International Conference on Intelligent Robots and Systems* 2015-Decem (2015), pp. 4698–4703. DOI: 10.1109/IROS.2015.7354046.

[6] Sihao Sun et al. "Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors". In: *IEEE Transactions on Robotics* 37.1 (2020), pp. 116–130. DOI: 10.1109/TRO.2020.3010626. arXiv: 2002.07837.

[7] Ewoud J.J. Smeur et al. "Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles". In: *Journal of Guidance, Control, and Dynamics* 39.3 (2016), pp. 450–461. DOI: 10.2514/1.G001490.

[8] E.J.J. Smeur et al. "Prioritized Control Allocation for Quadrotors Subject to Saturation". In: *IMAV 2017: International Micro Air Vehicle Conference and Competition 2017* September (2017), pp. 37–43. URL: http://www.imavs.org/papers/2017/51_imav2017_proceedings.pdf.

[9] Robert Mahony et al. "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor". In: *IEEE Robotics and Automation Magazine* 19.3 (2012), pp. 20–32. DOI: 10.1109/MRA.2012.2206474.

[10] Sihao Sun et al. "Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 3868–3875. DOI: 10.1109/lra.2019.2928758.

[11] Sihao Sun et al. "Quadrotor gray-box model identification from high-speed flight data". In: *Journal of Aircraft* 56.2 (2019), pp. 645–661. DOI: 10.2514/1.C035135.

A

Verification

The methodology and simulation performance have been verified thoroughly to ensure the accuracy of the results. The main verification has been threefold, consisting of unit verification of the individual components of the simulation through unit verification. Additionally, the functioning of the INDI controller is verified through simple tracking tasks to assess controller behaviour. Finally, the loss-of-control prediction methodology is verified by application of the theory to a simple 1-DoF dynamic model of a point mass.

A.1. Unit Verification

The flight simulations using the Bebop 2 under single rotor failure have been conducted in a Matlab/Simulink environment. This simulated environment provides a dynamic model of the quadrotor and its actuators and was first used in the research of Sun et al. [3] A high-fidelity aerodynamic model of the Bebop2 quadrotor under single rotor failure was used to model the nonlinear aerodynamic effects.[10]. This research added the outer-loop PID and inner-loop INDI controllers to this simulated environment.

To verify the functioning of the individual components of the model, successive unit verification has been performed. For each of the components, simple dummy values were provided as input. The output of these tests was compared to the theoretical values derived analytically. Any discrepancies between these results were investigated and subsequently resolved.

A.2. Controller Verification

Following the unit verification of the individual components, the INDI controller design is verified using the quadrotor dynamic simulation. Here it is analysed whether the controller performance is satisfactor. This is done by providing a step input in the reference signals Z_{ref}, n_x^B and n_y^B and comparing the controller response to the theoretical third order response.

The theoretical response of the output in the Laplace domain was shown to be:

$$Y_{i_{actual}}(s)\left(s^3 + (k_1 + \omega_B)\, s^2 + (k_1\omega_B + k_0)\, s + k_0\omega_B\right) = 0 \tag{A.1}$$

Therefore, the response to a step input of Eq. (A.1) was simulated and compared to the outputs of the INDI controller. The INDI controller was provided a step in the reference signal for each of the outputs separately. First, the altitude response was simulated when given a reference altitude of 1 m, starting from an altitude of 0 m. The response is shown in Fig. A.1, where the blue trace depicts the measured response and the orange line depicts the theoretical response. What is visible is that the reference signal is tracked quickly and strongly resembles the theoretical model. Small error exists, which could be attributed to model uncertainties, filtering delays or aerodynamic effects.

The same process was repeated for the outputs h_1 and h_2, where similar performance was achieved by the controller. This shows that the controller behaves as expected and verifies that the error dynamics behave similar to the theoretical error dynamics with actuator delay.

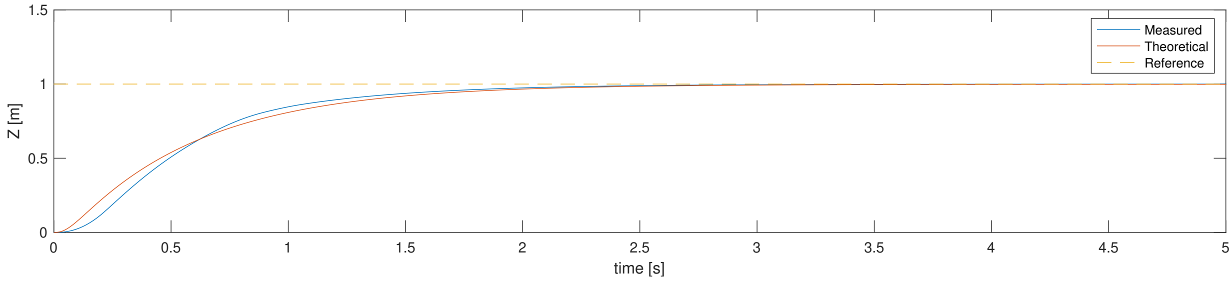

Figure A.1: Tracking performance of Z compared to the theoretical error dynamics provided by a second order model.

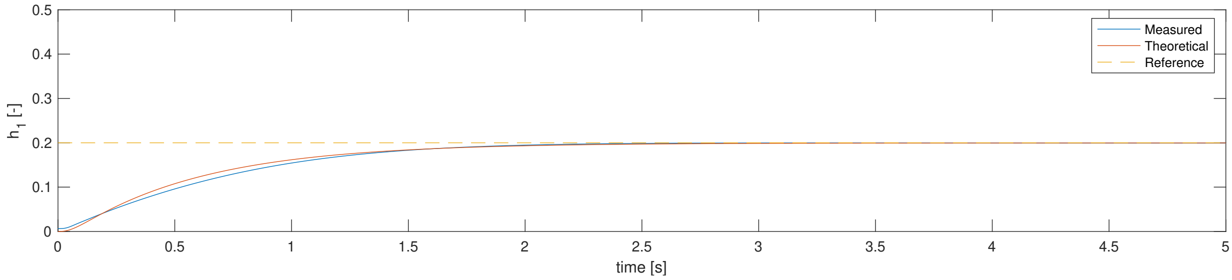

Figure A.2: Tracking performance of h_1 compared to the theoretical error dynamics provided by a second order model.

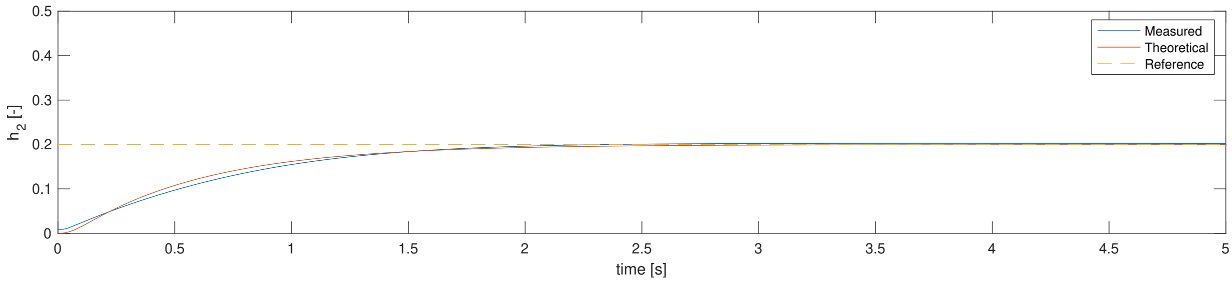

Figure A.3: Tracking performance of h_2 compared to the theoretical error dynamics provided by a second order model.

A.3. Prediction Method Verification

The prediction methodology based on the Lyapunov stability analysis was also verified to assess its accuracy. Whilst the results of the stability analysis on the SRF quadrotor seemed appropriate, the nonlinearity of the system makes them hard to interpret. Therefore, the same methodology is applied to a simple linear 1-DoF model of a point mass with $m = 1$. This point mass can moved by a controllable force, u, which acts with a first order lag. Additionally, an external disturbance force, d acts on the point mass.

The system dynamics are given by:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u + d \end{aligned} \tag{A.2}$$

This can be rewritten as:

$$\begin{aligned} &\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})u = \begin{bmatrix} x_2 \\ d \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ &y = x_1 \end{aligned} \tag{A.3}$$

The first order dynamics of the input force, including saturation u are taken as:

$$\dot{u} = 2\left(u_c - u - u_h\right) \tag{A.4}$$

Since the system is already in control canonical form as a series of output derivatives, no state transformation is required for control design. However, for clarity, the system is brought into the state transformed canonical form:

$$\begin{aligned} \dot{\boldsymbol{\xi}} &= \begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} x_2 \\ d \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ y &= \xi_1 \end{aligned} \tag{A.5}$$

Using an INDI design methodology, a controller can be designed that stabilises the point mass at $x_1 = 0$.

First a virtual control is defined as:

$$\nu = -\xi_1 - 2\xi_2 = -x_1 - 2x_2 \tag{A.6}$$

Subsequently, a control law is designed as:

$$u_c = G^{-1}\left(\nu - \ddot{y}_2\right) + u \tag{A.7}$$

Without actuator dynamics, the output dynamics then become:

$$\ddot{y} = \ddot{x_1} = -\xi_1 - 2\xi_2 \tag{A.8}$$

In the Laplace domain, this becomes:

$$Y(s)\left(s^2 + 2s + 1\right) = 0 \tag{A.9}$$

Under the influence of actuator dynamics, the actual error dynamics can be shown to be:

$$\overline{\;y\;} = G\left(u_c - u\right) \tag{A.10}$$

Applying the INDI control law for u_c to these dynamics, the actual dynamics under actuator delay become:

$$\begin{aligned} \overline{\;y\;} &= G\left(G^{-1}\left(\nu - \dot{x}_2\right) + u - u\right) \\ &= \nu - \dot{x}_2 \end{aligned} \tag{A.11}$$

In the Laplace domain, the dynamics are obtained by delaying the Laplace domain dynamics of $\ddot{y}$ by the first order actuator dynamics.

$$\begin{aligned} Y(s)\left(s^2 + 2s + 1\right)(s+2) &= 0 \\ Y(s)\left(s^3 + 4s^2 + 5s + 2\right) &= 0 \end{aligned} \tag{A.12}$$

Now, an INDI controller can be designed that compensates for the actuator delay. The virtual control is based on the Laplace domain response of the delayed system:

$$\nu = -4\dot{x}_2 - 5x_2 - 2x_1 \tag{A.13}$$

The control law can then be taken as:

$$u_c = \frac{1}{2}G^{-1}\left(-4\dot{x}_2 - 5x_2 - 2x_1\right) + u - u_h \tag{A.14}$$

The hedged control input u_h signifies the difference between the control input commanded by the controller, compared to the command that is sent to the actuator. Due to command saturation, this command differs from the desired control input.

$$\begin{aligned} u_{des} &= \frac{1}{2}G^{-1}\left(-4\dot{x}_2 - 5x_2 - 2x_1\right) + u \\ u_h &= u_{des} - u_c \end{aligned} \tag{A.15}$$

Extending the state vector $\boldsymbol{x} = [\begin{array}{cc} x_1 & x_2 \end{array}]^T$ with the input u, the extended state vector becomes $\boldsymbol{x_a} = [\begin{array}{ccc} x_1 & x_2 & u \end{array}]^T$. The external state vector is also extended with an additional derivative, leading to:

$$\boldsymbol{\zeta} = \left[\begin{array}{ccc} y & \dot{y} & \ddot{y} \end{array}\right]^T \tag{A.16}$$

The external dynamics can be rewritten as:

$$\dot{\boldsymbol{\zeta}} = \left(\boldsymbol{A_a} - \boldsymbol{B_a}\boldsymbol{K_a}\right)\boldsymbol{\zeta} \tag{A.17}$$

Incorporating the effect of the saturation of u, these dynamics become:

$$\begin{aligned} \dot{\boldsymbol{\zeta}} &= \left(\boldsymbol{A_a} - \boldsymbol{B_a}\boldsymbol{K_a}\right)\boldsymbol{\zeta} - 2Gu_h \\ \dot{\boldsymbol{\zeta}} &= \left(\boldsymbol{A_a} - \boldsymbol{B_a}\boldsymbol{K_a}\right)\boldsymbol{\zeta} - \nu_h \end{aligned} \tag{A.18}$$

Using the theory from the stability analysis, a Lyapunov function $V(\zeta)$ is defined as:

$$\boldsymbol{V}(\boldsymbol{\zeta}) = \boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{\zeta} \tag{A.19}$$

The matrix P is found through solving the following Lyapunov equation:

$$\boldsymbol{P}(\boldsymbol{A}_a - \boldsymbol{B}_a\boldsymbol{K_a}) + (\boldsymbol{A}_a - \boldsymbol{B}_a\boldsymbol{K}_a)^T\boldsymbol{P} = -\boldsymbol{I} \tag{A.20}$$

$$\boldsymbol{P} = \begin{bmatrix} 2.03 & 1.44 & 0.25 \\ 1.44 & 2.4167 & 0.3889 \\ 0.25 & 0.39 & 0.22 \end{bmatrix} \tag{A.21}$$

The derivative of the Lyapunov function is then found using:

$$\begin{aligned} \dot{\boldsymbol{V}}(\boldsymbol{\zeta}) &= \frac{\partial V(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\dot{\boldsymbol{\zeta}} \\ &= \boldsymbol{\zeta}^T\left(\boldsymbol{P}(\boldsymbol{A}_a - \boldsymbol{B}_a\boldsymbol{K}_a) + (\boldsymbol{A}_a - \boldsymbol{B}_a\boldsymbol{K}_a)^T\boldsymbol{P}\right)\boldsymbol{\zeta} - \boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B}_a\boldsymbol{\nu_h} - \boldsymbol{P}(\boldsymbol{B}_a\boldsymbol{\nu_h})^T\boldsymbol{\zeta} \\ &= -||\boldsymbol{\zeta}||^2 - 2\boldsymbol{\zeta}^T\boldsymbol{P}\boldsymbol{B}_a\boldsymbol{\nu_h} \end{aligned} \tag{A.22}$$

Following the same steps as the stability analysis methodology, the stability coefficient can be derived by manipulation of the Lyapunov derivative $\dot{V}(\boldsymbol{\zeta})$:

$$C_s = \frac{-2\boldsymbol{P}\boldsymbol{B_a}\boldsymbol{\nu_h}}{||\boldsymbol{\zeta}||^2} \tag{A.23}$$

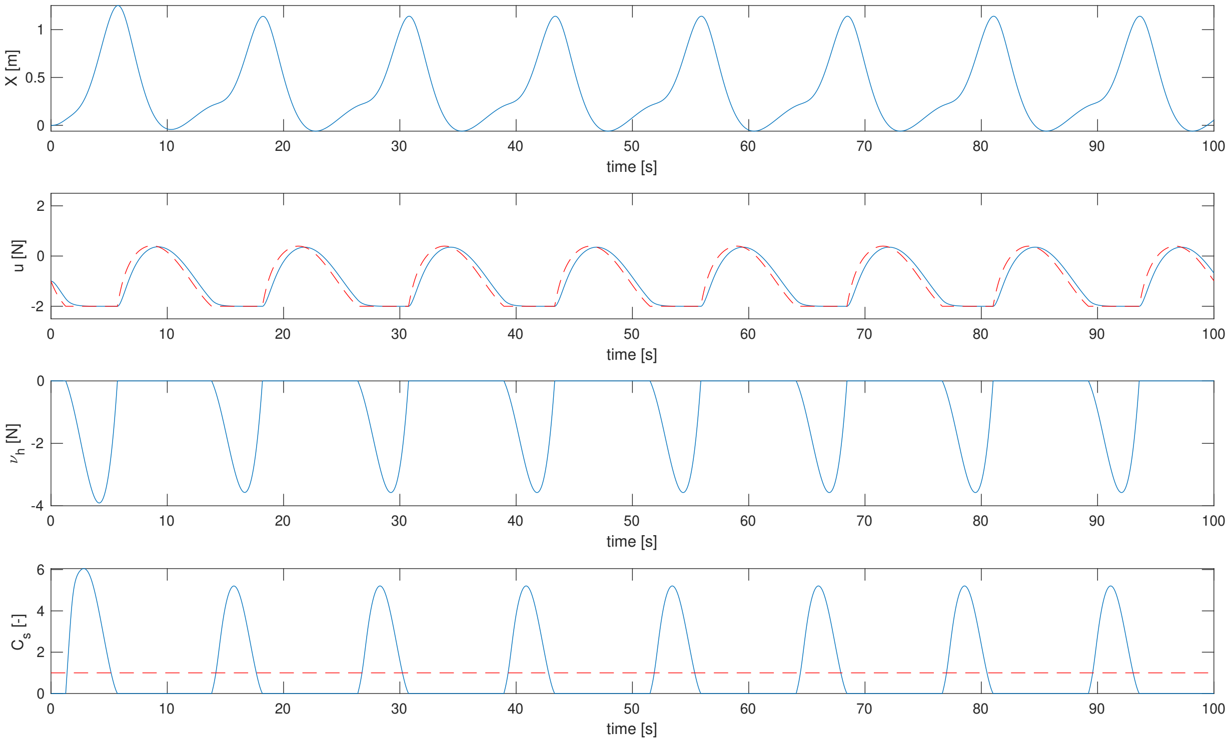

Figure A.4: The simulated results of the INDI controller with an initial position offset of 30m. Depicted are the position, input dynamics, hedged virtual control and the stability coefficient

The tracking performance of this system is now simulated to assess the performance, with a saturation on the input u defined by $u \in [-2, 2]$. Similar to the quadrotor, the commanded input also saturates for $u_c \in [-2, 2]$.

First the system is simulated with an initial condition of $x_{1_0} = 30$, the results can be seen in Fig. A.4. What is visible is that the system returns to its origin position under the INDI control law. Due to the large initial offset, large control effort is commanded, leading to saturation of the input. Initially a very high $\dot{u}$ is commanded that cannot be provided due to command saturation, leading to a very high value of ν_h. Due to the occurrence of saturation, the stability coefficient $C_s > 0$, reflecting the reduced control performance due to this occurrence. However,$C_s < 1$ for all t, since the error is steadily decreasing, despite saturation.

The observed behaviour of C_s is further explained by Fig. A.5, which depicts the desired control computed by the controller, compared to the input command that is provided to the actuator, u_c, which subject to command saturation $u_c \in [-2, 2]$. Initially, an extremely high input command is given to deliver the desired $\dot{u}$ given the actuator dynamics of this system. Command saturation subsequently leads to a much lower $\dot{u}$ than can be provided by the system, leading to a high value of ν_h and thus C_s.

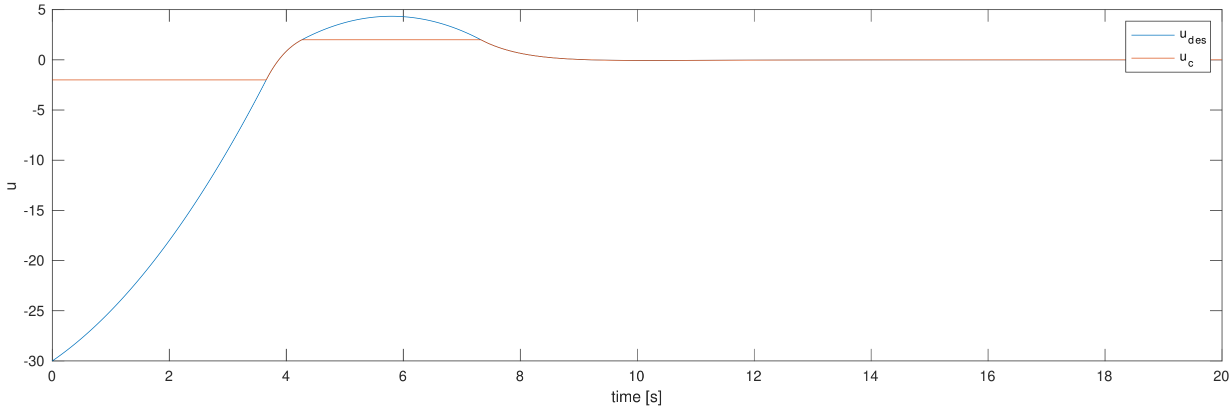

Figure A.5: The commanded control computed by the controller, compared to the input command that is provided to the actuator.

Now, a disturbance force is introduced as seen in Fig. A.6, for which the equation is given by:

$$d = 1 + 1.2sin(0.5t) \tag{A.24}$$

Figure A.7 depicts the response of the system as a result of this disturbance force acting on it. Here, it can be seen that the disturbance exceeds the minimum control force, leading to saturation and subsequent tracking errors. This is reflected by the hedged virtual control signal and the stability coefficient, which exhibits periodic peaks of instability, when the error can be seen to increase strongly. However, this error can be compensated over the period of the disturbance d, so the system remains approximately stabilised at $x_1 = 0$. This mirrors what can be seen in the behaviour of the quadrotor when it approaches the limit of its flight envelope.

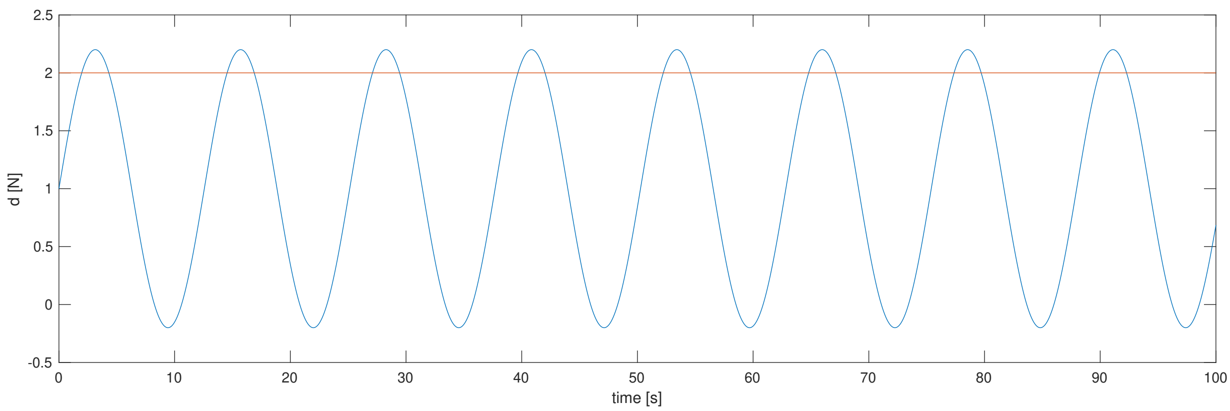

Figure A.6: The disturbance d acting on the system.

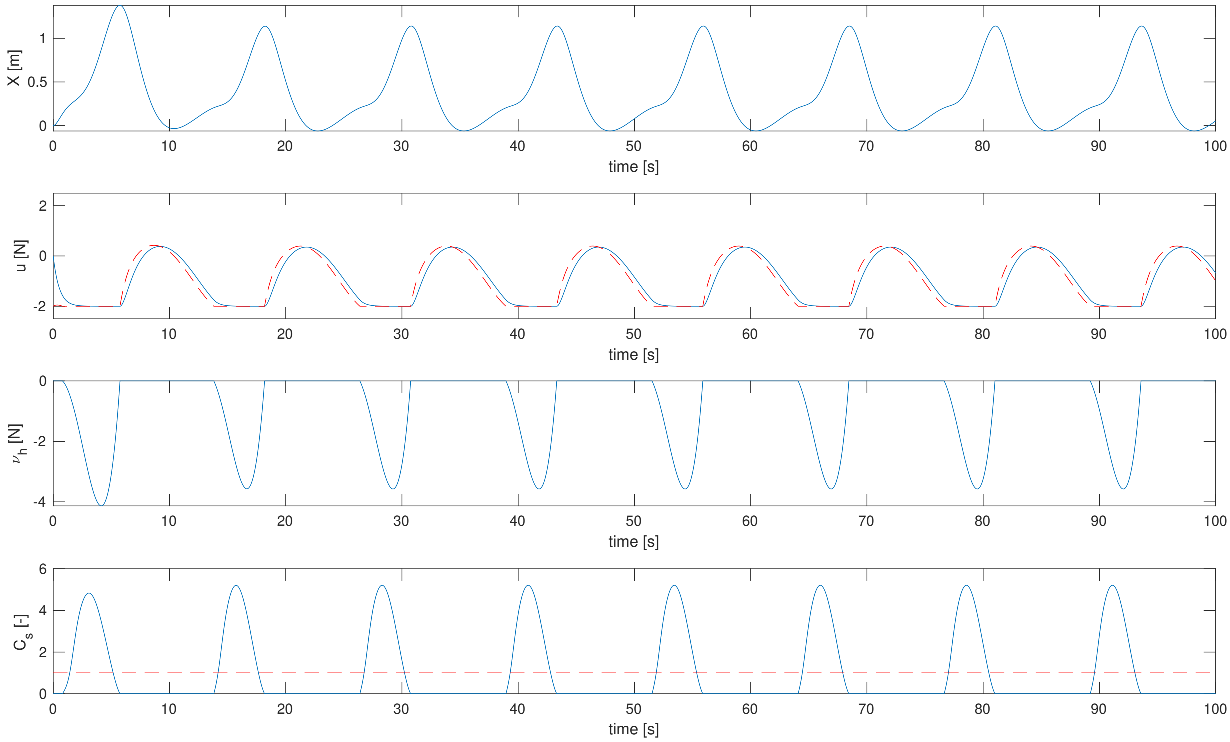

Figure A.7: The simulated results of the INDI controller with a disturbance d. Depicted are the position, input dynamics, hedged virtual control and the stability coefficient

Now, the disturbance force is slowly increasing, as seen in Fig. A.8, given by the equation:

$$d = 0.05t + 1.2sin(0.5t) \tag{A.25}$$

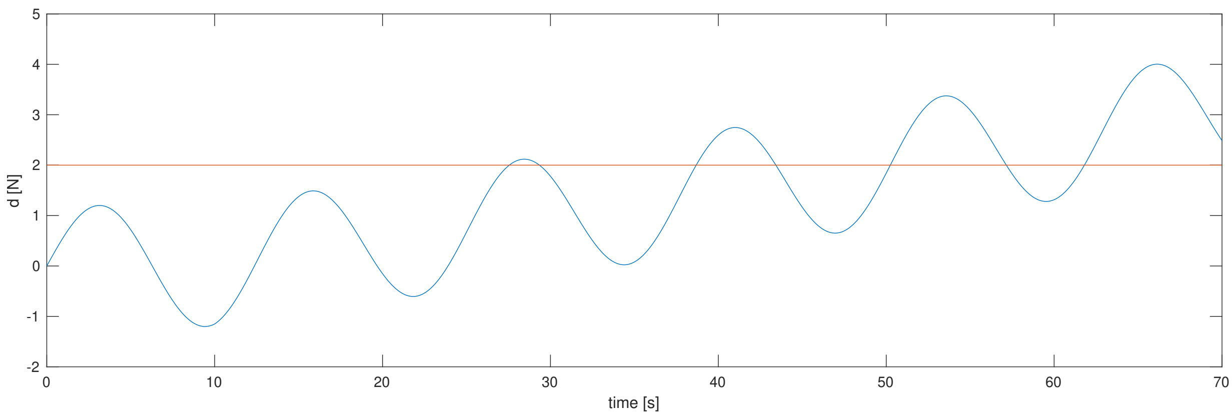

Figure A.8: The disturbance d acting on the system.

It is expected that ultimately, divergence of the error occurs, which should be reflected by the stability coefficient, C_s. The performance of the system is depicted in Fig. A.9. It can be seen that due to the increasing disturbance, the error ultimately diverges. The stability coefficient reflects this as its average starts to increase, ultimately leading to C_s remaining larger than 1 as the error starts to diverge.

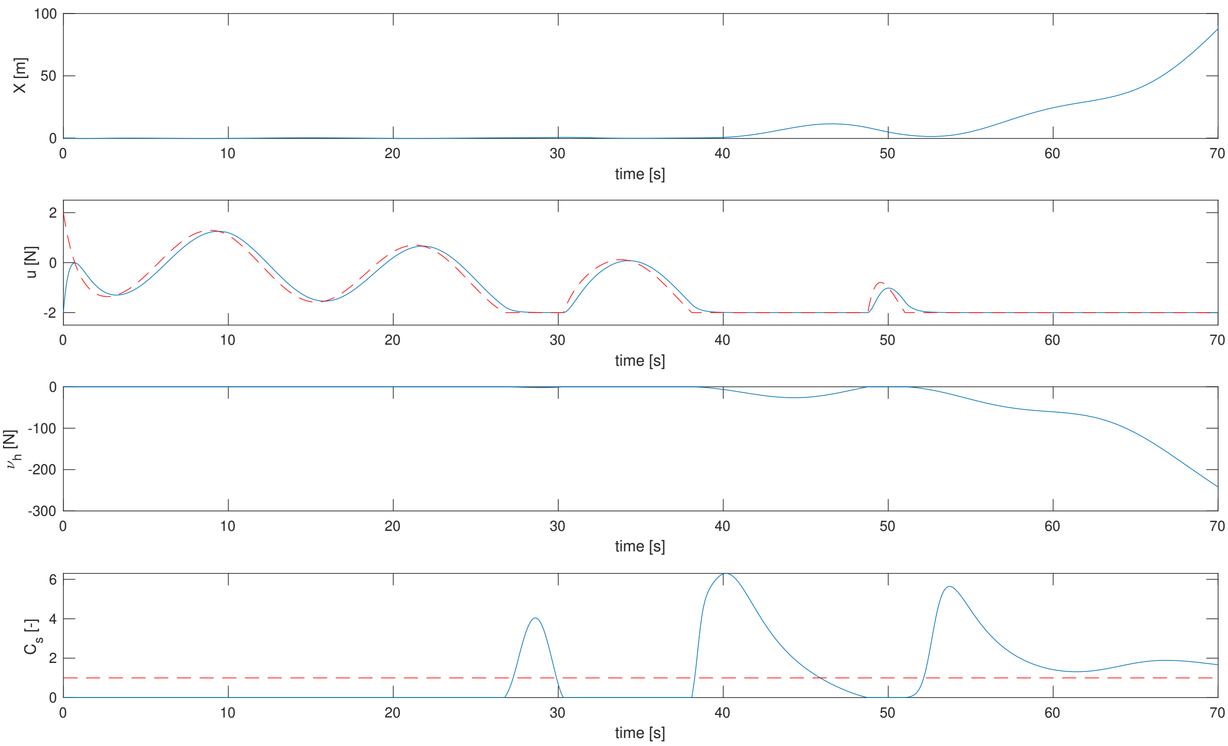

Figure A.9: The simulated results of the INDI controller with a disturbance d. Depicted are the position, input dynamics, hedged virtual control and the stability coefficient

Ultimately, this simplified model provides a more physically intuitive explanation of the mechanism behind the loss-of-control prediction. It is shown that the methodology behaves as expect given the scenarios that are provided.

Quadrotor Aggressive Maneuvering despite single rotor failure

T. Hoppenbrouwer 01-05-2021

Technische Universiteit Delft

TUDelft

Preliminary Report

Quadrotor Aggressive Maneuvering despite single rotor failure

by

T. Hoppenbrouwer 01-05-2021

Table 1: Version Registration

Report	**Version**	**Date**
Preliminary Report	1.0 First Draft	25-03-2021
Preliminary Report	2.0	01-05-2020

Author: Tom Hoppenbrouwer 4297628

Literature Study Duration: November 9, 2020 – March 2021

Supervisor: dr. ir. C.C. de Visser TU Delft

Contents

1 Introduction 1

2 Thesis Outline 2
- 2.1 Motivation . . . 2
- 2.2 Research Questions. . . . 3

3 Literature Study 5
- 3.1 Quadrotor Model . . . 5
  - 3.1.1 Dynamic Model . . . 5
  - 3.1.2 Aerodynamic Effects . . . 6
  - 3.1.3 Performance Characterisation. . . . 8
- 3.2 Quadrotor Fault-tolerant Control . . . 9
  - 3.2.1 Reduced Attitude Stability . . . 9
  - 3.2.2 Fault-tolerant Control Architecture . . . 10
  - 3.2.3 Position Control . . . 11
  - 3.2.4 Reduced Attitude Control. . . . 11
- 3.3 Trajectory Generation . . . 12
  - 3.3.1 Challenges . . . 12
  - 3.3.2 Parametrized Maneuver Sequences . . . 13
  - 3.3.3 Differential Flatness . . . 13
  - 3.3.4 Machine Learning Based Methods . . . 15
- 3.4 Literature Conclusions . . . 15

4 Research Goal and Scope 18
- 4.1 Research Objective . . . 18
- 4.2 Research Scope . . . 18

5 Thesis Approach 20
- 5.1 Work Package 1: Design . . . 20
  - 5.1.1 Modeling. . . . 20
  - 5.1.2 Trajectory Generation . . . 20
  - 5.1.3 Integration . . . 21
- 5.2 Work Package 2: Development . . . 22
  - 5.2.1 Build Simulation . . . 22
  - 5.2.2 Simulation . . . 22
  - 5.2.3 Algorithm Improvement . . . 22
- 5.3 Work Package 3: Implementation. . . . 22
  - 5.3.1 Code Development . . . 23
  - 5.3.2 Experiment Preparation. . . . 23
  - 5.3.3 Testing . . . 23
  - 5.3.4 Improvement . . . 23
- 5.4 Work Package 4: Finalization. . . . 24
- 5.5 Verification & Validation . . . 24
- 5.6 Experiment Set-up . . . 24
  - 5.6.1 Quadrotor Platform. . . . 24
  - 5.6.2 Quadrotor Simulator . . . 25
  - 5.6.3 Test Facility . . . 25

6 Project Planning 26
6.1 Work Package 1 26
6.2 Work Package 2 26
6.3 Work Package 3 27
6.4 Work Package 4 27

7 Conclusion 29

Bibliography 30

1
Introduction

The use of Unmanned Aerial Vehicles (UAVs) has explosively increased as of recently for both consumer and commercial purposes, while new potential use cases are developed at a rapid pace. Due to their adaptability, relatively simple operations and great agility, drones have become popular commercial assets. With decreasing costs and the advent of more potent hardware, drones are seeing use in package delivery, surveillance, filming, construction and agriculture, among others. With the increased use of consumer UAVs, safety is becoming an ever more critical aspect in the development of drones.

[23] This is specifically important regarding the widely adopted quadrotor drones, which are equipped with four rotors and have the advantage of relative simplicity and low cost compared to their multirotor counterparts. The downside to the quadrotor is its inherent vulnerability to the loss of a rotor. Upon such rotor loss, the quadrotor becomes underactuated, leading to a loss of full attitude controllability. If unaccounted for, this can lead to loss of control and ultimately mission failure. [29] [17].

Recently, there have been numerous developments in the area of fault-tolerant control that have enabled sustained flight for quadrotors experiencing, single, double or even triple rotor failure. By means of new attitude representations and the use of non-linear control methods, damaged quadrotors can be stabilized and controlled to such an extent that mission continuation is possible, albeit with a penalty to the overall performance envelope. These developments add to the reliability and versatility of quadrotors and largely negate its largest deficiency.

Currently, research efforts focus on further improving the capabilities of quadrotors, in an effort to ameliorate their dependability. Research has been conducted into recovering damaged quadrotors from large initial disturbances to prevent a loss of control, for example. Additionally, current research is being done into characterizing the safe flight envelope, which can be used in control design, preventing a quadrotor from ending up in a loss of control situation. With these modern developments, the goal of this research is to add to this active area of research and expanding the flight envelope of a damaged quadrotor. This goal has been narrowed down to the development of a quadrotor guidance algorithm that is capable of generating aggressive maneuvering trajectories that are feasible within the performance constraints posed by a single rotor loss. To the best of the author's knowledge, this has not been attempted before.

This goal of this report is to provide a clear outline of the goals and scope of this thesis project. Additionally, it will provide an overview of the relevant concepts that have been investigated throughout a literature review, in which a gap in the contemporary literature has been identified. Finally, the report will discuss the structuring of the thesis, identifying the most important research areas and establishing an envisioned methodology and planning for the upcoming months.

In this report, chapter 2 discusses the motivation and the research questions. Chapter 3 provides an overview of the literature study that has been conducted. In chapter 4, the research objectives are proposed, including an overview of the assumptions that have been made to limit the research scope. Chapter 5 defines the approach, research methodologies, development logic and the experimental set up. Finally, chapter 6 provides an overview of the project planning in the form of a Gantt chart, followed by the conclusions in chapter 7.

2
Thesis Outline

The first step is to establish a motivation for the research, which is meant to provide an overview of the relevance and importance of this research. This is followed by an overview of the research questions that will be used to guide and structure the research.

2.1. Motivation

In recent years, many advancements have been made in the development of fault tolerant control systems that enable control over a quadrotor, despite single, double or even triple rotor failure. While initially this meant the ability to remain in hover or land, methods have been developed that allow for the continuation of high speed flight, despite rotor loss. [17] [29][44]
First of all, the development of relaxed hover solutions enables a quadrotor experiencing rotor failure to maintain a steady hovering position, and enabling limited control over the vehicles position, independent of the vehicle yaw angle. This is done by ceding control over the yaw angle and controlling the orientation of the vehicles thrust axis by means of periodic application of pitch and roll commands. [17] [29] [23]
More recent applications of FTC incorporate active fault detection and isolation, enabling real world application of quadrotor control under rotor failure. [39] [19] And, finally, the application of Incremental Non-linear Dynamic Inversion (INDI) for attitude control and control allocation, reduced the model dependency and negated the need for predetermined periodic relaxed hover solutions. This enables the continuation of high speed flight, eliminating the need for an emergency landing upon rotor failure. [42]

Very recently, the first research has been conducted into recovery from upsets for quadrotors with a single failed rotor. These upsets are defined as a large deviation from normal operating conditions in which the controllability over one or multiple states has been lost. While this rarely occurs with highly controllable nominal quadrotors, a quadrotor with rotor failure is more vulnerable towards the occurrence of upsets.
With decreased control over the attitude upon rotor loss, unwanted pitch or yaw rotations beyond 90^0, quickly lead to loss of altitude control, especially in case of nonzero pitch and/or roll velocities. In previous investigations by Sun, Baert, Strack van Schijndel and de Visser a method was developed to recover from this loss of control, by performing a sequence of actions that lead to a step-by-step recovery of controllability over the uncontrollable states. [44]

While the current state-of-the-art method is very potent, there is still room for improvement, considering its reported success rate of 71%. Additionally, this existing method is said to be less effective upon encountering large initial rotational rates, specifically when these rotational rates are in the opposite direction of the required stabilizing yaw rotational rate. The main reason for this is the fact that the current computation of the rotation required for flipping the quadrotor can produce sub-optimal rotations, especially when rotation around multiple axes exist upon initialisation. A potential means of improving the performance of quadrotor upset recovery is the generation feasible recovery trajectories that take into account the initial conditions upon upset, as well as the constraints on performance due to the loss of control authority. [44]

The development of such a feasible trajectory generation algorithm would enhance the dependability of the quadrotor platform. Recent fault-tolerant control innovations have increased the safety and reliability of quadrotors and have shown that despite the inherent vulnerability to rotor loss, the quadrotor is a very dependable form of transport. To enhance this dependability, there is value in expanding the flight envelope of the damaged quadrotor to the point that it can perform a vast majority of the missions of a nominal quadrotor system. One of such capabilities is to perform aggressive maneuvers, such as perching, obstacle avoidance and flips. The ability to do so would allow a quadrotor to continue its mission despite a rotor failure and the occurance of upsets, which would enable the ability to escape from a hazardous environment. This, combined with the research on high speed flight using a quadrotor experiencing rotor failure would demonstrate greater versatitility of quadrotors and would strongly negate their main perceived weakness.

Currently, a wide variety of trajectory generation methods for quadrotors or rigid bodies in general exist. To achieve success, it is critical to find and develop a trajectory generation method that can handle non-zero initial conditions, guarantees feasibility of the generated trajectory and is capable of doing so in real-time on the quadrotor hardware. Additionally, it is important that trajectories can be defined independent of the yaw angle.

In recent years, many suitable real-time trajectory algorithms have been developed using jerk or snap limitation, dynamic motion primitives or differential flatness properties. However, as of January 2021, no attempt has been made to perform aggressive maneuvering with quadrotors experiencing rotor failure. However, reduced attitude control has been used in the past to track a flip trajectory, proving that trajectory generation based on reduced attitude control is feasible. This however, was done on a quadrotor drone with four functioning motors with a trajectory that was generated in advance. [3] Therefore, a significant gap seems to exist in generating feasible trajectories for quadrotors that experience rotor failure, even though significant benefits exist, as mentioned before.

2.2. Research Questions

Having established a strong motivation, relevance of future research, as well as a research gap, the next step is to structure the work and identify important aspects of the work. Following the initial literature investigations, a multitude of important topics was identified, which were condensed into research questions. These questions address the most prominent challenges in being able to realize upset recovery and aggressive maneuvering for quadrotors and are used to structure the future stages of research and development. The main research question is as follows:

Main Question: How can a quadrotor experiencing single rotor failure generate and track feasible aggressive maneuvers and perform recovery from attitude upsets?

In this main research question, the aggressive maneuvers are to be defined as maneuvers such as flips, perching or high speed obstacle avoidance, which are generally far from the hover condition in terms of attitude, velocity and rotational rates. The subject of attitude upset recovery additionally adds the difficulty of being able to operate in real time from non-zero initial conditions. To structure the results, a number of sub-questions has been condensed that divide the research into smaller research domains.

RQ1: How can a trajectory generation problem be formulated efficiently to enable aggressive maneuvers with quadrotors experiencing rotor failure, given the dynamics of a quadrotor upon rotor failure?
- How can the optimisation problem of the trajectory generation be formulated effectively?
- How can a trajectory be parametrized for a quadrotor experiencing rotor failure?
- How can the trajectory optimisation be guaranteed to be computationally feasible within the existing hardware constraints?

RQ2: How can the feasibility of trajectories be guaranteed?
- How can the safe flight envelope be characterised efficiently?
- How can this safe flight envelope characterisation be translated into constraints for a trajectory generation algorithm?

RQ3: How can the trajectory generation guidance best be integrated into existing fault-tolerant control architectures?
- How can trajectory generation provide suitable reference signals to inner loop reduced attitude controllers?
- How can the trajectory generation algorithm be made sufficiently robust to deal with tracking errors?

RQ4: Can a trajectory generation algorithm be used to perform recovery from attitude upsets upon rotor failure?
- How can a trajectory generation algorithm cope with initial conditions that lie outside of the estimated performance envelope?
- How can online trajectory generation be made sufficiently fast to perform real-time recovery from upsets?

RQ5: To what extent does the developed online trajectory generation increase the capabilities and performance envelope of quadrotors upon single rotor failure?

The first question addresses the primary problem of identifying effective methods to formulate and solve a trajectory generation problem for a damaged quadrotor. Due to the continuous yaw rotation used by fault tolerant control and the reduced control over the vehicle attitude, motion planning for a damaged quadrotor is expected to have added complexity, compared to nominal quadrotors. This question is designed to identify effective methods for the formulation and optimization of the trajectory planning problem for a damaged quadrotor and to find feasible methods for parametrizing a trajectory.

The second question is used to identify methods to guarantee that the trajectories that are generated are feasible, given the significant constraints on the safe flight envelope of a quadrotor upon rotor failure. First it is to be addressed how these constraints can be characterized, after which it is to be answered how these can be incorporated into the trajectory generation.

Research question 3 addresses the potential difficulties in integrating the trajectory generation algorithm with state-of-the-art fault tolerant control. Given the significant research that has been performed into fault-tolerant control for quadrotors in the past years, it would be beneficial to design a solution that is compatible with existing fault-tolerant control architecture. The main points of interest are the generation of reference inputs that are compatible with the FTC methods, as well as finding ways of coping with potential tracking errors.

Research question 4 examines the main difficulties in implementing complicated and highly dimensional trajectory generation algorithms in real time, for arbitrary initial conditions. To enable performing upset recovery, recovery trajectories should be generated within a short time frame to prevent loss of control, from arbitrary initial conditions. RQ4 is used to examine if and how this can be performed.

Finally, research question 5 is meant to inquire what kind of new capabilities real-time trajectory generation enables for quadrotors experiencing rotor failures, in comparison to quadrotors without such system. This would entail investigating what kind of aggressive maneuvering is possible with a fully developed algorithm. Additionally it requires quantifying whether the developed algorithm can improve the performance and safety for recovery from attitude upsets, and if so, to what extent.

3

Literature Study

This chapter presents an overview of the available literature on the current state-of-the art in fault-tolerant control and trajectory guidance for quadrotor UAVs. The purpose of this is to identify key knowledge for the development of fault-tolerant trajectory guidance and simultaneously define a gap in the current state-of-the art that can be explored during the subsequent thesis phase. First, section 3.1 characterizes the dynamics of a quadrotor upon single rotor failure. Additionally, some modelling considerations are discussed that of importance in the design of the trajectory generation and the control systems. Consequently, section 3.2 provides an overview of state-of-the-art fault-tolerant control strategies that allow for continued quadrotor operation, despite the single rotor failure. Section 3.3 then explores the possible methodologies for the generation of feasible trajectories and evaluates different methods for the implementation of such trajectories.

3.1. Quadrotor Model

The first important step in the design of an aggressive maneuvering controller is the definition of a dynamic model for the quadrotor upon damage. First, the equations of motion required for the design of trajectory generation algorithms and control systems are presented in section 3.1.1. Secondly, important aerodynamic effects that come into play upon aggressive maneuvers are presented in section 3.1.2. Finally, the limitations on the performance of the damaged quadrotor are treated in a discussion of the attainable moment set in section 3.1.3.

3.1.1. Dynamic Model

For the development of the controllers and trajectory generation, equations of motion are required for the quadrotor upon single rotor failure. The quadrotor is a UAV with four fixed pitch rotors, for which the attitude and accelerations are controlled by means of varying the rotor thrusts. In literature on quadrotor control, two types of quadrotor models exist. Conventionally, quadrotor modelling is performed using a dynamic model based on Euler angles, as proposed by Mahony et al. [23]. This type of model is also used in fault-tolerant control of quadrotors in many different applications. [19][42][45]. The problem with the use of Euler angles is the existence of singularities and a phenomenon called Gimbal lock at pitch angles of $[\pm\frac{\pi}{2}]$. The singularity that occurs prevents the calculation of the Euler angular rates, while gimbal lock makes pitch and roll rotations interchangeable.[5] For operation near a hover condition, this is not problematic, but at larger deviations, problems start to occur. To overcome this, quadrotor equations of motion exist based on quaternions, as proposed by El-Badawy and Fresk.[5][7] Due to the lack of singularities, these models are more appropriate for the execution of aggressive maneuvers, for which larger pitch deviations are expected. [5] [7]

The Euler angle based model is represented in eq. (3.1) through eq. (3.5). The equations of translational motion in the inertial frame are shown in eq. (3.1) and eq. (3.2), in which $\boldsymbol{\zeta}$ denotes the position vector in the inertial frame and $\boldsymbol{v}$ denotes the inertial velocity vector. Additionally, m represents the vehicle mass, $\boldsymbol{g}$ is the gravity vector, $\mathbf{L}_{IB}$is the transformation matrix from the body frame to the inertial frame and $\mathbf{F}^B$ is the vector of forces in the body frame. [42]

Equation (3.3) and eq. (3.4) describe the rotational equations of motion. [42]

$$\dot{\boldsymbol{\xi}} = \boldsymbol{v} \tag{3.1}$$

$$m\dot{\boldsymbol{v}} = mg + \boldsymbol{L_{IB}}\boldsymbol{F_a^B} \tag{3.2}$$

$$\dot{\boldsymbol{L}}_{IB} = \boldsymbol{L}_{IB}\Omega \times I_v \tag{3.3}$$

$$\boldsymbol{I_v}\dot{\boldsymbol{\Omega}} = -\tilde{\boldsymbol{\Omega}}\boldsymbol{I_v}\boldsymbol{\Omega} + \boldsymbol{M^B} \tag{3.4}$$

$$L_{IB} = \begin{bmatrix} c\psi c\theta - s\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\phi c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix} \tag{3.5}$$

Traditionally, the attitude would be represented using the principal Euler angles, ϕ, θ and ψ, as is the case for conventional quadrotor control. The conversion of the rotational rates in the body frame to the Euler angular rates would lead to singularities at pitch and roll angles of $\pm\frac{\pi}{2}$, making the use of this model infeasible for aggressive maneuvering. Only, Mueller and d'Andrea [28] present a novel way of representing attitude called the reduced attitude, which eliminates this problem. This approach enables the use of the more conventional Euler equations of motion, despite high attitude angle deviations in the research of Sun et al. [44] This approach will be elaborated upon in more detail in section 3.2.1.

Despite this fact, some sources in literature make use of a quaternion model for aggressive maneuvering, such as El-Badawy et al. [5] and Brescianini et al. [3]. The main advantage over the Euler angle or DCM approach is the computational efficiency of the quaternion approach. The use of the conventional model requires the computation of many sines and cosines which are very demanding on the computational hardware, especially if operated on lower cost systems.

Equation (3.6) describes the equation of translational motion, defined using quaternions.[7] In this equation, $\boldsymbol{q}$ is the quaternion rotation vector, q^* is its conjugate, while $\otimes$ is the Kronecker product. The difference with the translational equations of motion in the Euler/DCM model in eq. (3.1) is that the rotation matrix has been replaced by a more computationally efficient quaternion multiplication. The equations of rotational motion can be seen in eq. (3.7) and eq. (3.8).[7][5] Herein it is important to note that eq. (3.8) is the same equation as eq. (3.4), but was included for a more complete overview of the equations of motion.

$$m\dot{\boldsymbol{v}} = m\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \mathbf{q} \otimes \boldsymbol{F_a^B} \otimes \mathbf{q}^* \tag{3.6}$$

$$\dot{\boldsymbol{q}} = -\frac{1}{2}\begin{bmatrix} 0 \\ \boldsymbol{\Omega} \end{bmatrix} \otimes \boldsymbol{q} \tag{3.7}$$

$$\dot{\boldsymbol{\Omega}} = I_v^{-1} \cdot \boldsymbol{M^B} - I_v^{-1}\left[\boldsymbol{\Omega} \times (I_v \cdot \boldsymbol{\Omega})\right] \tag{3.8}$$

3.1.2. Aerodynamic Effects

The equations of motion contain an aerodynamic force and moment vector $\boldsymbol{F^B}$ and $\boldsymbol{M^B}$ that has not yet been discussed. First of all, these two terms entail generated by the four rotors, both by thrust and other aerodynamic effects that occur due to rotor movement. Apart from aerodynamic forces and torques due to rotor thrust, external forces and moments that act on the fuselage exist, which occur mainly at higher speeds . In many control applications for quadrotors, the external aerodynamic forces are considered negligble [19] [28] [29] and are compensated through the use of aggressive gain selection [23] or the use of integral control on the outer position loop. [42].

With flight at higher speed combined with the loss of a rotor, however, more pronounced non-linear external aerodynamic effects exist, that can have a strong impact on tracking performance, as described by Sun et al.[41] [42] Additionally, performing aggressive maneuvers may induce additional non-linear aerodynamic effects, which have to be taken into account for the synthesis of control and trajectory generation methods. This section investigates known models for the external aerodynamic effects for the various envisioned flight conditions for the quadrotor over the course of this thesis.

The aerodynamic effects for a nominal quadrotor near hover conditions and at lower speeds have been discussed in detail by Mahony et al [23] and Martin et al. [24]. First of all, starting at low flight speeds, the airflow over the rotor leads to aeroelastic effects on the rotors, causing a difference between the properties of the advancing and retreating rotor blades. These aeroelastic effects on the rotors due to this blade flapping phenomenon cause an induced drag in the xy-plane of the quadrotor body frame. Mahony et al [23] present a linear model for

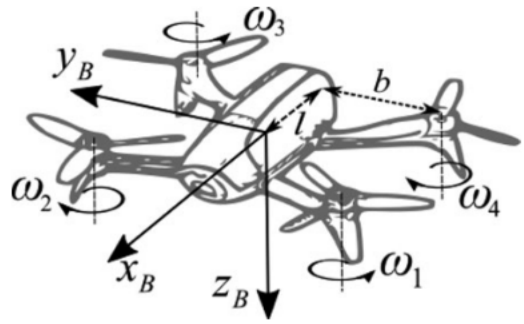

Figure 3.1: Schematic representation of the body fixed frame for the Parrot Bebop 2 quadrotor drone.[42]

this drag, which is dependent on a number of physical rotor properties and the quadrotor translational velocity. This aerodynamic effect can cause degradation of the tracking performance of the control systems [24] and needs to be taken into account when designing position and velocity control.

At higher speeds using a nominal quadrotor, variation of thrust can occur due to variations in the angle of attack on the individual rotors. While the rotor thrust is assumed to be dependent only on rotor speed ω_i and thrust coefficient κ_0, the rotor thrust for a given rotor speed can vary with velocity as described by Huang et al. [11] This change in rotor effectiveness impacts the controller performance of quadrotors, due to the disparity between the actual and modelled dynamics.

Additional aerodynamic effects occur in the form of drag torque, as a result of rotational velocity. In some literature, this drag torque is taken to be proportional to the vehicle yaw rate for simplicity. [19] [28]. Mueller [29] also proposes a model that assumes a quadratic relationship between the vehicles rotational velocity vector and the reaction torque. This reaction torque is especially important, due to the non-zero angular velocity of a quadrotor with rotor failure, as will be elaborated upon in section 3.2. This moment is especially noticeable at higher flight speeds and can strongly degrade control performance. [42].

Given this knowledge, later research expanded on this subject by means of identifying extensive aerodynamic models experimentally, which take into account complex aerodynamic interaction effects and can achieve a higher accuracy than conventional reduced aerodynamic models. [43][13]

With flight of a quadrotor upon rotor failure, the external aerodynamic disturbances that occur are different from those in nominal conditions. This is due to the continuous rotation that is required for the stabilization and control of the damaged quadrotor, which is further elaborated in section 3.2. These altered circumstances render the existing literature into quadrotor aerodynamics less useful. Research into the aerodynamics of quadrotors upon rotor failure has, however, been done in depth by Sun and de Visser. [42][41]. In [42], highly non-linear aerodynamic effects were identified, and for which an extensive model was developed in [44]. It was observed that the measured drag torque varied for different yaw angles, despite constant aerodynamic velocity, adding great complexity to designing robust controllers. Additionally, it was found that the linear models for induced drag in the lateral direction are invalid for high speed flight with a single rotor failure. [42] Given this knowledge, a comprehensive non-linear aerodynamic model was identified, that takes into account the complex interference effects that occur upon single rotor failure and can be used for the development of guidance and control algorithms. [41]

$$\boldsymbol{F}_a^B = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} \sum F_{x,i} \\ \sum F_{y,i} \\ -\sum T_i \end{bmatrix} + \boldsymbol{F}_f^B \tag{3.9}$$

Equation (3.9) represents a model for the aerodynamic forces acting on the fuselage[44], which is split into a component that is dependent on rotor inputs and a component of external aerodynamic forces acting on the fuselage, $\boldsymbol{F}_f$. In this equation, $F_{x,i}$ and $F_{y,i}$ describe the forces in the x- and y-directions on individual rotors, whereas T_i describes the thrust of each rotor.

$$\begin{bmatrix} F_{x,i} \\ F_{y,i} \end{bmatrix} = k_1 \begin{bmatrix} u_i\omega_i \\ v_i\omega_i \end{bmatrix} + k_2 \begin{bmatrix} v_i s_{n,i}\omega_i \\ -u_i s_{n,i}\omega_i \end{bmatrix} \tag{3.10}$$

$$\boldsymbol{F}_f = \frac{1}{2}\rho|\boldsymbol{V}|^2 S \begin{bmatrix} C_x \\ C_y \\ C_z \end{bmatrix} \tag{3.11}$$

The first term describes the effects of the thrust generated by the rotors, which acts in the body-frame z-direction, as well as the effects of the complex rotor interaction effects, such as blade flapping, which are described by a linear model in eq. (3.10). In this equation, k_1 and k_2 are constant parameters to be estimated in parameter estimation, u_i and v_i are the local air velocities over the rotor, ω_i are the individual rotor speeds and $s_{n,i}$ is a parameter that takes a value of 1 or -1 depending on the direction of rotation of the rotor.[44] The aerodynamic forces on the fuselage are described in eq. (3.11). These forces are dependent on the airspeed of the quadrotor and are described by three coefficients, C_x, C_y and C_z, which can be identified using the system identification methods described by Sun et al.[44]

$$\boldsymbol{M}^B = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} \sum s_{l,i} bT_i + \sum M_{x,i} \\ \sum s_{m,i} lT_i + \sum M_{y,i} \\ \sum s_{l,i} bF_{x,i} + \sum s_{m,i} lF_{y,i} + \sum M_{z,i} \end{bmatrix} + \boldsymbol{M}_f^B \tag{3.12}$$

Equation (3.12) models the aerodynamic moments that act on the the quadrotor, which similarly has a component that depends on the thrust and aerodynamics effects of the individual rotors, as well as external aerodynamic moments on the fuselage. The first term in eq. (3.12) describes the resultant torques due to the individual rotor thrusts, combined with the torques that occur due to the aerodynamic effects on the rotor blades, such as blade flapping. In this equation, $s_{l,i}$ and $s_{m,i}$ is positive or negative sign that takes a value depending on the direction of the rotors contribution to the roll and pitch moments, with k_3 and k_4 being two estimated parameters. The torques due to aerodynamic effects on the individual rotors are described by eq. (3.13). The aerodynamic effects on the fuselage are described by eq. (3.14). The parameters for this model are found through the system identification process described by Sun et al. [44]

$$\begin{bmatrix} M_{x,i} \\ M_{y,i} \end{bmatrix} = k_3 \begin{bmatrix} -v_i\omega_i \\ u_i\omega_i \end{bmatrix} + k_4 \begin{bmatrix} u_i s_{n,i}\omega_i \\ v_i s_{n,i}\omega_i \end{bmatrix} \tag{3.13}$$

$$\boldsymbol{M}_f = \frac{1}{2}\rho|\boldsymbol{V}|^2 Sb \begin{bmatrix} C_l \\ C_m \\ C_n \end{bmatrix} \tag{3.14}$$

3.1.3. Performance Characterisation

For the purpose of guidance and control of the quadrotor, it is critical to have a proper conception of the quadrotor's performance limitations upon damage. Rotor loss for a quadrotor logically leads to a loss of performance, reduced controllability and a smaller safe flight envelope. [28][17] [42] For the purpose of designing feasible trajectory generation and control systems, it is critical to properly quantify the remaining controllability of the damaged quadrotor to prevent pushing a damaged quadrotor into an upset from which it is unable to recover. [44] Characterising the flight envelope and its limitations, however, is quite a challenging subject.
For quadrotors, only one source can be found that provides a methodology for characterising the safe flight envelope. [40]. In this research, the forward and backwards reachable sets of a quadrotor using a Monte Carlo analysis. The intersection of these two sets is called the safe flight envelope in which the quadrotor is not expected to end up in a loss of control situation. This process, however has never been attempted for a faulty quadrotor and estimating this in this research is beyond scope, as it would require a full research in its own right.

Using a simplified model, using a sum-of-squares decomposition can be performed to estimate the recoverable set of a damaged quadrotor, as demonstrated in [34]. However, this model only takes into account rotation and therefore neglects the complex aerodynamic effects that occur upon rotation with a nonzero velocity vector, which is therefore also not suitable for this research. Finally, Mueller and d'Andrea [28] include a controllability analysis for a damaged quadrotor for a linearized model around a relaxed hover condition. While conceptually interesting, its linearized model is only valid for small angle deviations from its trim condition, rendering it infeasible for this research.

To still be able to characterise the controllability of a quadrotor experiencing rotor failure, Sun [44] and Narasimhan [31] propose the use of the attainable moment set (AMS), also known as the attainable virtual control set (AVCS). The AMS or AVCS is a mapping of the moments that can still be attained using the three remaining rotors, which is graphically represented in fig. 3.2. This is useful, because the main control impairment for a quadrotor is the loss of control over its attitude. What is visible is that the loss of a rotor strongly reduces

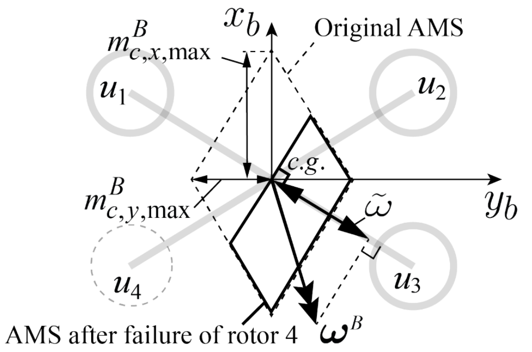

Figure 3.2: The attainable moment set in the xy-plane, including a graphical representation of the unsuppressable velocity $\tilde{\omega}$.[44]

the attainable moment set, making it impossible to generate uncoupled positive moments around the x-axis and uncoupled negative moments around the y-axis. What is specifically important to note, is the existance of unsuppressable rotational velocities due to the reduction in the attainable moments. In fig. 3.2, this is represented by $\tilde{\boldsymbol{\omega}}$, when given a rotation vector $\boldsymbol{\omega^B}$. Equation (3.15) provides a simplified first order ODE to model the rotational velocity around the x- and y-axes, assuming constant yaw rate. By explicit solution of this ODE using the current rotational velocity as initial condition and assuming a constant input for a specific time horizon t_h, an expression can be found for this unstoppable rotation at this time horizon. This expression can be seen in eq. (3.16) and can be used for the generation of constraints for trajectory generation algorithms, to prevent pushing the quadrotor into upset conditions. [44] [31]

$$\begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_x \end{bmatrix} = \begin{bmatrix} 0 & \frac{I_y - I_z}{I_x}\omega_z \\ \frac{I_z - I_x}{I_y}\omega_z & 0 \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \end{bmatrix} + \hat{G}_m u \tag{3.15}$$

$$\tilde{\omega}(t_h) = \phi \begin{bmatrix} \omega_x(t_h) \\ \omega_y(t_h) \end{bmatrix} = \phi\Phi_0(t_h) \begin{bmatrix} \omega_x \\ \omega_y \end{bmatrix} + \phi\Phi_1(t_h)\hat{G}_m \boldsymbol{u} \tag{3.16}$$

3.2. Quadrotor Fault-tolerant Control

With the loss of one rotor, the attitude control becomes underactuated, meaning that it is impossible to control the full vehicle attitude. As a consequence, a conventionally controlled quadrotor, as defined by [23] or [5], will become unstable and additionally becomes unable to control velocity and position. Fault-tolerant controllers have been designed that are capable of stabilizing quadrotors upon rotor loss and perform limited control over a quadrotors position, as shown by Mueller and d'Andrea [28][29] and Lanzon, Freddi and Longhi [17]. More recent developments have expanded the flight envelope of a quadrotor upon rotor loss with non-linear control methods, enabling higher speed flight and even recovery from large angle upsets. [19] [42] [44]

First, the reduced attitude method for the control over the vehicle attitude is described in section 3.2.1. Following this, different control methods and architectures that are available in literature are discussed in section 3.2.2. The subsequent subsections discuss the different elements of these control solutions.

3.2.1. Reduced Attitude Stability

At the heart of fault-tolerant control for quadrotors after full rotor failure is a new attitude representation that allows regaining control over the quadrotor atttude. Mueller and d'Andrea first defined a new method to represent the vehicle attitude by defining a constant vector in the body frame, $\mathbf{n}^B$ [30][28], as seen in figure fig. 3.3. This vector signifies the average thrust direction in the body frame, also known as the primary axis in other works. [19] [42] [45].

Using this new attitude representation, Mueller and d'Andrea [28][29] defined a method that is capable of stabilizing the quadrotor in a periodic stable solution, called a relaxed hover solution. Control over the yaw angle is given up and a reduced form of attitude control is performed through application of pitch and roll rates. For this relaxed hover, a periodic solution is defined with a constant non-zero angular velocity vector, where the translational velocity and acceleration may be non-zero, but averages to zero over the period of one rotation. With that, the vehicle will remain in a roughly constant position. [28]. By aligning the vector $\mathbf{n}^B$ in the opposite the

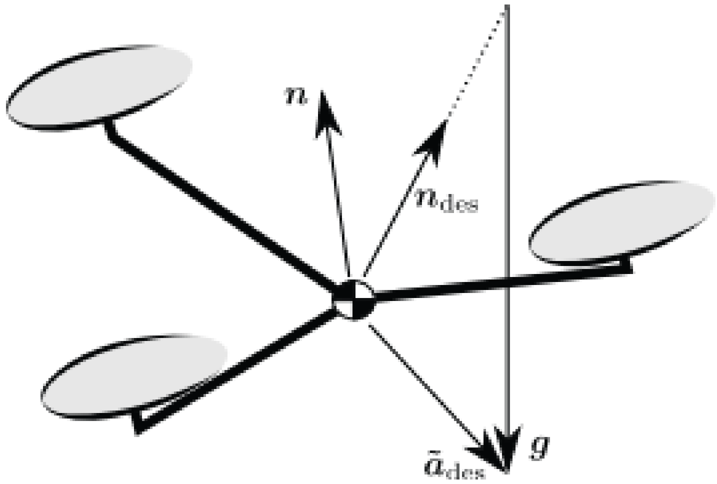

Figure 3.3: A graphical representation of the position control by means of aligning the primary axis $\mathbf{n}^B$ with the desired primary axis $\mathbf{n}^B_{des}$ [29]

direction of gravity at all times through application of a constant rotation around this vector, the position of the quadrotor can be kept approximately constant, despite non zero angular rates and non-zero linear accelerations in the body frame. Mueller and d'Andrea describe an elaborate methodology for the computation of a relaxed hover solution, which specifies a constant primary axis, $\mathbf{n}^B$, rotational rate setpoints, Ω and a thrust setting.[28][29]

In more recent times, Lu and van Kampen [19], as well as Sun and de Visser expanded on this concept with the development of non-linear controllers that negate the need for a predefined periodic hover solution. [42] Additionally, the use of these non-linear controllers enables higher speed flight, due to their inherent robustness against aerodynamic disturbances, which are difficult to model due to their highly non-linear behaviour. [42]

3.2.2. Fault-tolerant Control Architecture

A quadrotor fault tolerant control system generally consists of a cascaded loop structure, with an outer position loop, and an inner loop performing reduced attitude control and control allocation to the rotors. A generalized representation of such a loop structure can be seen in fig. 3.4. First, the position control method is discussed in section 3.2.3, followed by a discussion of the control of the thrust direction by means of reduced attitude control. Finally, the computation of the rotor thrusts by means of control allocation methodologies is discussed.

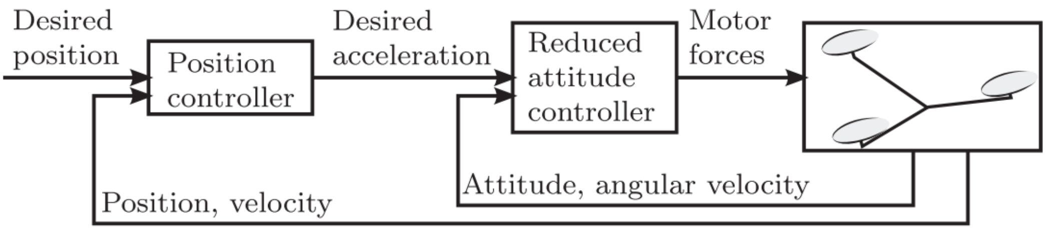

Figure 3.4: Schematic representation of a commonly used fault tolerant quadrotor controller [29]

3.2.3. Position Control

Commonly, quadrotor fault-tolerant position control is performed by means of a PID controller. In essence, the outer loop of a fault-tolerant control system is very similar to the outer loop position and velocity controllers for nominal quadrotor control, as proposed by Mahony et al. in [23] and El-Badawy et al. in [5]. The main difference being that a nominal control system generally outputs a pitch, roll and yaw angle target, combined with a desired thrust vector. In the case of a fault-tolerant controller, the outputs of the outer loop are a desired thrust magnitude, T_{des}, and an acceleration vector in inertial space, $\boldsymbol{a}_{des}$. The acceleration vector is converted to a desired pointing direction $\boldsymbol{n^B}_{des}$ in the body frame, which is subsequently fed as a reference input to the reduced attitude controller. Equation (3.17) describes the PID position controller in the outer loop, in which $\boldsymbol{\xi}$ is the position, v is the vehicle velocity in the inertial frame and $\boldsymbol{a}_{des}$ is the desired acceleration vector in the inertial space. Additionally, k_p is the gain on the position error, k_d is the gain on the velocity error, whereas k_i is an integral gain that is used to compensate the offset due to aerodynamic effects, such as induced drag. This outer loop is taken from research by S.Sun et al [42], but very similar approaches have been adopted in [44] [2] [19].

To prevent pushing the quadrotor into upsets, S.Sun and M. Baert [42] [44] have defined limits on the offset angle between the desired acceleration vector and the inertial fixed z^I axis. Additionally, the altitude control is scaled, dependent on this total offset angle, to prevent saturating the rotors. For larger offsets from the inertial z^I axis, the control over the altitude is ceded, to prevent excessive thrust commands that lead to saturations and degradation of tracking performance. While these constraints are important elements in the fault-tolerant control, it is assumed that this will not be required in this work, considering that a trajectory generation algorithm should avoid providing infeasible acceleration demands that may cause upsets.

$$\begin{gathered}\boldsymbol{a}_{\text{des}} = k_p\left(\boldsymbol{\xi}_{\text{des}} - \boldsymbol{\xi}\right) + k_d\left(\dot{\boldsymbol{\xi}}_{\text{des}} - \boldsymbol{v}\right) + k_i \int\left(\boldsymbol{\xi}_{\text{des}} - \boldsymbol{\xi}\right) dt \\ \left(k_p, k_d, k_i > 0\right)\end{gathered} \tag{3.17}$$

Based on the acceleration demand, a desired primary axis, $\mathbf{n}^B_{des}$, is computed using eq. (3.18) and eq. (3.19), which is graphically represented in eq. (3.18).

$$n^I_{des} = \frac{a_{des} - g}{\|a_{des} - g\|} \tag{3.18}$$

$$n^B_{des} = L_{BI} n^I_{des} \tag{3.19}$$

3.2.4. Reduced Attitude Control

Given the demanded desired primary axis vector in the body frame, the primary axis control is tasked with making n^B track the desired n^B_{des}. For this, an a priori determined n^B, which is fixed in the body axis must be determined. Using the framework supplied by Mueller and d'Andrea, a body axis $\mathbf{n}^B$ can be selected that minimizes rotor velocities and thus energy consumption in the relaxed hover condition. [29] For reduced attitude control, there are a few different possible solutions. Effective reduced attitude control methodologies have been proposed by Lu [19], Sun [42][44][45] and Brescianini [3]. Each of these methods can provide global reduced attitude stabilization and has been proven experimentally. For the purpose of this research, however, the INDI based methodologies developed by Sun [42][44] seem the most promising, because of their inherent robustness to complex aerodynamic effects. In 2018, first a two loop cascaded controller was defined, with an outer loop NDI controller and an inner loop Incremental Non-linear Dynamic Inversion (INDI) controller for control allocation.[42]

The outer loop equation is given in eq. (3.20), in which p_{des}, q_{des} are the desired pitch and roll rates, h_1, h_2 and h_3 are the desired primary axis components in the x^B, y^B and z^B directions and $\hat{\dot{\mathbf{n}}}^I_{des}$ is the inertial frame derivative of the desired primary axis, transformed to the body frame using the $\mathbf{L}_{BI}$ rotation matrix.

$$\begin{bmatrix} p_{\text{des}} \\ q_{\text{des}} \end{bmatrix} = \begin{bmatrix} 0 & 1/h_3 \\ -1/h_3 & 0 \end{bmatrix}\left(\nu_{\text{out}} - \begin{bmatrix} h_2 \\ -h_1 \end{bmatrix} r - \hat{\dot{n}}^I_{\text{des}}\right) \tag{3.20}$$

The inner loop computes the desired rotor speed increments when given the desired pitch and roll rates. Since this INDI method makes use of the measured and filtered outputs, $\dot{\boldsymbol{y}}_{\boldsymbol{f}}$, the controller is inherently very robust towards the highly nonlinear aerodynamic effects, modelling errors and faulty measurements. [42][45]. Equation (3.21) shows the taylor series expansion of the inner loop dynamics, while eq. (3.22) and eq. (3.23) represent the controller that is based on these dynamics. For this control allocation method, the pitch and roll accelerations are required, as represented by the $\dot{\boldsymbol{y}}_{\boldsymbol{f}}$ in eq. (3.22). This can be done by computation of the numerical derivative of the pitch and roll rates, p and q and filtering these using a low-pass filter to remove high frequency noise. The input measurement, $\boldsymbol{u_0}$, is also filtered using the same low pass filter to compensate for the filter induced lag.

$$\dot{\boldsymbol{y}} = \dot{\boldsymbol{y}}_0 + \hat{\boldsymbol{G}}\Delta\boldsymbol{u} + \boldsymbol{\epsilon} = \begin{bmatrix} \dot{p}_0 & \dot{q}_0 & \bar{f}^B_{z,0} \end{bmatrix}^T + \begin{bmatrix} b\kappa_0/I_{vx} & -b\kappa_0/I_{vx} & -b\kappa_0/I_{vx} \\ l\kappa_0/I_{vy} & l\kappa_0/I_{vy} & -l\kappa_0/I_{vy} \\ -\kappa_0/m & -\kappa_0/m & -\kappa_0/m \end{bmatrix} (\boldsymbol{u} - \boldsymbol{u}_0) + \boldsymbol{\epsilon} \tag{3.21}$$

$$\boldsymbol{u} = \hat{\boldsymbol{G}}^{-1}\left(\boldsymbol{\nu}_{\text{in}} - \dot{\boldsymbol{y}}_f\right) + \boldsymbol{u_f} \tag{3.22}$$

To add to the robustness, control allocation can additionally be used to improve the performance of the controller near the saturation limits of the rotors. When flying at high speeds or performing fast rotation, actuator saturation may occur for the rotors. Control allocation, for example quadratic programming or P2 allocation, can be used to optimise the thrust allocation to each rotor. Additionally, this allows for adding constraints to the inputs to prevent creating a unsuppressable rotations. [2][44][3].

$$\boldsymbol{\nu}_{\text{in}} = \begin{bmatrix} \dot{p}_{des} + k_1\left(p_{des} - p\right) \\ \dot{q}_{des} + k_2\left(q_{des} - q\right) \\ \bar{f}^B_{z,des} + k_3 \int \left(\bar{f}^B_{z,des} - \bar{f}^B_z\right) dt \end{bmatrix} \tag{3.23}$$

3.3. Trajectory Generation

To enable safe and accurate aggressive maneuvering, accurate motion planning capabilities are required. The objective of these is to generate feasible position and attitude trajectories for a quadrotor with a single rotor failure that meet the constraints on both the inputs and quadrotor dynamics. In literature, motion planning is often separated into two different disciplines. The first being the generation of a geometrically feasible path through obstacles, the second being the generation of a trajectory along a defined path that is dynamically feasible given the vehicle dynamic properties. The primary objective of this research is to define a method that allows the generation of dynamically feasible trajectories, to enable aggressive maneuvering and upset recovery. While obstacle avoidance is an interesting aspect that can be incorporated, it is initially a secondary objective in this research effort. The goal is to develop a method that is capable of characterising an aggressive maneuver, which is dynamically feasible, to guarantee the safety of the maneuver and prevent loss of control.

In literature, a multitude of solutions has been proposed for the problem of feasible dynamic trajectories for fully functioning quadrotors. To the best of the author's knowledge, no attempt has been made to generate trajectories for quadrotors suffering from rotor damage. Therefore, a clear gap exists in the design of feasible trajectories for quadrotors, despite the presence of rotor failure. To be able to overcome this gap, adaptation of existing strategies for quadrotor motion planning will be investigated.

3.3.1. Challenges

Trajectory generation for quadrotors, however, is a non-trivial subject for nominal quadrotors, that is further complicated by the loss of a rotor. Over the course of the literature study phase, a number of key challenges has been identified that will need to be solved. The main challenge encountered in motion planning is the computational cost of generating feasible trajectories. Generally speaking, motion planning is a high dimensional complex problem, for which finding optimal solutions is computationally expensive. This is in contrast with the limited on-board computational capabilities of common quadrotors, which demands algorithms of low computational cost. [33] [26][16] Putting additional strain on this challenge is the fact that real-time capabilities are required to be able to achieve the goal of generating upset recovery trajectories. Additionally, the capability of starting at arbitrary initial conditions is required, which adds additional complexity to the problem. [10][27][16] The second challenge is the fact that the quadrotor with fixed pitch rotors is inherently underactuated, as its attitude and position cannot be independently controlled, even without the presence of a rotor failure. The presence of a rotor failure limits the safe flight envelope on the quadrotor, imposing additional constraints on its dynamic capabilities. Overcoming this problem requires a good characterization of the dynamic limits on the quadrotor upon rotor failure, which has to be translated to proper constraints on the trajectory generation. [25] [32] [27]

Thirdly, the complex aerodynamic effects as highlighted in section 3.1.2 are hard to estimate and modelling them in the motion planning would put additional strain on the limited on-board computational power. [30]

The literature on motion planning has therefore been assessed primarily on how the solution complexity is reduced and how feasibility is guaranteed within the solution.

3.3.2. Parametrized Maneuver Sequences

A common simplification method is is to parametrize the trajectory and discretizing an aggressive maneuver as a sequence of input or control segments. These so-called motion primitive sequences allow for performing aggressive maneuvers such as flips or loopings, albeit at sub-optimal conditions to reduce computational complexity.

El-Badawy et al propose an energy based optimisation method which is designed to eliminate the singularities of looping trajectories. [5] Lupashin et al propose an open loop aggressive maneuvering method that employs an iterative learning strategy to optimise the input parameters to perform a flips or a sequence of flips. In this strategy, the flip is characterized as a sequence of bang-bang control inputs on the edge of the control envelope to ensure near optimality. The maneuvers are optimised by minimization of the error on the end-states as opposed to the full trajectory to reduce computational effort. [22]. Finally, Gillula et al [9] propose a method to decompose a complete maneuver as a set of discrete, simplified hybrid modes that are guaranteed to be feasible by means of computing the safe reachable sets of the quadrotor. These three methods are very useful for performaning maneuvers at low computational complexity, however, these strategies make use of simplified 2D-models, which assume accurate stabilization and tracking of the out-of-plane dynamics, which is invalid in the case of the spinning damaged quadrotor.[20]

The methods employed in the aforementioned research are extended into three dimensional motion by Mellinger et al [26] and Lupashin et al. [20][21]. Mellinger proposes the decomposition of a trajectory into three different controller modes that can be seen as individual hybrid system modes. These modes are attitude control, hover control and 3D-path following, which are separately tuned to guarantee stability for their different operational modes. Switching between modes can be performed either by setting a specific maneuver time in advance, or event-based, in which switching occurs when a certain state is reached. [25][26] Lupashin employs the same parametrization and optimization methods as mentioned before, but extends these into 3D-space. [20][21]

While these methods are useful background information and provide key knowledge into the design of trajectories, the usefulness of such methods is limited, owing to the limited real-time trajectory generation capability. The approaches require parametrization and optimization of the parameters of the maneuver offline in advance off-board of the quadrotor. Additionally, the bang-bang control strategy is difficult to implement, because the in-plane rotational limits of the quadrotor vary with the yaw angle. While the hybrid decomposition of the research by Gillula is interesting, application of such methods on a 3D-model of a damaged quadrotor is infeasible, in terms of both computational capabilities and research scope.

3.3.3. Differential Flatness

A more effective method of reducing the computational cost of trajectory planning can be achieved through the simplification of the quadrotor dynamic model by means of its differential flatness property. This means that all of the outputs of the conventional full state model can be described by means of a smaller selection of outputs, called the flat outputs. This reduces the dimensionality of the complex and large full state model and simplifies the dynamics to a double, triple or quadruple integrator system.[14] Constraints can be placed on the flat outputs and their derivatives, Due to the flat output representation, constraining the derivatives of the flat outputs also implicitly constrains the remaining states [10][25]. This representation greatly reduces the dimensionality of the optimization problem and reduces its computational cost. A comprehensive method for deriving a flat output representation for a dynamic system is provided by Rigatos. [35] This could potentially be used to derive a differentially flat output representation for a damaged quadrotor, in case the common models are not found suitable in a later stage.

Given the differentially flat model, a trajectory can then be optimized by means of minimizing the derivatives of acceleration, such as jerk or snap, possibly combined with the total flight time. [10][33][25] The non-convex optimization is generally solved using quadratic programming methods.[25][16][6] The output of this optimization is then a position, velocity, acceleration and jerk and/or snap profile, that can be parametrized using polynomials [10][25][27][30] or splines. [6] [15] [8] The differential flatness approach also allows the adoption of model predictive control strategies, in which the trajectory is continuously updated by means of optimal control methods to compensate for errors in the tracking of the trajectory. [27][12][10]. Optimization can be performed based on optimal control methods, for example the Pontriyagin's minimum principle based optimization performed by Mueller et al [27] and Hehn et al[10]. A more detailed background on this type of optimal control based dynamic programming method is provided by Smith et al. [37]

More advanced methods even include the possibility of obstacle avoidance and the navigation of safe corridors, combining pathfinding through obstacle fields with the generation of feasible trajectories. [33][16][12]

These methods combine decision tree machine learning methods for pathfinding with steering functions based on differentially flat models to verify the dynamic feasibility. [12][33]

A first type of differentially flat model is provided by Mellinger et al., who devised a differentially flat fourth order integrator model to represent the quadrotor dynamics[25]. In this model, the vehicle motion is described using four flat outputs, the x-,y- and z-position and the yaw angle, ψ. Mellinger defines keyframes along the trajectory that the quadrotor moves through which each specify a position and a yaw angle. An optimization algorithm minimizes the trajectory snap and yaw rotation to generate a trajectory that ensures a smooth transition between these keyframes.[25] This differentially flat representation has subsequently used by many others to reduce model complexity and computational cost. [32] [6][15] This differential flatness based approach is especially useful for the use of camera vision based guidance, since constraining the yaw angle makes it possible to track aggressive trajectories while keeping the camera pointed in a direction of interest.[38][18][50][12]

While a very potent real-time method, the problem is the fact that this optimization assumes accurate yaw angle tracking, which is impossible upon a rotor failure. However, possibly, adaptations to both the objective function, as well as the differentially flat model using the theory provided by Rigatos [35] could potentially allow the use of this type of algorithm. The advantage of this would be that the use of a fourth order model enables constraining the angular acceleration or input torques over the trajectory,[12][25] which means the limits on the attainable moment set can be incorporated directly into the optimization.

A more promising concept is found in the generation of minimum jerk trajectories. First, real-time methods were developed that allow trajectory generation from arbitrary initial conditions that are verifiably feasible by Mueller et al [30] [27] and Hehn et al. [10] In this research, the quadrotor dynamics are characterized by a third order integrator system, after which a trajectory is generated through the minimization of either the jerk and flight time or both.

Generally speaking, the minimum jerk trajectory problem is described by eq. (3.24). The objective function is designed to minimize the integral of the jerk over the trajectory, possibly in addition to the total travel time. The initial conditions and final conditions are specified for the position, velocity and accelerations, which are allowed to be nonzero. The inputs on the system are the total thrust $\overline{f}^{\boldsymbol{B}}$ and the pitch and roll rates, p and q. The yaw rate, r is a design parameter that can be set by the algorithm designer at any arbitrary, constant value, which makes this method very compatible with the fault-tolerant control strategy from section 3.2. Constraints are set on the trajectory velocity, acceleration and jerk.[10] [16] [27]. By employing the constraints on the acceleration and jerk, implicit constraints are placed on the rotational rates of the vehicle as well, as proven by Mueller [27] and Hehn [10]. This is particularly useful, because using a well designed constraint on trajectory jerk could potentially prevent pushing the quadrotor into unsuppressable rotational velocities, defined in section 3.1.3.

$$
\begin{aligned}
&\min_{\mathbf{j}(t),T,t\in[0,T]} \left\{ J = \int_{t=0}^{T} m + \mathbf{j}(t)^2 dt \right\} \text{ s.t.}\\
&\mathbf{p}(0) = \mathbf{p}_{ini}, \quad \mathbf{p}(T) = \mathbf{p}_{\text{ref}}\\
&\mathbf{v}(0) = \mathbf{v}_{\text{ini}}, \quad \mathbf{v}(T) = \mathbf{v}_{\text{ref}}\\
&\mathbf{a}(0) = \mathbf{a}_{\text{ini}}, \quad \mathbf{a}(T) = \mathbf{a}_{\text{ref}}\\
&\dot{\mathbf{p}}(t) = \mathbf{v}(t)\\
&\dot{\mathbf{v}}(t) = \mathbf{a}(t)\\
&\dot{\mathbf{a}}(t) = \mathbf{j}(t)\\
&-\mathbf{v}_{\max} \le \mathbf{v}(t) \le \mathbf{v}_{\max}, \quad \forall t \in [0,T]\\
&-\mathbf{a}_{\max} \le \mathbf{a}(t) \le \mathbf{a}_{\max}, \quad \forall t \in [0,T]\\
&-\mathbf{j}_{\max} \le \mathbf{j}(t) \le \mathbf{j}_{\max}, \quad \forall t \in [0,T]
\end{aligned}
\tag{3.24}
$$

These described methods for jerk limited trajectories treat the subject of generating dynamically feasible trajectories between two points or a sequence of points. To add to the potential of this trajectory generation method, more modern methods incorporate obstacle avoidance and safe corridor navigation, as performed by Lan [16] and Lai [33]. The method by Lai makes use of pre-defined safe flight corridors and defines a trajectory in which position constraints within this safe corridor are enforced to prevent collisions with obstacles outside of these corridors. With predefined safe corridors, this method is capable of operating in real-time on board quadrotors, given an outer loop operating frequency between 20-50 Hz. [33] Lan et al define a method combines the capability of collision avoidance and the generation of dynamically feasible trajectories, aided by machine learning techniques. A safe path through obstacles is generated using a Batch Informed Tree (BIT^*), whereas a pre-trained Multilayer Perceptron (MLP) neural network is used to boost the convergence rate of a two-point boundary value optimization that is used as a steering function that propagates the quadrotor state through the envisioned trajectory and verifies the dynamic feasibility. Although this method is nearing fast real-time capability, some computation time is required to generate a trajectory. [16].

3.3.4. Machine Learning Based Methods

Finally, methods exist based on machine learning methods that (partially) take away the dependence on accurate dynamic models. While machine learning is employed in aforementioned methods, those models do incorporate a detailed dynamic model of the quadrotor. The models discussed here treat the quadrotor dynamics in a more grey-box or black box approach, in which a machine learning algorithm generates its own model of the quadrotor dynamics.

First of all, methods exist based on apprenticeship learning, in which a human pilot is tasked with performing maneuvers, which are subsequently used to train reinforcement learning algorithms, with the intention of replicating the maneuver and improving it. This strategy is employed by both Tang et al [48] and Abbeel et al[1]. While interesting approaches, this methodology is infeasible for the purposes envisioned in this research, because it would require an expert pilot to provide an excessively large set of pre-learned maneuvers for use in real-time applications.

More recently, neural network based networks have been defined that are capable of generating real-time feasible trajectories, often called *G&CNets*. Camci et al [4], for example, propose a strategy in which reinforcement learning is trained in simulation to build a large library of motion primitives. A library of up to 30,000 motion primitives is generated offline in simulation, which can subsequently be combined to generate trajectories in real time. The model inputs, in the form of pitch, roll and yaw rates are characterised using Bézier curves that mimic expert pilot inputs. During the trajectory generation phase, a variable called the confidence level is introduced. When generating a trajectory, this confidence level denotes whether the algorithm is confident whether it can execute the desired trajectory. In case the confidence level is 0, the trajectory is not executed and a conventional trajectory generation algorithm, such as defined by Mellinger or Hehn, has to be employed to generate a feasible trajectory. [4]

Tang et al propose a similar methodology, using a neural network to learn how to map parameters onto optimal trajectories by means of an offline database of maneuvers. This method can generate real-time predicted trajectories online in milliseconds, which are then refined by a quadratic programming algorithm. [47]

The advantage of using such methods is the fact that the reinforcement learning algorithm is capable of learning the limitation on the quadrotor dynamics without complex constraint computation. The downside of this type of method is the use of a black-box model, which is inherently not explainable, making it hard to predict the feasiblity of this approach up front. Additionally, a large library of motions is required to be able to use this methodology real time, which is possibly prohibitive.

3.4. Literature Conclusions

What became apparent over the course of the literature study is that a definite gap exists in the generation of feasible trajectories for quadrotors experiencing quadrotors. The developments in fault tolerant control methods currently enable the recovery from attitude upsets and mission continuation at high speed despite the failure of a rotor. A next possible step in expanding the mission capabilities of damaged quadrotors would therefore be to enable aggressive maneuvers for these damaged quadrotors such as obstacle avoidance at higher speeds through cluttered environments, perching on surfaces or performing flips. This could be done through the use of online trajectory generation methods, which to the best of the author's knowledge has not been performed before. Additionally, these methods could be used to improve the performance of current attitude recovery strategies in case of attitude upsets.

A multitude of methods has been reviewed during the literature study and have been assessed on their applicability to this specific problem. Each of these methods provides a different approach to reducing the dimensionality and computational complexity of the highly dimensional motion planning problem and allows the planning of feasible trajectories. The methods considered here are parametrized maneuver sequences, minimum snap trajectories, minimum jerk trajectories, apprenticeship learning methods and G&CNets.

Three main challenges were identified in section 3.3.1 that are critical to the success of a trajectory generation algorithm for the intended purposes. First of all, the solution must be of sufficiently low computational cost that it can operate in real-time, either using on-board equipment or off-board computing power. Secondly, it must be capable of starting from arbitrary initial conditions, to enable attitude upset recovery. Finally, the solution must take into account the dynamic constraints and coupling effects that occur due to the occurrance of a rotor loss. To assess the usefulness of each of the solutions, trade-off is performed, using criteria based on these three challenges. This trade-off is used to identify the most appropriate solution methodology, which is to be used in the further stages of development, which will be further elaborated upon in chapter 5. Five criteria have been devised to assess the usefulness of each solution, being:

1. Real-time capabilities
2. Arbitrary Initial Conditions
3. Dynamic Feasibility
4. Solution Complexity
5. Adaptability

For each criterion, scores are awarded between 1 and 5, with each criterion weighted equally. The solution with the highest total score is selected as the most feasible solution to the defined problem. The first criterion concerns the ability to use a method to generate a trajectory for a maneuver in real-time, which is specifically critical for upset recovery. The second criterion considers the ability to use a method under arbitrary initial conditions, which is also important for use in case of an upset. The third criterion concerns the capability of incorporating constraints on the trajectory to guarantee feasibility of the trajectory, despite the performance impairments due to the rotor failure. The fourth criterion concerns the difficulty of developing existing methods into a a functional algorithm for a quadrotor with rotor failure, with a higher score awarded for lower complexity solutions. The final criterion concerns the ability to adapt to different circumstances, which primarily concerns the ability to perform unique maneuvers, such as sequences of different maneuvers to avoid various obstacles. A low score is awarded to a method that only allows a very limited set of maneuvers, whereas a high score is awarded to methods that can generate a large variety of arbitrary maneuvers. The results of the trade-off can be seen in fig. 3.5.

The first method, the parametrized sequence of maneuvers, elaborated upon in section 3.3.2, has a low score for real-time capabilities, arbitrary initial conditions and adaptability. This is because this type of solution generally depends on a-priori optimization of a predefined maneuver trajectory with known initial conditions that makes it impossible to generate trajectories in real time and adapt them to different circumstances. [20] [21] Generally, this type of solution is limited to a specific kind of maneuver, such as loopings or flips and cannot be used to execute any other.[5] While dynamic constraints can be set effectively for nominal quadrotors, these constraints often require stable in-plane dynamics and are likely hard to implement on quadrotors with rotor loss. [22][5] Also, in terms of solution complexity, this type does not score very high. While 2D-maneuvers are of relatively low complexity [5][22], the 3D-implementation quickly gets more complex. [20][25].

	Real Time Capability	Arbitrary Initial Conditions	Dynamic Feasibility	Solution Complexity	Adaptability	Total Score
Parametrized Maneuver Sequence	1	1	3	3	1	9
Minimum Snap Trajectory	5	5	2	2	5	19
Minimum Jerk Trajectory	5	5	4	5	5	24
Apprenticeship Learning	1	1	1	3	1	7
G&CNets	4	5	3	3	3	18

Figure 3.5: The trade-off scores for each of the possible methods, with a color scale going from red for a low score to green for a higher score.

The second solution of minimum snap trajectory generation is a very promising one. Real-time, online solutions are available that can generate trajectories from arbitrary initial conditions within miliseconds, giving it the maximum score on these criteria. [6][12] Additionally, effective constraints can be set on the angular acceleration demands, making it very compatible for damaged quadrotors, which have a strong limitation on the Attainable Moment Set. However, the dynamic constraints for feasibility generally require a constraint on the yaw angle, which are infeasible for quadrotors with rotor failure, giving it a low score in this respect. [25][12][32] At first sight, this issue make this solution less suitable for this problem, unless a solution can be found in literature. This does however mean that complex adjustments to the dynamic model are required, which makes it a relatively complex solution to develop, giving it a low score in solution complexity. Its adaptability is, however, very high, as this method can be used to generate complex sequences of unique maneuvers. [25]

Similarly to minimum snap trajectories, minimum jerk trajectories score very high on their real-time capabilities, the ability to operate from arbitrary conditions and their adaptability. [27][30] The dynamic constraints on minimum jerk trajectories are very effective, however, they do not allow for constraints on the angular accelerations. Instead, constraints can be set on angular velocities to prevent unsuppressable rotations, which is expected to be less effective than constraining the moment set.[10] In terms of solution complexity, the minimum jerk trajectory is very promising, since very few alterations are required to the dynamic models of the current state of the art. This is primarily due to independence of the solution from the yaw angle, which makes it possible to select a desired yaw rate setpoint. This setpoint can be computed as part of a desired relaxed hover solution. [10] Additionally, they are highly adaptable, and can be used to generate complex maneuvers through obstacle cluttered environments. [16][15].

The apprenticeship learning concepts, such as those by Abbeel et al. [1], are not a very realistic solution to this problem, considering their limited real-time capabilities, inability to start at non-zero initial conditions and their limitation to specific pre-learned maneuvers. This is reflected by very low scores on most criteria.

Finally, the G&CNets are considered. These solutions are expected to operate real-time, but when insufficient confidence exists in the model-free solution, a failsafe conventional solution is required, which reduces the reliability of their real-time performance. Operation from arbitrary initial conditions is possible, giving it full score in that category.[4][47] While a potent method, the limited explainability of the solution, owing to the black-box nature of this approach, leads to lower scores in the final three criteria. First of all, it is difficult to guarantee that the system will always converge to a dynamically feasible solution, because of this lack of explainability. Additionally, which it is difficult to predict the development complexity a priori because of this black-box nature. Finally, the adaptability to different types of maneuvers is limited by the size of the maneuver database. The more complicated a sequence of maneuvers becomes, the larger the likelihood that the algorithm is incapable of finding a feasible solution with sufficient confidence. [47]

What can be seen from the results, is that the minimum jerk trajectory generation method has the greatest potential for further research into the identified gap in the literature. Especially the lower envisioned complexity for the development of this algorithm makes it the most promising for the next stages of the research. Minimum snap trajectories are similarly promising, but the envisioned difficulties implementing an effective differentially flat model without constraining the yaw angle makes it less feasible. The G&CNet solutions are also very interesting for future development, but their usefulness is primarily limited due to the fact that they as of now cannot operate as a standalone solution and require a back-up algorithm as a failsafe. In later research, these solutions can most likely be combined to increase the potential of the algorithm.

4

Research Goal and Scope

Following the establishment of research questions and the identification of a gap in the literature, the next step in the development of the fault-tolerant control system is to establish the goal of the research and define a clear scope, in order to properly focus the research and design efforts. This chapter establishes the research goal and envisioned final product and additionally limits the development scope.

4.1. Research Objective

The goal for this research is to design an online trajectory generation algorithm that is capable of generating feasible trajectories that lie within an approximation of the safe flight envelope of a quadrotor with single rotor failure. With this trajectory generation, the quadrotor is to be able to perform aggressive maneuvers that lie far from the relaxed hover trim conditions. Examples of such maneuvers are high speed flight through a window at large attitude angles or perching on designated surfaces as defined by Mueller et al. [26], (multi-)flips, as proposed by [5], [22][49] or higher speed flight through an obstacle-cluttered area, as defined by Lan et al and Lai et al.[33][16] A second objective is to perform recovery from a loss-of-control situation at large upset angles and arbitrary initial velocity and rotational rates, as defined by Sun et al. [44].

This system has a few key requirements before it can be deemed succesful. These key requirements will be verified and validated over the course of the simulation and testing phases as explained in section 5.5. Currently, the requirements mention terms as 'aggressive maneuvers at large deviations from hover condition', which do not specify quantified performance metrics. Over the course of the study, these functional key requirements are to be expanded with more exact performance requirements, as soon as more information on the design specifications are available. They are included now to provide an overview of the most important research goals. The requirements are as follows:

- **REQ-KEY-FUNC01-**The designed algorithm shall enable the generation of trajectories aggressive maneuvers at high speeds and large deviations from hover condition.
- **REQ-KEY-FUNC02-**The designed algorithm shall be able to start at arbitrary initial velocities, attitude angles and attitude rates.
- **REQ-KEY-FUNC03-**The designed algorithm shall generate trajectories in real time on a Parrot Bebop2 quadrotor platform.
- **REQ-KEY-FUNC04-**The designed algorithm shall generate trajectories that are verifiably feasible.

4.2. Research Scope

To limit the research effort to a realistic scope, a few assumptions have been made it advance. First of all, the existence of functional active fault tolerant control is assumed. At the Control & Simulation, effective fault-tolerant control method designed by Lu et al [19] and Sun et al [41] [45] exist that can serve as a strong basis for further research. Secondly, it is also assumed that effective fault-detection is available, so that the rotor failure occurrence is detected and isolated before the simulations and experiments. Effective methods have been treated in the literature by Lu et al.[19] Thirdly, the assumption is made that effective and accurate state estimation is available during flight. Currently, state estimation using on-board sensors is insufficiently accurate for trajectory tracking with quadrotors, which means that external equipment has to be used to provide accurate estimation,

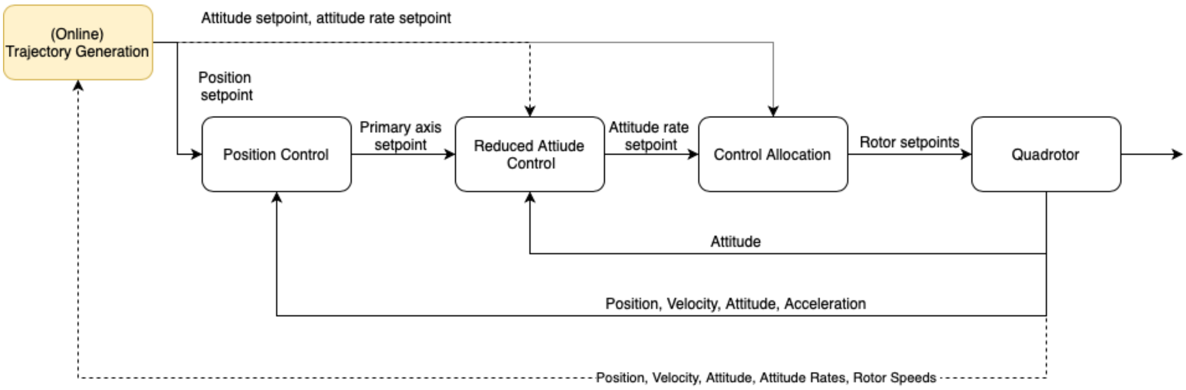

Figure 4.1: A systematic block diagram of the envisioned trajectory guidance loop.

as discussed elaborately in section 5.6, It is expected that in the future, vision based methods will be capable of providing sufficiently accurate state estimation methods, as defined by Sun. [46]. Finally, it is assumed that effective obstacle recognition is available, as including obstacle recognition in the scope would be too demanding for the duration of this thesis project. Similarly to the assumption regarding state estimation, it is expected that vision based state estimation for spinning MAVs will be able to provide obstacle recognition in the future. [46]

Given these assumptions and the research goal, fig. 4.1 provides an overview of the envisioned solution. An outer guidance trajectory generation algorithm is envisioned that generates a feasible trajectory for a quadrotor with single rotor failure. This trajectory generation algorithm provides inputs to a fault tolerant controller, as described in section 3.2. Depending on the selected trajectory generation method, this controller can be updated in real time and adjust the trajectory, to compensate for tracking errors due unmodelled dynamics.

5

Thesis Approach

In this section, the approach and methodology of the thesis is discussed. The approach has been encompassed into four different work packages, being design, development, implementation and finalization, which are discussed in section 5.1, section 5.2, section 5.3 and section 5.4. The work breakdown structure can be seen in fig. 5.1, in chapter 6 this work breakdown structure is used to devise a planning for the thesis. Additionally, the important verification and validation strategies are discussed in section 5.5. Finally, section 5.6 treats the experiment set-ups that will be used to assess the designed trajectory generation method.

5.1. Work Package 1: Design

The first step in the thesis process is to design the guidance algorithm that has been described in chapter 4. The goal of this phase is to generate a theoretical design for a trajectory generation method, which is to be developed and implemented in the subsequent phases. The main challenge is to make a design choice for the trajectory generation method, with the goal of devising a method to reduce the complexity of the optimisation problem, choose a method of parametrizing the trajectories and to integrate the solution into the existing fault-tolerant control methods. This approach consists of three important steps that are done in conjunction, being modeling, trajectory design and the control loop integration.

5.1.1. Modeling

The modelling step is concerned with the development of a dynamic model of reduced dimensionality for use in the optimisation. A good example of this are the differentially flat models as described in section 3.3 for use in minimum snap or minimum jerk trajectories. The goal of this is to reduce the complexity and dimensionality in the dynamic model to reduce the computational cost of the optimization.

Alongside this development, a characterisation of the safe flight envelope is to be made, to be used in the design of the trajectory generation algorithm to guarantee feasibility. In this step these limitations are characterised, so that they can be used in the generation of constraints for the trajectory generation algorithm. The exact method for pursuing this goal is still to be developed and also dependent on the modeling strategy that is employed, but could be done either theoretically or with a series of experiments. Section 3.1.3 provides a good starting point for this research.

5.1.2. Trajectory Generation

For the generation of trajectories, first a method is to be developed to parametrize a trajectory in such a way that it can be used as input to the control algorithms, in the form of for example, position, acceleration, attitude and attitude rate setpoints. This can for example be done by defining a discretized sequence of constant inputs, or a sequence of dynamic motion primitives characterized by a desired acceleration, jerk or snap profile. In this step, a method is to additionally be developed for obstacle avoidance or flight through safe corridors. This is done in conjunction with the generation of an optimisation algorithm that optimises the trajectory. This consists of the definition of an objective function, selection of initial conditions and the generation of trajectory constraints. The objective function can be chosen to minimize various parameters, such as total flight time, trajectory jerk, trajectory snap or total energy. The constraints can include limits on position, velocity, acceleration, jerk, snap, rotational rates and rotational accelerations. The final step is to define an optimisation method to solve the

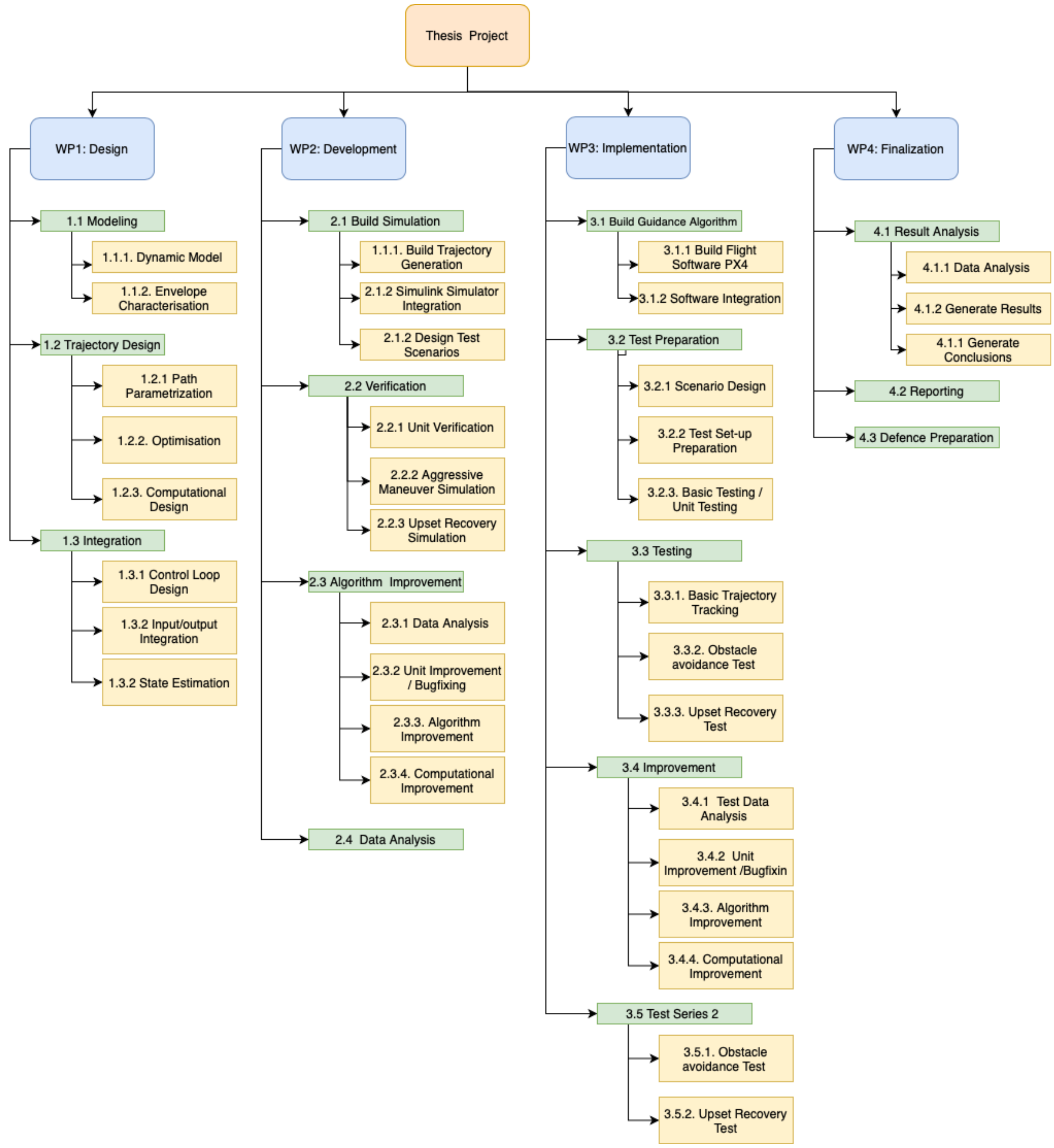

Figure 5.1: Work Breakdown Structure of the thesis research.

optimisation problem. In this step, it is critical to define a method that is computationally efficient to guarantee the real-time capabilities of the algorithm. Therefore, the computational efficiency of the solution is thoroughly investigated in this step.

5.1.3. Integration

The final step in the design phase is to integrate the outer loop guidance algorithm into a fault-tolerant control system. While this could potentially be a relatively simple integration, potentially design changes have to be made to the current state-of-the-art control systems. It is highly expected that a change in the outer position control loop is required to accommodate real-time trajectory generation, which generally outputs acceleration targets. Additionally, it is important to define an appropriate operating frequency for the guidance loop that is computationally feasible and leads to stable behaviour from the controller. [27] [10]

Finally, it is important to integrate the guidance algorithm with the state estimation capabilities. The combination of the guidance and control system has to be sufficiently robust to cope with state estimation errors and discrete operating frequencies of the measurements.

5.2. Work Package 2: Development

Following the design of the algorithm, the algorithm software is developed. This development is done in two stages, in which the first stage is to design a Simulink based version of the algorithm, which is tested and finetuned through a series of flight simulations. The use of simulations for this purpose allows the development of software improvements in quick succession, without having to conduct painstaking flight tests with imperfect flight software. [2]

5.2.1. Build Simulation

The first objective in this stage is to design the simulink application of the trajectory generator. Currently, a Simulink model of the Bebop quadrotor is available at the Control & Simulation department that accurately models the dynamics. Previous research efforts have developed this model to include fault-tolerant control when experiencing rotor failure, as well as a simulation model of state estimation. [19][44][45]

First, the trajectory generation algorithm is built in Simulink, after which its integrated into the existing Parrot Bebop Simulator. In this development process, a Git source control method is employed to track changes and additions to the simulator. Using Git, all users have access to the most recent release of the simulator, allowing the separate and simultaneous development of new additions, without making undesired alterations to the work of others. This is done by working on separate branches containing different new developments in progress. Once a developed code addition reaches a sufficient maturity level, it can be released into the main branch and be incorporated in the simulator. [36]

5.2.2. Simulation

Following the development of the simulation, verification of the designed algorithm is performed through a series of simulations. First of all, this verification serves to identify flaws in the design through a series of unit tests, in which the individual functionalities of the guidance and control systems are assessed separately. Using these unit tests, bugs and errors in the model are removed to ensure a properly functioning algorithm in the full scale simulations.

In these full functionality simulations, the capabilities of the algorithm are assessed, with the goal being to identify potential flaws in the design from work package 1. First, simulations are conducted to assess the performance of the model in aggressive maneuvers, as defined in section 4.1. Possible maneuvers include flips, perching or fast maneuvering through a safe corridor between an obstacle course that is designed and mapped into the algorithm in advance. A second series of tests is designed to assess the performance in recovery from upsets, as is done in the research of [44]. In this case, the quadrotor is placed in upset conditions, initialized at various initial rotational rates and upset angles to verify whether the algorithm is capable of recovering the quadrotor from these upset conditions.

5.2.3. Algorithm Improvement

Simultaneous to the simulation work, a continuous improvement process is performed to improve on the design generated in work package 1. In this package, the data from the simulations is analysed to identify possible problems with the design. Potential difficulties could include improper model structures, inaccurate performance constraint modeling, computationally heavy optimization or unstable dynamics in combination with the control system. Given these identified design flaws, the design steps from work package 1 are reiterated to improve the guidance algorithm and increase its performance, until a satisfactory performance level is achieved in the Simulink simulations.

5.3. Work Package 3: Implementation

When satisfactory performance is achieved in the simulations and the simulated algorithm is capable of achieving its objectives, a real-world application of the algorithm is built for use on a Bebop quadrotor. In an iterative process, the algorithm is tested and refined, until sufficient performance is achieved in physical tests. The goal is to either achieve the reserarch objectives or to prove that these are infeasible.

5.3.1. Code Development

The first step is to develop the simulated algorithm into functional flight software, for use on a Parrot Bebop quadrotor. The fault tolerant control software used at the Control & Simulation department runs on the PX4 framework, a widely adopted C-based flight control framework for unmanned aerial vehicles.

To fully develop this PX4 based flight software from the ground up would be a very painstaking process. To solve this difficulty, wrapper software is available that can build and compile Simulink code to function in the PX4 flight control framework. This software handles the interface between the Simulink algorithm and the PX4 flight control framework, which makes it possible to add the new features as subsystems to the flight control software.[2] Important focus in this process is to monitor the impact of using such a Simulink wrapper on the computational efficiency of the algorithm. For this preliminary investigation, this was beyond scope, but this must be investigated prior to development. The version control in this development is done using a Github framework, similar to the development of the simulation.

5.3.2. Experiment Preparation

Following the development of the flight software, physical flight tests are conducted with the trajectory generation algorithm to assess its real world performance. The primary objective of these tests is to validate the algorithm and determine whether it is capable of achieving the research objectives from section 4.1. In the experiment preparation, the experiments are designed and the performance metrics are defined for use in the assessment.

The first step in the testing phase is to design a test set-up that is suitable for validating the algorithm performance in physical tests. First, a series of test maneuvers is to be defined to assess the performance of the trajectory generation algorithm. Data from the simulations will provide an accurate indication of what is possible, so the performance from simulation will be used as a baseline for the design of experiments. To assure a smooth process during the testing weeks, the maneuvers are to be designed and tested in simulation in advance. Using this, metrics are to be defined to assess the performance of the algorithm in these flight tests. Possible maneuvers include flips, loopings or obstacle avoidance runs in a cluttered area. In the cyberzoo, obstacles can be placed at a predetermined location, which can to be mapped into the trajectory generation algorithm as unsafe locations. The goal is to perform aggressive maneuvering trajectories in the presence of obstacles that are to be avoided with rapid turns.

Following the design of the test set-up, time is devoted to preparation of the experiment location and the equipment. The main goal of this is the integration of the flight software on the test hardware and preparing the interfaces between the quadrotor and the Cyberzoo equipment, such as Optitrack and possibly a computer for off-board computation. This is followed by unit tests to verify the functioning of the basic systems and consequently the execution of simple trajectory tracking tasks to verify the algorithm functionality in a low risk scenario. More detailed explanations on the experimental set-up will be provided in section 5.6.

5.3.3. Testing

Following the experiment preparation and subsystem verification, flight tests are conducted to validate the algorithm performance in flight tests. This data is used for continuous improvement of the algorithm until sufficient performance is achieved. The desired result of this phase is to have a fully functioning guidance algorithm, but it is also possible that this phase leads to the conclusion that the objectives are infeasible.

The first task is to track basic trajectories, such as figure eight patterns, as has been performed in a most fault-tolerant control literature to verify the operational functionality of the algorithm. Following this, an aggressive maneuvering test is to be performed, in which the aggressive maneuvers that have been designed in the previous phase are executed in the Cyberzoo. Secondly, an upset recovery test is performed as done by Sun et al [44], in which the quadrotor is thrown into the air at random upset angles and varying rotational rates. In this test, it is to be assessed whether the quadrotor is capable of recovering from this loss of control situation without resulting in a ground collision.

5.3.4. Improvement

Alongside the testing, continuous algorithmic improvement is performed in an iterative cycle. The data from the initial verification and flight tests is used to improve on the algorithm design and the developed code. Initially this effort is focused on identifying errors and bugs in the developed flight software. In a later stage, when the obstacle avoidance and upset recovery experiments are conducted, the focus will shift towards making alterations to the design to improve real world performance.

The main expected difficulty is that there is a discrepancy between the performance in simulation and in real world testing. This can either be due to infidelities in the simulator approach, but can also occur due to unexpectedly high computational cost or insufficient robustness to measurement imperfections, such as noise, delay or errors.

This phase serves to identify these flaws through data analysis and is used to make corrections to the design in an effort to improve performance.

5.4. Work Package 4: Finalization

The last phase of the research is concerned with wrapping up the work that has been performed over the full research period. First, the test data is analyzed to consolidate all the findings from simulations and testing and generate satisfactory results for reporting. Using these findings and a consolidated overview of the proposed method, conclusions can be drawn on the research and its contribution to the field.

Using these results and conclusions, a scientific article is written to serve as the final deliverable of the thesis project. Alongside this, the thesis defence is prepared as the final milestone of the thesis phase.

5.5. Verification & Validation

During the project, a multitude of verification and validation stages throughout the development and implementation phases are put in place to ensure that the developed algorithm is capable of achieving the design objectives as intended on systems and subsystems level. These phases are used to monitor the development progress and the results of these activities are used to improve the algorithm.

Verification is initially performed in the simulation stage, in which first the basic functionality is tested to assess whether the individual subsystems and functionality perform as intended. This is done through the use of simplified unit tests in the Simulink environment, in which the functionalities of separate system components can be tested individually to identify programming errors or low level design flaws. Subsequently, full system verification is carried out, in which the algorithm is simulated in representative scenarios with realistic system dynamics. This is done to verify whether the system is capable of performing the tasks it is assigned to do, as defined in the research objectives.

In the flight testing, another series of unit verifications is carried out to assess the functioning of all of the subsystems and interfaces. First of all, the interfaces between the quadrotor and external applications such as the optical tracking system and possible external computation are required before proceeding to testing. Additionally, unit testing is performed on the flight control subsystems to minimize the risk of bugs or computational errors such as overflows making their way into actual flight testing. The final verification performed in this stage is to perform low risk simple trajectory tasks, such as figure eights or spiral trajectories. The goal is to assess whether the system as a whole is indeed capable of performing trajectory generation and tracking, without going towards the edge of the performance envelope, minimizing the risk of damage to the quadrotor.

In the validation runs, it is assessed whether the system meets the performance objectives set in the design stage. At this stage, these performance requirements have not been sufficiently defined, but during WP1 work, these will be expanded with more specific performance targets. Arguably, the first series of validation is the set of flight simulations, which are intended to be sufficiently realistic to attain a clear indication of the achievable performance of the quadrotor. This performance assessment is used in an iterative algorithm improvement cycle, up until there is sufficient confidence that the algorithm is capable of performing in real world flight tests.

The second validation phase is set to take place in the flight tests in the cyberzoo. In this stage, it is validated whether the performance requirements can be met in real-life as well, when subjected to effects as computational delay, measurement imperfections and actual aerodynamic disturbances. Similarly, this test series is used to iteratively improve the performance of the algorithm.

5.6. Experiment Set-up

This section elaborates on the experimental set-up that is used to perform the verification and validation test cycles. The first series is conducted in simulation in Simulink, whereas the second series consists of flight tests using a Parrot Bebop quadrotor.

5.6.1. Quadrotor Platform

Currently, it is expected that the experiments are to take place with the Parrot Bebop 2 quadrotor, a commercial, lightweight, low-cost quadrotor platform. This quadrotor is equipped with a front monovision camera, as well as a low-resolution bottom camera for vision applications, while an MPU6050 IMU provides acceleration and attitude rate measurements at 512 Hz. On-board processing is provided by a Parrot P7 dual-core Cortex CPU module that runs the flight software. [45][41] The on-board computational capabilities of the Parrot Bebop are very limited, which may put a strong constraint on the computational cost of the trajectory generation solution. In previous research, it was possible to run both an INDI-based fault-tolerant control algorithm and an Extended Kalman Filter (EKF) on board at 500 Hz. [44] The outer loop guidance algorithms generally run at lower frequency, estimated between 20-50 Hz. [33] If the processor is incapable of running the envisioned real-time trajectory guidance at this frequency, it is also possible to use off-board computing through the connection of an external computer to provide the computations. While this is undesirable in finding a feasible real-time solution, it is expected that future platforms will have significantly larger computational performance. An example of this is the platform used by Sun for vision based state estimation. [46]

In earlier experiments, the camera module of the Bebop2 has been removed to decrease weight, in combination with the deployment of a lighter battery module. This increases the maneuverability of the quadrotor, reducing its mass to 347 grams.

Although the use of the Bebop quadrotor is expected, Parrot has discontinued the support for this model. Therefore it is possible that the platform is soon replaced by a new in-house designed platform. In this case, the experimental set-up is expected to change drastically.

5.6.2. Quadrotor Simulator

As mentioned in section 5.2.2, the Control & Simulation department has developed a realistic simulator for the Parrot Bebop quadrotor. This simulator models realistic quadrotor dynamics and incorporates realistic simulation of sensor behavior. This makes it very suitable to assess the performance of the designed control algorithms. This simulator has been used extensively in earlier research by Sun, de Visser, Baert, Strack van Schijndel, Lu and van Kampen. [42][44][45]. These simulations were found to be representative for the actual quadrotor behavior, so it can be assumed that this simulator is very suitable for this research as well. Therefore, this simulation will be used to get a realistic indication of the real-world performance of the guidance algorithm in executing aggressive maneuvering trajectories, in which both the trajectory generation and control system performance can be assessed. The only thing that may be of difficulty to take into account in the simulation is the computational cost of the proposed solutions. It is key to make a sufficiently accurate estimation of this computational cost, in an attempt to include this in the simulation phase, to prevent the implementation of an infeasible solution.

As mentioned in the previous section, the Parrot Bebop2 may be replaced by an in-house developed MAV as the platform for flight testing. In this case, adaptations to the simulations will have to be made to accommodate this change.

5.6.3. Test Facility

Flight testing is to take place in the Cyberzoo facility at the Faculty of Aerospace Engineering. This is a 6x6x6m environment, encapsulated by nets that is safe to perform extensive testing with MAVs.

The main advantage of this location is the presence of Optitrack motion tracking system, that makes use of 12 motion capture cameras to provide accurate state estimation to the quadrotor at 120 Hz. By fitting the quadrotor with 6 white dots, the system is capable of providing highly accurate position, velocity and attitude estimates to the quadrotor, mitigating the difficulties with state estimation using on-board sensors. At larger attitude angles, however, Optitrack has difficulties accurately tracking the quadrotor motion. In earlier experiments, this is mitigated by means of an Extended Kalman Filter that fuses vision data and on-board sensor measurements. [44]

The Cyberzoo has a wide range of objects that can serve as obstacles or apertures to navigate through, such as large columns or window panes. These can be used for obstacle avoidance testing, once these obstacles have been accurately mapped as constraints in the trajectory generation algorithm. Vision based methods to accurately recognize these obstacles is beyond scope of this research and has not properly been researched yet for quadrotors experiencing rotor failure.

The upset recovery tests can be conducted by throwing the quadrotor into the air at arbitrary velocity, upset angles and angular rates. While the cyberzoo is an excellent location for these tests, there is a vertical space limitation of 6m, which gives limited time for recovery. Ideally, this would be tested outside, but as of now, it seems infeasible to perform outdoor testing, considering the limited capability of on-board sensing equipment. The vision-based state estimation research by Sun et al [46] do propose a methodology for state estimation in outdoor environments, however, these methods are currently only deemed feasible for simpler trajectory tracking towards a safe landing zone.

6 Project Planning

Given the approach specified in chapter 5 a planning is created for the thesis phase. This planning is incorporated into a Gantt chart, with a specification of important deliverables and milestones. The Gantt chart can be seen in fig. 6.1, this chapter provides an elaboration on this visual planning. This planning is based on the work breakdown in fig. 5.1, which is discussed in detail in chapter 5.

Table 6.1 contains the start and end date of each phase and provides an overview of the envisioned deliverables per phase. This will be discussed in more details in the following sections.

Phase	Name	Start Date	End Date	Deliverables
WP 1	Design	April 19	May 28	• Guidance Loop Design • Control Loop Design
WP 2	Development	May 31	July 2	• Guidance Simulation • Simulink Guidance Algorithm • Simulation Dataset
WP 3	Implementation	July 12	September 3	• Guidance Software (PX4) • Experiment Dataset • Midterm Review
WP 4	Finalization	September 20	November 26	• Final Report • Thesis Defence Presentation

Table 6.1: Overview of start and end dates per thesis phase, including an overview of the deliverables per phase

6.1. Work Package 1

The first work package starts April 19th, with the modeling phase, which lasts three weeks. The goal of this modeling phase is to derive a dynamic model representation of the damaged quadrotor. Two weeks into this modeling process, the design of the trajectory generation algorithm starts alongside the modeling effort, due to the interdependencies between these two elements. This process is estimated to take three weeks in total. Finally, in the final two weeks of WP1, the design of the interfaces between the guidance algorithm and the other subsystems such as system identification and control systems is designed. The deliverables of this phase are design concepts for the guidance and control loop architecture.

6.2. Work Package 2

Work package 2 is slated to start 31st and will commence with the build of the simulation, which will take approximately two weeks. The second week of WP2, verification will begin with unit verifications on the simulator, which is expected to be fully operational on June 14th. The total simulation and verification phase will last 4 weeks in total, which is largely done in conjunction with continuous model improvements. At the end of WP2, the expected deliverables are a functional quadrotor guidance simulator and an analysed dataset of the simulation test results, to be used in the final phases of the project.

6.3. Work Package 3

The third workpackage is focused on the physical implementation of the guidance algorithm and will commence on July 12th after a week long summer holiday. It commences with the development of the PX4 guidance algorithm, which is slated to take two weeks. The experiment preparation is initiated on July 19th and will last two weeks as well. The continuous improvement process of WP3.4 is initiated during the experiment preparation on July 195h, during which the bugs are removed from the system and the developed algorithm is refined. On August 2, the first series of flight tests is to be conducted, lasting roughly two weeks, performed in conjunction with the continuous improvements on the algorithm. In the subsequent two weeks, there is time to refine the algorithms and software to improve the performance, after which a second series of tests is conducted to assess the improvements. If possible during the holidays, a **Midterm Review** is envisioned on **July 30th, 2021**.

6.4. Work Package 4

The final phase starts on September 20, with analysis of the experimental data for the purpose of generating results and conclusion, a process that is expected to take four weeks. On September 21, reporting is started, for which 6 weeks are available. The envisioned date for the **Greenlight Review** is currently set at **Octobre 6**, after which a **Thesis Hand-in** of **November 12** is planned. Finally, the **Thesis Defence** is currently planned for **November 26, 2021**.

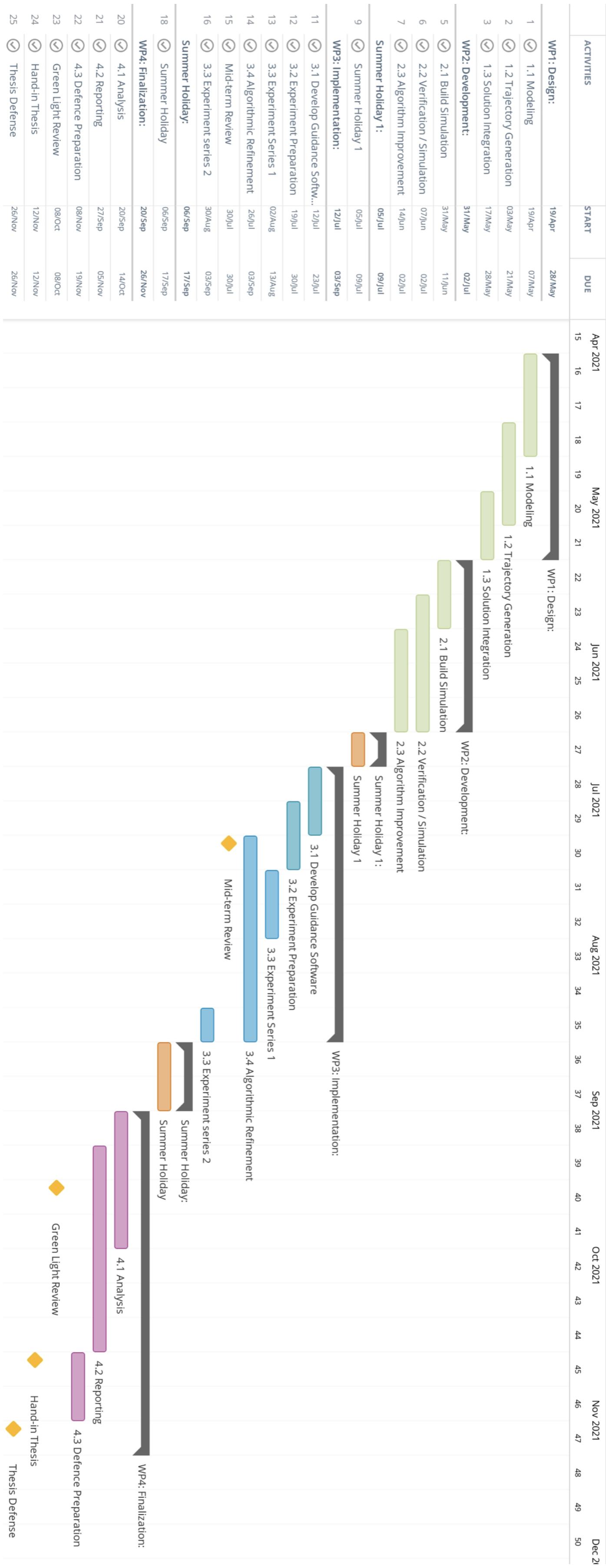

Figure 6.1: A Gantt chart providing an overview of the planning of the Thesis phase.

7 Conclusion

This report has provided an overview of the literature study and preliminary phase of a thesis research into aggressive maneuvers for quadrotors experiencing single rotor failure. This report served to provide an overview of the state-of-the-art in fault-tolerant control and trajectory generation for quadrotor Micro-aerial vehicles, after which a project approach was asserted using this information.

Regarding the literature study, extensive research has been conducted into quadrotor modeling, fault-tolerant control and trajectory generation. During this research, a clear literature gap has been identified in the development of a real-time trajectory generation algorithm for quadrotors experiencing rotor loss. The objective of this research is to enable the damaged quadrotor to perform aggressive maneuvers, as well as recovery from loss-of-control situations during large attitude upsets.

During the literature study, the main challenge has been identified to be the development of a method that is capable of generating trajectories in real-time, while taking into account the significant constraints on the quadrotor performance and its computational capacity.

Given this objective, combined with the observed challenges and the knowledge of the current state-of-the-art, an approach for the upcoming project was defined, including verification and validation procedures and experimental set-ups. Finally, this information is condensed into a planning for the thesis project, subdivided into four different work packages.

Bibliography

[1] Pieter Abbeel, Adam Coates, and Andrew Y. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *International Journal of Robotics Research*, 29(13):1608–1639, 2010. ISSN 02783649. doi: 10.1177/0278364910371999.

[2] Matthias Baert. Quadrotor Upset Recovery after Rotor Failure. (June), 2019. URL `http://resolver.tudelft.nl/uuid:4e1011c7-2db9-4dfe-a910-f2843414b960`.

[3] Dario Brescianini and Raffaello D'Andrea. Tilt-Prioritized Quadrocopter Attitude Control. *IEEE Transactions on Control Systems Technology*, 28(2):376–387, 2020. ISSN 15580865. doi: 10.1109/TCST.2018.2873224.

[4] Efe Camci and Erdal Kayacan. Learning motion primitives for planning swift maneuvers of quadrotor. *Autonomous Robots*, 43(7):1733–1745, 2019. ISSN 15737527. doi: 10.1007/s10514-019-09831-w. URL `https://doi.org/10.1007/s10514-019-09831-w`.

[5] Ayman A. El-Badawy and Mohamed A. Bakr. Quadcopter Aggressive Maneuvers along Singular Configurations: An Energy-Quaternion Based Approach. *Journal of Control Science and Engineering*, 2016, 2016. ISSN 16875257. doi: 10.1155/2016/7324540.

[6] Karam Eliker, Guoqing Zhang, Said Grouni, and Weidong Zhang. An Optimization Problem for Quadcopter Reference Flight Trajectory Generation. *Journal of Advanced Transportation*, 2018, 2018. ISSN 20423195. doi: 10.1155/2018/6574183.

[7] Emil Fresk and George Nikolakopoulos. Experimental evaluation of a full quaternion based attitude quadrotor controller. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2015-Octob(644128):15–18, 2015. ISSN 19460759. doi: 10.1109/ETFA.2015.7301555.

[8] Hiroyuki Fujioka and Hiroyuki Kano. Control theoretic B-spline smoothing with constraints on derivatives. *Proceedings of the IEEE Conference on Decision and Control*, (1):2115–2120, 2013. ISSN 01912216. doi: 10.1109/CDC.2013.6760194.

[9] Jeremy H. Gillula, Haomiao Huang, Michael P. Vitus, and Claire J. Tomlin. Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1649–1654, 2010. ISSN 10504729. doi: 10.1109/ROBOT.2010.5509627.

[10] Markus Hehn and Raffaello Dandrea. Real-Time Trajectory Generation for Quadrocopters. *IEEE Transactions on Robotics*, 31(4):877–892, 2015. ISSN 15523098. doi: 10.1109/TRO.2015.2432611.

[11] Haomiao Huang, Gabriel M. Hoffmann, Steven L. Waslander, and Claire J. Tomlin. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3277–3282, 2009. ISSN 10504729. doi: 10.1109/ROBOT.2009.5152561.

[12] Avraiem Iskander, Omar Elkassed, and Ayman El-Badawy. Minimum Snap Trajectory Tracking for a Quadrotor UAV using Nonlinear Model Predictive Control. *2nd Novel Intelligent and Leading Emerging Sciences Conference, NILES 2020*, pages 344–349, 2020. doi: 10.1109/NILES50944.2020.9257897.

[13] Derya Kaya and Ali Türker Kutay. Aerodynamic modeling and parameter estimation of a quadrotor helicopter. *AIAA AVIATION 2014 -AIAA Atmospheric Flight Mechanics Conference*, (June):1–28, 2014. doi: 10.2514/6.2014-2558.

[14] Torsten Kröger and Frederik Wahl. On-Line Trajectory Generation in Robotic Systems: Basic Concepts for Instantaneous Reactions to Unforeseen (Sensor) Events. *Springer Tracts in Advanced Robotics*, 58(1): 1–230, 2010. ISSN 16107438. doi: 10.1007/978-3-642-05175-3_1.

[15] Shupeng Lai, Kangli Wang, and Ben M. Chen. Dynamically feasible trajectory generation method for quadrotor unmanned vehicles with state constraints. *Chinese Control Conference, CCC*, pages 6252–6257, 2017. ISSN 21612927. doi: 10.23919/ChiCC.2017.8028351.

[16] Menglu Lan, Shupeng Lai, and Ben M Chen. Towards the Realtime Sampling-based Kinodynamic Planning for Quadcopters. pages 772–777, 2017.

[17] Alexander Lanzon, Alessandro Freddi, and Sauro Longhi. Flight control of a quadrotor vehicle subsequent to a rotor failure. *Journal of Guidance, Control, and Dynamics*, 37(2):580–591, 2014. ISSN 15333884. doi: 10.2514/1.59869.

[18] Yanhui Li and Chao Liu. Efficient and Safe Motion Planning for Quadrotors Based on Unconstrained Quadratic Programming. *Robotica*, 39(2):317–333, 2021. ISSN 14698668. doi: 10.1017/S0263574720000387.

[19] Peng Lu and Erik Jan Van Kampen. Active fault-tolerant control for quadrotors subjected to a complete rotor failure. *IEEE International Conference on Intelligent Robots and Systems*, 2015-Decem:4698–4703, 2015. ISSN 21530866. doi: 10.1109/IROS.2015.7354046.

[20] Sergei Lupashin and Raffaello D'Andrea. *Adaptive open-loop aerobatic maneuvers for quadrocopters*, volume 44. IFAC, 2011. ISBN 9783902661937. doi: 10.3182/20110828-6-IT-1002.03205. URL `http://dx.doi.org/10.3182/20110828-6-IT-1002.03205`.

[21] Sergei Lupashin and Raffaello D'Andrea. Adaptive fast open-loop maneuvers for quadrocopters. *Autonomous Robots*, 33(1-2):89–102, 2012. ISSN 09295593. doi: 10.1007/s10514-012-9289-9.

[22] Sergei Lupashin, Angela Schöllig, Michael Sherback, and Raffaello D'Andrea. A simple learning strategy for high-speed quadrocopter multi-flips. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1642–1648, 2010. ISSN 10504729. doi: 10.1109/ROBOT.2010.5509452.

[23] Robert Mahony, Vijay Kumar, and Peter Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics and Automation Magazine*, 19(3):20–32, 2012. ISSN 10709932. doi: 10.1109/MRA.2012.2206474.

[24] Philippe Martin and Erwan Salaün. The true role of accelerometer feedback in quadrotor control. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1623–1629, 2010. ISSN 10504729. doi: 10.1109/ROBOT.2010.5509980.

[25] Daniel Mellinger and Vijay Kumar. Minimum Snap Trajectory Generation and Control for Quadrotors. pages 2520–2525, 2011.

[26] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *International Journal of Robotics Research*, 31(5):664–674, 2012. ISSN 02783649. doi: 10.1177/0278364911434236.

[27] Mark W. Mueller and Raffaello D'Andrea. A model predictive controller for quadrocopter state interception. *2013 European Control Conference, ECC 2013*, pages 1383–1389, 2013. doi: 10.23919/ecc.2013.6669415.

[28] Mark W. Mueller and Raffaello D'Andrea. Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2014. doi: 10.1109/ICRA.2014.6906588.

[29] Mark W. Mueller and Raffaello D'Andrea. Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles. *International Journal of Robotics Research*, 35(8):873–889, 2016. ISSN 17413176. doi: 10.1177/0278364915596233.

[30] Mark W Mueller, Markus Hehn, and Raffaello D Andrea. A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification. pages 3480–3486, 2013.

[31] Abischek Narasimhan, Coen de Visser, and Cristophe De Wagter. Fault Tolerant Control of Multirotor UAV for Piloted Outdoor Flights. pages 1–30.

[32] Ngoc Thinh Nguyen, Ionela Prodan, and Laurent Lefevre. Effective angular constrained trajectory generation for thrust-propelled vehicles. In *2018 European Control Conference, ECC 2018*, number 1, pages 1833–1838. European Control Association (EUCA), 2018. ISBN 9783952426982. doi: 10.23919/ECC.2018.8550134.

[33] Shu peng Lai, Meng lu Lan, Ya xuan Li, and Ben M Chen. Safe navigation of quadrotors with jerk limited trajectory. *Frontiers of Information Technology and Electronic Engineering*, 20(1):107–119, 2019. ISSN 20959230. doi: 10.1631/FITEE.1800719.

[34] Xin Qi, Didier Theilliol, Yuqing He, and Jianda Han. Recoverable Set Computation for Post-fault / failure Quadrotors based on Sum Of Squares (SOS). pages 73–78, 2016.

[35] Gerasimos G. Rigatos. *Differential flatness theory and flatness-based control*, volume 25. 2015. ISBN 9783319164199. doi: 10.1007/978-3-319-16420-5_2.

[36] Ferdinando Santacroce, Aske Olsson, Rasmus Voss, and Jakub T A T T Narębski.

[37] David K. Smith and Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control Volume 1, Volume 2.*, volume 47. 1996. ISBN 1886529264. doi: 10.2307/3010291.

[38] Igor Spasojevic, Varun Murali, and Sertac Karaman. Perception-aware time optimal path parameterization for quadrotors, 2020. ISSN 23318422.

[39] Bram Strack Van Schijndel, Sihao Sun, and Coen de Visser. FLoED : Fast Loss of Effectiveness Detection on a Quadrotor using Onboard Sensors and a Kalman Estimation Approach.

[40] S. Sun and C. C. de Visser. Quadrotor safe flight envelope prediction in the high-speed regime: A monte-carlo approach. *AIAA Scitech 2019 Forum*, (January), 2019. doi: 10.2514/6.2019-0948.

[41] Sihao Sun and Coen de Visser. Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime. *IEEE Robotics and Automation Letters*, 4(4):3868–3875, 2019. ISSN 2377-3766. doi: 10.1109/lra.2019.2928758.

[42] Sihao Sun, Leon Sijbers, Xuerui Wang, and Coen De Visser. High-Speed Flight of Quadrotor Despite Loss of Single Rotor. *IEEE Robotics and Automation Letters*, 3(4):3201–3207, 2018. ISSN 23773766. doi: 10.1109/LRA.2018.2851028.

[43] Sihao Sun, Coen C. De Visser, and Qiping Chu. Quadrotor gray-box model identification from high-speed flight data. *Journal of Aircraft*, 56(2):645–661, 2019. ISSN 15333868. doi: 10.2514/1.C035135.

[44] Sihao Sun, Matthias Baert, Bram Strack Van Schijndel, and Coen De Visser. Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2020. ISBN 9781728173955. doi: 10.1109/ICRA40945.2020.9197239.

[45] Sihao Sun, Xuerui Wang, Qiping Chu, and Coen de Visser. Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors. *IEEE Transactions on Robotics*, 37(1):116–130, 2020. ISSN 19410468. doi: 10.1109/TRO.2020.3010626.

[46] Sihao Sun, Giovanni Cioffi, Coen De Visser, and Davide Scaramuzza. Autonomous Quadrotor Flight despite Rotor Failure with Onboard Vision Sensors: Frames vs. Events. *IEEE Robotics and Automation Letters*, 6(2):580–587, 2021. ISSN 23773766. doi: 10.1109/LRA.2020.3048875.

[47] Gao Tang, Weidong Sun, and Kris Hauser. Learning Trajectories for Real- Time Optimal Control of Quadrotors. *IEEE International Conference on Intelligent Robots and Systems*, pages 3620–3625, 2018. ISSN 21530866. doi: 10.1109/IROS.2018.8593536.

[48] Jie Tang, Arjun Singh, Nimbus Goehausen, and Pieter Abbeel. Parameterized maneuver learning for autonomous helicopter flight. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1142–1148, 2010. ISSN 10504729. doi: 10.1109/ROBOT.2010.5509832.

[49] Teodor Tomić, Moritz Maier, and Sami Haddadin. Learning quadrotor maneuvers from optimal control and generalizing in real-time. *Proceedings - IEEE International Conference on Robotics and Automation*, (Section III):1747–1754, 2014. ISSN 10504729. doi: 10.1109/ICRA.2014.6907087.

[50] Hanzhang Wang, Yisha Liu, and He Deng. A novel interactive system for quadrotor path planning in cluttered outdoor environments. *2019 IEEE International Conference on Real-Time Computing and Robotics, RCAR 2019*, 2019-August:852–857, 2019. doi: 10.1109/RCAR47638.2019.9043974.