Fault Tolerant Control in Over-Actuated Hybrid Tilt-Rotor Unmanned Aerial Vehicles

Thesis Report

Nico Voß

Fault Tolerant Control in Over-Actuated Hybrid Tilt-Rotor Unmanned Aerial Vehicles

Thesis Report

Thesis report

by

Nico Voß

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on June 17th, 2024 at 13:00

Thesis committee:	
Chair:	Dr. Coen de Visser
Supervisors:	Dr. Ewoud J. J. Smeur
	Alessandro Mancinelli, PhD Candidate
External examiner:	Dr. Alessandro Bombelli
Place:	Faculty of Aerospace Engineering, Delft
Student number:	4550749

Faculty of Aerospace Engineering · Delft University of Technology

Contents

1
Introduction

With the rise of drones in both private and commercial use the search for more capable systems continues. One of the most prevalent trends is to extend the flight envelope of vehicles to combine high efficiency forward flight and hover capability. Combining the previous experience from quadcopters and other drones, tailsitters are one of the major developments in the chase for ever more capable systems. Another solution, embodied by the subject of this paper, is to make use of tilt mechanisms. A tilt configuration adds additional control inputs to the system, which can be used to optimise the control allocation in each flight condition. Due to their overactuated nature they open up new possibilities for fault tolerant control. With the drone under consideration these capabilities can already be used to perform landings on slanted or actively tilting surfaces such as ships. The vehicle and payload can be oriented largely independent of desired linear accelerations. Provided sufficient thrust to weight ratio, drones with such configurations could remain in fully controlled flight with several concurrent failures. This Thesis investigates the implementation of a controller on the vehicle shown in Figure 1.1 utilising a modified optimisation based nonlinear controller.

Figure 1.1: The drone at Valkenburg Unmanned Valley.

Research Formulation

As established in the included literature study the goals for the project were defined from an early stage. As the work was performed on an existing vehicle and the goal of the project was to introduce fault tolerant control, the objective was already given. What remained was to phrase a set of research questions to be answered over the course of the project, which will be presented below.

Research Objective: Test in simulation and implement on the prototype drone a means to achieve fault tolerant control in the presence of actuator failure.

Research Questions:

- Where are the issues with the current implementation?
- What changes have to be made to the current layout to enable FTC?
- How well does the proposed architecture perform on the vehicle?

Part I

Scientific Article

Fault Tolerant Control in Over-Actuated Hybrid Tilt-Rotor Unmanned Aerial Vehicles

Nico Voß
TU Delft, Faculty of Aerospace Engineering

Ewoud J. J. Smeur and Alessandro Mancinelli
Supervisors, TU Delft, Faculty of Aerospace Engineering

***Abstract*-Quad-planes combine hovering and vertical takeoff and landing capability with fast and efficient forward flight. Regular Quad-planes with dedicated pusher motor can be subject to gust disturbances, and are not well-equipped to deal with actuator faults. Dual-axis Tilt-Rotor quad-planes are more maneuverable due to their overactuation. This also increases their gust resilience and allows them to hover statically after actuator failures. The vehicle in this paper uses an Incremental Nonlinear Dynamic Inversion (INDI) controller, combined with a nonlinear Sequential Quadratic Programming (SQP) Control Allocation (CA) algorithm, which can also find hover solutions in the case of actuator failures. We investigate both a combined allocation of linear and angular accelerations, as well as a cascaded allocation scheme. Due to large required changes in roll and pitch angles, the cascaded approach is selected in this research. Introduction of a tertiary control effort term, separation of attitude and actuator command optimization and a simulated Fault Detection and Identification (FDI) mechanism led to repeated successful recovery from a motor failure in hover. Position tracking was demonstrated under failure in the reconfigured flight condition.**

***Index Terms*- Tilt-rotor, dual-axis tilt, quad-plane, FTC, over-actuated, control allocation**

I. Introduction

Historically, the development of Vertical Takeoff and Landing (VTOL) aircraft is driven by the benefit of combined efficient high-speed long-range flight and the capability of landing in unprepared and/or size restricted areas. Before the era of unmanned systems, several projects investigated full-scale concepts to achieve this combination, making use of separate or multi-purpose propulsion systems. Despite higher mechanical and control complexity, making use of a single propulsion system for both modes of flight is desirable for efficiency. Tilt-wing and tilt-rotor systems have established themselves as the current industry standard allowing a single propulsion system throughout hover, transition and forward flight regimes by adjusting the thrust vector accordingly.

For smaller and more agile systems, the research has expanded towards utilising the tilt-mechanisms in additional ways, i.e. gust rejection or physical interaction tasks [1, 2]. At the same time powerful System-on-Chip (SoC) solutions offer affordable computing power to implement complex real-time CA algorithms within limited power and weight budgets. The potential advantages of rotor tilt increase with the number of additional tilt axes in the CA problem. By adding as few as two additional servos and spherical joints to a quad-layout to control the common motor tilt, Zheng et al. [3] demonstrated attitude independent thrust. This configuration tilts the common thrust vector via a gimbal linkage to control motor rotation parallel to body pitch and roll axes.

Several other studies have previously investigated the increased maneuverability resulting from tilt configurations, including Junaid et al. [4]. Their study especially noted improvements in cornering flight and obstacle avoidance. A step further in complexity is independent tilt of individual rotors: Mousaei et al. [5] previously developed a quad-plane with independent single axis tilt along the vehicle pitch axis on all 4 rotors and investigated motor failure in hover and forward flight. With their quad-plane shown in Figure 3 they successfully simulated the recovery from a motor failure in hover and forward flight modes.

Independent dual-axis tilt of individual motors grants 4 independent thrust vectors at the cost of greater allocation complexity from the additional actuators in the optimisation problem. A realisation of such a concept is the quad-plane developed by Mancinelli et al. [6] shown in Figure 1. Featuring 4 independent dual-axis tilt-rotors, the project vehicle showed increased capability in disturbance rejection and maneuverability. Compared to the previously introduced quad-plane, the additional tilting of motors parallel to the vehicles roll axis allow for lateral thrust vectoring and reorientation in roll and pitch under motor failure.

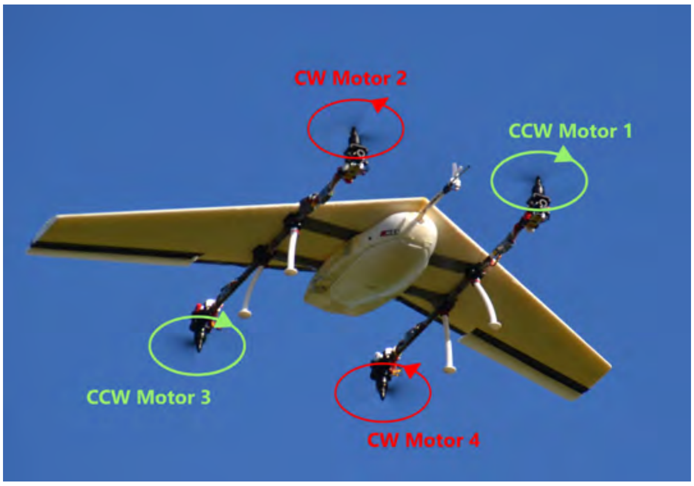


Fig. 1 Dual-axis tilt Tilt Rotor Unmanned Aerial Vehicle (TRUAV) quad-plane developed by Mancinelli et al. [6]

In order to mitigate actuator failure the following two major approaches to Fault Tolerant Control (FTC) can be discerned: In active FTC, the system is actively monitored and faults need to be detected and communicated to the controller. With the provided information on the failure, which relies on an FDI mechanism, the controller takes action to mitigate the failure. In passive FTC, the controller is instead constructed such that it is robust to failures in the system (within established limits). This usually leads to a more conservative design and reduced overall performance, but does not require any active monitoring component[7–9]. As both introduced tilt-rotor quad-planes rely on an on-line prediction of model dynamics for their control allocation, a means to inform the controller of the changed configuration is required. For this reason, active fault tolerance is applied to modify the actuator characteristics accordingly.

Fig. 2 Wang et al. quadplane [10]

In a previous paper Wang and Sung focused on the effect a lifting body has on the more classical types of recovery for quadrotors and proposed a novel Incremental Adaptive Sliding Mode Control (I-ASMC) approach to mitigate uncertainties in the modeled aerodynamic forces and interactions of rotor and wing [10]. The proposed solution simulated flight with only three rotors and had the spinning quad-plane follow a rectangular reference trajectory. The vehicle is shown in Figure 2.

This relaxed hover state by allowing yaw rotation is a proven approach to overcome the inherent under-actuation of quad-copters[11]. Sun et al. have previously demonstrated flight with two rotors for a commercial drone [12] and Zhang et al. designed a vehicle that can track position with only a single actuator [13].

With dual-tilt vehicles the independent thrust vectoring and optimization based CA allows stabilizing a motor failure without yaw spin. The dual-tilt enabled static hover also benefits from not having to model and adapt to the complex aerodynamic interactions of a spinning wing.

Within this project, the on-board optimization is used to demonstrate fault tolerant control on the dual-axis tilt quadplane by establishing a new static hover condition. The introduction of a cascaded nonlinear control allocation separates the attitude and actuator command optimization. A tertiary control effort term is introduced to minimize the control effort and smooth the optimized commands in the transition to the new hover attitude after an actuator failure is introduced. The paper derives the updated control law and presents the results of implementing the same control law on the flying vehicle.

Fig. 3 Carnegie Mellon AIRLab TRUAV [5]

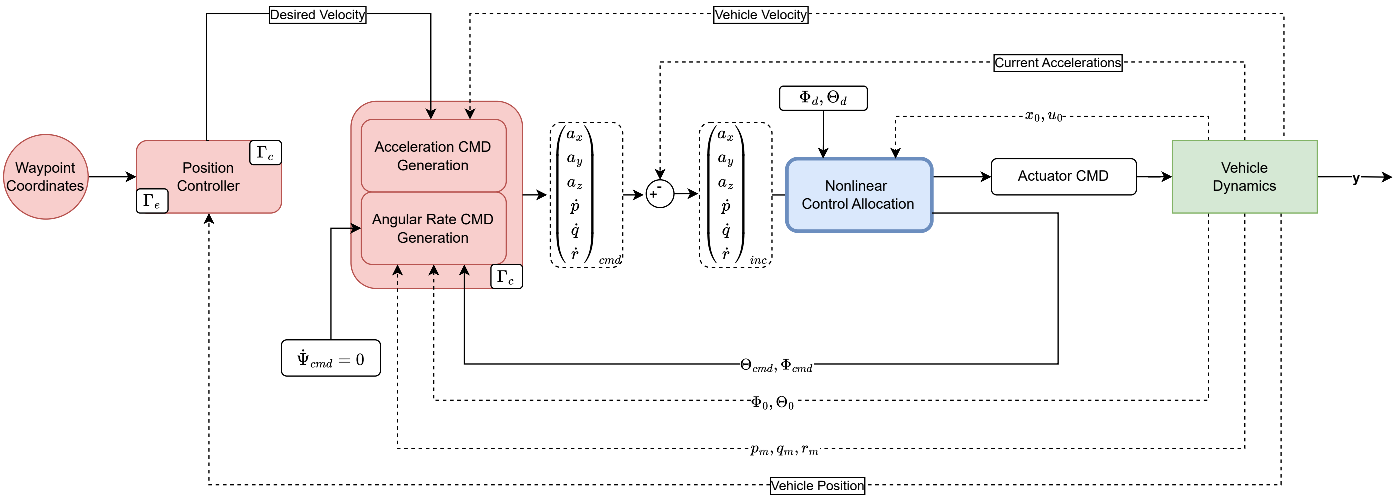


Fig. 4 The original controller layout including the error controllers.

II. Method

A. Reference Frames Definition - Equations of Motion

For the setup of the controller, a number of different right-handed reference frames are of importance to understand the orientations, commands and actuator controls.

- Γ_e Earth reference frame (NED):
  - Origin fixed to earth surface reference point
  - x_e pointing towards North
  - y_e pointing towards East
  - z_e positive to Earth center
- Γ_b Body reference frame:
  - Origin is fixed to vehicle Center of Gravity:
  - x_b pointing forward along vehicle roll axis
  - y_b pointing out of right wing (pitch axis)
  - z_b pointing down (yaw axis)
- Γ_c Control reference frame:
  - Origin is fixed to vehicle Center of Gravity:

- x_c pointing towards nose projected onto earth surface
- y_c pointing right seen from above, perpendicular to z_c and x_c
- z_c pointing down to Earth center

- Γ_p^i Propeller reference frame: Origin is fixed at i−th gimbal point, the axis directions are aligned with the body frame under zero gimbal controls, which can be seen on the front right motor 2 in Figure 6.

The visualisation of these reference frames of the considered vehicle can be seen in Figure 5.

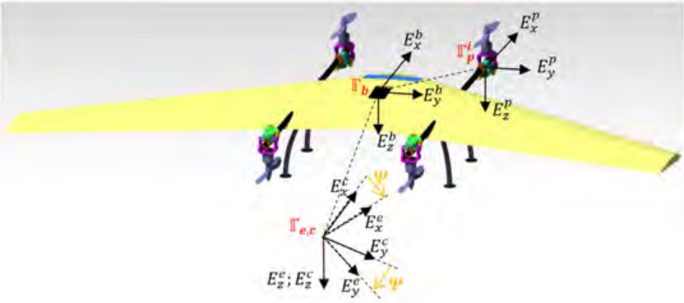


Fig. 5 Definition of reference frames [14]

Due to the independent actuation of the rotors, a set of four primary rotation matrices is required to establish the equations of motion and control laws. The first rotation matrix describes the transformation from earth to control reference frames Γ_e to Γ_c:

$$R_{ec} = \begin{bmatrix} \cos(\Psi) & \sin(\Psi) & 0 \\ -\sin(\Psi) & \cos(\Psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

This transformation is used primarily within the position error controller to generate the desired velocities and accelerations in the control frame. Additionally, the transform between the body reference Γ_b and control reference frame Γ_c is given by:

$$R_{cb} = \begin{bmatrix} \cos(\Theta) & \sin(\Phi)\sin(\Theta) & \cos(\Phi)\sin(\Theta) \\ 0 & \cos(\Phi) & -\sin(\Phi) \\ -\sin(\Theta) & \sin(\Phi)\cos(\Theta) & \cos(\Phi)\cos(\Theta) \end{bmatrix} \tag{2}$$

Finally, due to the independent orientation of the motors with respect to the body frame, computation of forces and moments requires an additional coordinate transformation for each motor. The transformation of the i−th motor to the body frame $\Gamma_p^i \rightarrow \Gamma_b$ is described by:

$$R_{bp}^i = \begin{bmatrix} \cos(b^i) & 0 & \sin(b^i) \\ \sin(g^i)\sin(b^i) & \cos(g^i) & -\sin(g^i)\cos(b^i) \\ -\cos(g^i)\sin(b^i) & \sin(g^i) & \cos(g^i)\cos(b^i) \end{bmatrix} \tag{3}$$

The definition of angles b_i, g_i is shown in Figure 6 denoting the elevation and azimuth tilt of the considered motor respectively. The dynamics of the vehicle within the CA are expressed by the following equations:

$$\begin{cases} \ddot{P}_c = \dfrac{1}{m}(F_p + F_a) + g z_c \\ \dot{\omega} = I_b^{-1}(-\omega \times I_b \omega + M_p^T + M_p^D + M_a) \end{cases} \tag{4}$$

In this, $\ddot{P}_c$ are the linear accelerations within the control reference frame Γ_c and $\dot{\omega}$ is the array of body rate derivatives. The contributing terms are composed of: F_p denoting the sum of thrust generated by the individual motors, rotated to the control frame. F_a the aerodynamic forces generated, also in control frame.

The thrust force in the control frame is exemplary for the utilization of the previously presented rotation matrices and computed from the thrust coefficient K_p^T and motor speed Ω_i as follows:

$$F_p = \sum_{i=1}^{N} R_{cb} R_{bp}^i \begin{pmatrix} 0 \\ 0 \\ -K_p^T \Omega_i^2 \end{pmatrix} \tag{5}$$

The moment components are simplified from the full model, as discussed in [14]. In order to decrease the problem complexity for the on-line computation, the following motor related moment terms are assumed negligible: The precession (both from gimbal and body rate), torque due to rotational speed changes and torque from moving the motor mass about the gimbal. The remaining components are M_p^T the torque generated from the motor thrust, M_p^D the torque generated by rotor drag and M_a the combination of aerodynamic moments action on the vehicle.

For the current flight test scenario, where only hover tests were conducted, the aerodynamic forces and moments were negligible since no significant airspeed was achieved at any time. The interaction between the lifting body and the inflow generated by the motors was not modeled. A previous study investigated the roll moment interaction between the wing and the motors [15], but this was not further considered for this project.

B. The Controller Layout

The vehicle's controller contains two main components. A single-loop Control Allocation algorithm forms the primary component, generating commands for 13 physical actuators (8 tilting servos, 4 motor RPM commands, and 1 aileron servo pair) and two virtual actuators that control the vehicle's roll and pitch angles. The Control Allocation algorithm determines the optimal actuator commands to achieve the desired accelerations by minimizing a cost function that includes a model of the vehicle dynamics, as derived in the previous section. The Control Allocation algorithm receives angular and linear acceleration inputs generated by a linear error controller.

The error controller provides acceleration references for both linear and angular accelerations. For linear acceleration references, the error controller receives a setpoint for the desired vehicle position. It then uses feedback from the vehicle's current position and speed to generate the necessary linear acceleration references. A similar method is applied for angular acceleration generation, where a PD (Proportional-Derivative) error controller, with feedback on body rates and Euler angles, is used to generate the angular acceleration references.

This design ensures precise control of the vehicle and helps correct for unmodeled dynamics or external disturbances. A schematic representation of the architecture can be seen in Figure 4.

C. The Cost Function

The cost function is the expression that the CA algorithm tries to minimize for the generation of the control input commands. By adjusting the contributions to the cost function, the algorithm can prioritize certain control objectives. This approach integrates with the previously presented dynamics and control scheme by aiming to match the acceleration increments generated by the upstream error controllers using the equations of motion in Equation 4.

The following function shows a breakdown of the cost function structure and individual components:

$$C(u) = ||W_v(f(x_0,u) - v_n)||^2 + \gamma_u||W_u(u - u_d)||^2 + \gamma_{du}||W_{du}(u - u_0)||^2 \tag{6}$$

$$u_s = \arg\min C(u)$$

$$\text{subject to}$$
$$u_{min} < u < u_{max},$$

where the goal is to minimise the cost $C(u)$ by finding a u_s to minimize difference between desired and predicted acceleration increment $(f(x_0,u) - v_n)$. The desired acceleration of the vehicle in angular and linear terms are denoted v_n, with $f(x_0,u)$ as function computing the achieved accelerations from u and the current state x_0. Here v_n is the increment already adjusted by the acceleration to be expected from the current actuator setting. It also aims to minimize the difference between commanded and desired actuator settings $(u - u_d)$ and the step size between commanded and current actuator setting $(u - u_0)$.

$$u_d = (150, 150, 150, 150, 0, 0, 0, 0, 0, 0, 0, 0, 0, \Theta_d = 0, \Phi_d = 0), \tag{7}$$

where the last two entries only serve as virtual control for the desired optimal body attitude. The control vector u with the computed commanded controls and attitudes is structured as:

$$u = (\Omega_1, \Omega_2, \Omega_3, \Omega_4, b_1, b_2, b_3, b_4, g_1, g_2, g_3, g_4, \delta_a, \Theta_{cmd}, \Phi_{cmd}), \tag{8}$$

where Ω_i denote individual motor rotational speed, b_i the motor gimbal elevation, g_i the motor gimbal azimuth and δ_a the aileron deflection. The definition of the tilt angles can be seen in Figure 6.

The terms directly incurring cost from actuator use are scaled by $\gamma_u = 0.5e-6$ and $\gamma_{du} = 0.5e-6$, as they are secondary objectives to matching the desired accelerations. The weights in the matrices W_v, W_u and W_{du} as used during the test are given in Table 3. The goal is to find the optimal control solution u_s under the given constraints u_{min}, u_{max} as given in Table 1.

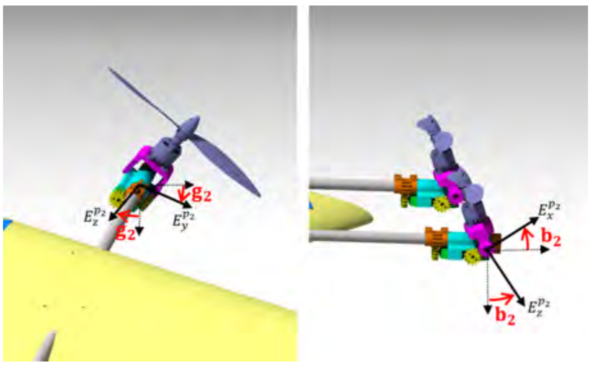


Fig. 6 Definition of gimbal angles as per Mancinelli et al.[14]

D. Addition of Tertiary Cost Term

Due to the secondary γ_u term being designed to keep the vehicle aligned with a desired reference attitude, the controller would fight to remain close to the originally intended hover attitude of Φ_d, Θ_d. This can be used in non-fault condition to control attitude independent from translation, provided a proper selection of W_u. Adding the tertiary γ_{du} term penalising large control actions within one optimizer iteration gives control about the actuator use beyond reference state deviation. This approach of incurring cost from the changes in actuator commands is commonly used in Model Predictive Control (MPC) to minimise the overall control effort [16, 17], and was introduced to improve the transition to the new recovered attitude by smoothing computed actuator and attitude changes. Penalizing changes in actuator commands has the negative side-effect of artificially slowing down the system dynamics and response. An appropriate selection for the weights has to be made, reaching a compromise between consistency and response.

E. Failure Information - Constraint Sets

The introduction highlighted the need to implement a means to inform the drone of failure and prevent the allocation algorithm from making use of non-effective actuators. Information was passed by an additional control channel, restricting the optimizer constraints based on the failure scenario. The non-fault actuator constraints can be seen in the Table 1, which are used during regular flight. Note that pitch and roll limits were set to prevent large angular ambiguities arising from the ZYX rotation order in the presented rotation matrices. Also, due to hardware constraints the azimuth servos are limited to around 85° in each direction such that excessive roll had to be limited.

Table 1 Default actuator and angle constraints.

Constraint	Minimum	Maximum
Motor speed [rad/s]	150	1000
Tilt Elevation [deg]	-130	20
Tilt Azimuth [deg]	-80	80
Theta cmd [deg]	-15	50
Phi cmd [deg]	-90	90

Once the failure is triggered, the controller internally switches to a different set of constraints that cause the selected motor to cut thrust and oriented into a stored orientation. This activated set of constraints can be seen in Table 2. The motor speed was never fully reduced to 0, as the framework did not allow to lower the motor speed Ω_i below 150 rad/s. Additional changes were made to the weights for the Θ and Φ references, as well as the motor weight.

By lowering the weights on the body angle reference and allowing the rotor gimbal to be controlled indirectly by motor cost in W_u, this additional tertiary term and the switch activated weights can stabilise the transition to a recovered attitude. The underlying concept is that aligning the motors for minimum motor speed also causes convergence to a certain gimbal configuration with the motors generating the required thrust as efficiently as possible. The additional cost from by the tertiary control effort term then serve to stabilize the solution around a found equilibrium once accelerations are matched.

Table 2 Adjustment in failure.

No Fault	Failed M3
$150 \leq \Omega_{m_3} \leq 1000$	150
$-130 \leq \delta_{el3} \leq 20$	$\delta_{el3} = 0$
$-80 \leq \delta_{az3} \leq 80$	$\delta_{az3} = 0$
$W_\Theta = 50$	$W_\Theta = 0.5$
$W_\Phi = 100$	$W_\Phi = 0.15$
$W_{motor} = 20$	$W_{motor} = 4$

F. The CA solver

The presented optimization problem is solved using the SQP approach, and makes use of the Matlab *fmincon* function. This implementation includes suggested improvements to K. Schittkowski's well-documented explanation of the algorithm [18] and is build on the work of Nocedal and Wright [19]. The choice of using the Matlab function also allows for the use of the Coder toolbox, accelerating the process of implementing the developed controllers on the drone.

Within the CA algorithm running on the drone, the current cost and gradient functions are evaluated and passed to the *fmincon* SQP method. This way the model based non-linearity of the system is included explicitly in the optimisation process. The relevant sub-functions were generated with the Matlab symbolic package, simplifying the resulting expressions for faster execution times.

G. Cascade Modification

From the original schematic layout of the controller in Figure 4, there are two primary concerns that also resulted in issues in preliminary simulation.

1) The controller computes a commanded attitude and all actuator commands are relative to this attitude. This does not take into account the current actual attitude or dynamics of the vehicle and actuators. Due to the actuator commands being relative to the commanded attitude, this leads to large deviation from the intended thrust and moments. This issue has previously been noted for regular flight but becomes much more pronounced in the presence of motor failure and the resulting rapid attitude changes.

2) The difference between the current and commanded attitude is used to generate the angular acceleration increment. In the event of motor failure, the vehicle shall reorient into a new attitude. The assumption that the actuator and attitude response of the vehicle have the same bandwidth, causes the discrepancy mentioned above. The difference between current and commanded attitude is additionally used to generate the angular acceleration reference. Especially in the transition this caused discrepancies which propagated back to the controller angular acceleration input. To counter this a cascaded approach was implemented in which the vehicle attitude is calculated independent of the actuator commands, solving the bandwidth issue. The desired angular acceleration increment is also disconnected from this optimizer, by setting the desired values to 0 instead. This removes the destabilizing feedback effect. Instead the first optimizer now acts like the original, but only the desired attitude is used to generate a further input for the second stage:

$$\begin{pmatrix} v_n & \Theta_d & \Phi_d \end{pmatrix}_{inc} \xmapsto[CA-1]{v_n \rightarrow u_s} \begin{pmatrix} \cancel{u_s} & \Theta_{cmd} & \Phi_{cmd} \end{pmatrix} \tag{9}$$

The desired attitude is only based on the current vehicle and actuator state and desired linear accelerations now and fed to an intermediate error controller. This controller is located between the stages and adds an angular acceleration increment to the overall commanded increment for the second stage optimizer, as can be seen in Figure 7.

The PD control of the intermediate error controller is structured as:

$$r_{des} = R_{euler \rightarrow pqr} \cdot \begin{bmatrix} (\Phi_{cmd} - \Phi_0) P^P_\Phi \\ (\Theta_{cmd} - \Theta_0) P^P_\Theta \\ \dot{\Psi}_{cmd} \end{bmatrix} \tag{10}$$

Where the rotation from euler angles to attitude rates $R_{euler \rightarrow pqr}$ is defined according to:

$$R_{euler \rightarrow pqr} = \begin{bmatrix} 1 & 0 & -\sin(\Theta) \\ 0 & \cos(\Phi) & \sin(\Phi)\cos(\Theta) \\ 0 & -\sin(\Phi) & \cos(\Phi)\cos(\Theta) \end{bmatrix} \tag{11}$$

In an intermediate step, the desired attitude rate is generated by subtracting the current body rates p^b_0, q^b_0, r^b_0 and applying derivative gain:

$$\dot{r}_{des} = \begin{bmatrix} P^D_\Phi \\ P^D_\Theta \\ P^D_\Psi \end{bmatrix} [r_{des} - \begin{bmatrix} p^b_0 \\ q^b_0 \\ r^b_0 \end{bmatrix}] \tag{12}$$

Lastly, the current body rate is subtracted for increment generation which forms the additional input replacing the zeros in Equation 14:

$$\dot{r}^{des}_{inc} = \dot{r}_{des} - \begin{bmatrix} \dot{p}^b_0 \\ \dot{q}^b_0 \\ \dot{r}^b_0 \end{bmatrix} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}_{inc} \tag{13}$$

This stage has been modified to only compute outputs for the actuator channels based on the desired increment and does not take into account attitude reference angles in the cost function. Instead, only the angular acceleration increment generated from the first stage optimiser is used.

$$\begin{pmatrix} ax & ay & az & \dot{q} & \dot{q} & \dot{r} \end{pmatrix}_{inc} \xmapsto[CA-2]{v_n \rightarrow u_s} u_s \tag{14}$$

The revised architecture can be seen in Figure 7. Despite the additional computation step, we verified and ensured that the sampling frequency for the optimization algorithm consistently remains above 200 Hz.

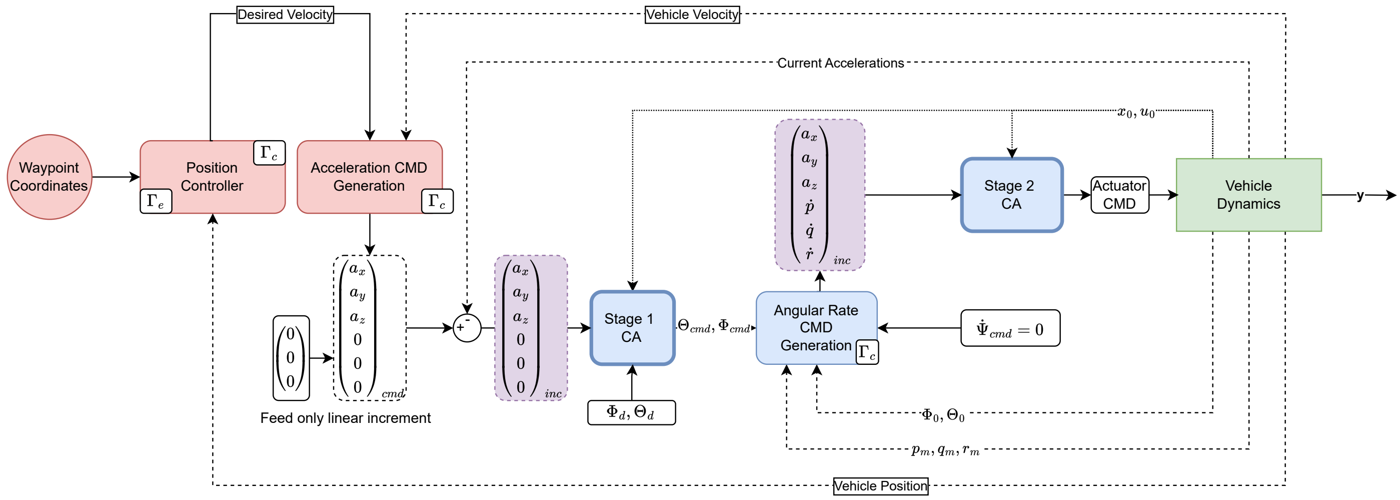


Fig. 7 The cascaded layout optimising attitude and actuators in different steps.

H. Additional notes on weights

In the light of the different weights presented, it should also quickly be discussed what effects the weights of Table 3 have on the behaviour of the drone in nominal and failure conditions. As a secondary objective is minimal consumption low during failure, the motor cost was an important factor besides the primary acceleration tracking. It was observed to influence the computed optimal attitude and motor orientation, where i.e. a too low value would result in the motors not pointing straight up during hover. A high motor cost aims to minimise the motor use and thus indirectly causes vertical alignment, making efficient use of the available thrust. The body angles and servo costs have a similar purpose during hover, both costs stabilise the system back towards pointing the motors and vehicle straight up. The ailerons were deactivated during the test.

It can be seen that several interactions are involved, and most of them change in case of a failure. During a motor failure, it is undesired for the drone to expend needless power to remain close to the hover reference attitude. Instead, it should find a new optimal orientation. To do so, the weights are adjusted as shown in Table 2. Lowering the weights on the body attitude gives the cascaded optimizer the opportunity to explore space far away from the initial hover configuration, without a large cost from the attitude deviation. Thus, the first stage optimiser as seen in Equation 9 is less constrained to find a new attitude far away from Φ_d, Θ_d

During the tests, the cost on the absolute use of servos was also set to 0, such that the motor orientation was optimised by taking into account the desired attitude and minimising the motor use.

I. Test Scenario

Being the most power intensive flight condition, failure of a motor during hover was selected as the primary failure case to be investigated. While forward flight offers the advantage of lift generated by the wing and the effective use of aerodynamic control surfaces, there is no such support during hover. Additionally, failure of a motor makes the respective gimbal servos ineffective as no thrust-vector is present anymore. With the identified thrust coefficient $K_T^P = 1.106465e^{-5}[\frac{N}{rad/s^2}]$ from Equation 5 the maximum thrust the propulsion can supply is about 44.3N. With a vehicle mass of 2.5kg, this means the hover thrust to weight ratio with all engines operable lies around 1.8. Failure of a singular engine thus reduces maximum thrust and T/W ratio to 33.2N and 1.35 respectively. Further motor failures would result in insufficient amount of thrust, not taking into account potential moments and linear accelerations to be generated.

This margin is essential for the 3 motor configuration as the controller needs to balance the vehicle into a new asymmetric thrust equilibrium and counter the torque generated by the co-rotating pair.

The considered failure case for the performed test flights is the failure of motor number 3 (back right). For the purpose of our test-flights, the system was informed of the failure by a switch activated by the pilot or a timed switch block in simulation. The drone was flown up to a pre-selected hover point. Once stable hover at the selected position has been achieved, the controller is informed of the failure of motor 3. The system response and on-board data was recorded and the failure was repeatedly introduced. A representation of the flight test plan can be seen in Figure 8 with the selected failure case of losing motor 3 (back right). Subsequently a waypoint tracking scenario was flown to check the maneuver capability with the three remaining motors.

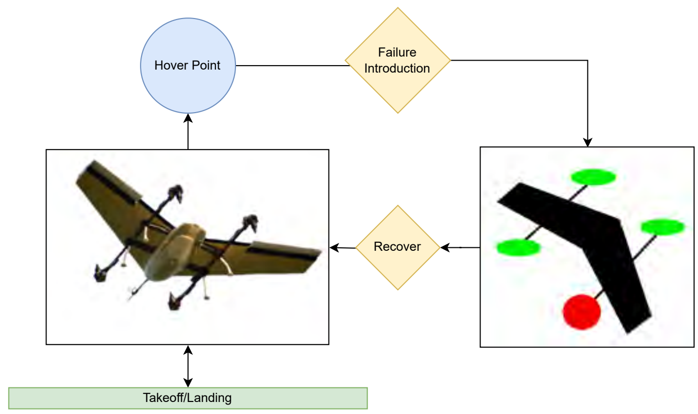


Fig. 8 Test flight representation of maneuver.

Table 3 Weight terms from Equation 6

	Component	Term	Effect
W_u	Motor	$W_{\Omega} = 20$	Penalize absolute motor speed.
	Servos	$W_{\delta_{az}}, W_{\delta_{el}}, W_{\delta_{ail}} = 0$	Penalize deviation from reference orientation (straight up motors in hover, 0 deg aileron).
	Body attitude	$W_{\Theta} = 50,\ W_{\Phi} = 100$	Penalize deviation from reference attitude.
W_v	Accelerations	$W_{ax}, W_{ay}, W_{az} = 0.05$ $W_{\dot{p}}, W_{\dot{q}} = 0.1,\ W_{\dot{r}} = 0.05$	Terms to weight which acceleration terms take priority in matching the references.
W_du	Motor	$W_{\Omega_{du}} = 20$	These terms penalize changes from the current actuator state.
	Servos	$W_{\delta^{du}_{az}}, W_{\delta^{du}_{el}} = 20,\ W_{\delta^{du}_{ail}} = 0$	
	Body Angles	$W_{\Theta_{du}}, W_{\Phi_{du}} = 25$	

III. Results and Discussion

Within this section, the data gathered from preliminary analysis of the problem, simulation and flight tests will be presented. Preliminary analysis was performed to identify potential attitude the vehicle might re-orient to under certain sets of optimization weights. This was used to achieve an estimate beyond the empirically theorized scenarios. Beyond this, the failure scenario was also performed on a dynamic model, using a mathematical representation of the vehicle and controller within Simulink. Finally, the compiled code was transferred to the onboard hardware and a set of test-flights performed with the real drone.

A. Optimal Attitude Identification

To determine the optimal attitude configuration with a motor fault, we analyzed the shape of the cost function presented in Equation 6, considering a fault in motor number 3 starting from an initial hovering flight condition ($\Phi_0 = 0$ and $\Theta_0 = 0$). For this analysis, the failure information was provided to the controller to activate the previously discussed changes in actuator constraints and weights. The desired acceleration increment was set to include only an upward component (z_c). To map out non-optimal sections, the constraints were confined to rectangular areas of the $\Phi - \Theta$ plane, providing an overview of the cost values relative to the vehicle's attitude.

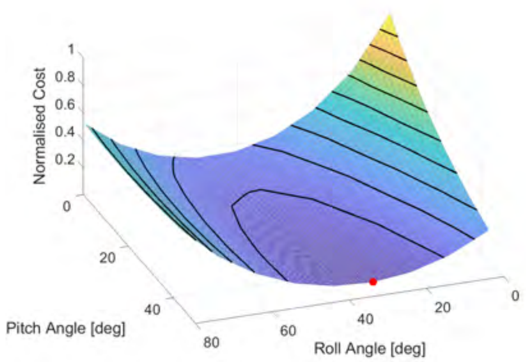


Fig. 9 Normalised solution cost w.r.t the body angles as computed from steady hover upon failure of motor 3.

It can be seen in Figure 9 that the controller attempts to orient to a pitch up attitude of 50° and around 35° right roll. For this figure, the cost was scaled to the range of computed values. Realistically, the points on this surface that reach close to 1 are infeasible due to inadequate tracking performance of the desired accelerations. The graph only provides an initial and instantaneous visualisation of the cost, limited to the two variables Φ, Θ. For the entire flight, the shape of this surface changes according to the desired accelerations, vehicle states and actuator states*..

B. Preliminary Simulation Results

With the existing Simulink model and proposed changes, the scenario was first tested in Simulation. The results still showed oscillation in transition to the new attitude, but stabilised the vehicle despite a failure of motor 3. The resulting attitude is close to what was expected from the initial investigation of recovery attitude in Figure 9. Differences to the optimum attitude can be explained by different actuator state and acceleration reference fed to the optimizer. The commands from the Simulink failure case are shown in Figure 10. There is still noise that was not be observed to that extend in Figure 13 and Figure 14, which implemented an additional filter on the accelerations as computed from the current actuator settings. The actuators assume a position that is in line with the co-rotating pair situation. Motors 2 and 4 generate a torque that needs to be cancelled, and the motors are reoriented to point upwards, compensating the body attitude. Motor 1 additionally points forward to counter the generated torque.

Past the initial simulation tests, the development primarily used the flight-test data as the step over to the hardware implementation did show performance differing from the offline simulation. As can be seen in the following section, the actuator deflections did show different behaviour especially for the gimbal of motor 1.

*For an animated version, recreated from the later presented test-flight data, see https://www.youtube.com/playlist?list=PLleZWIsVvmSBqVSoune02p6FpacddIxAU

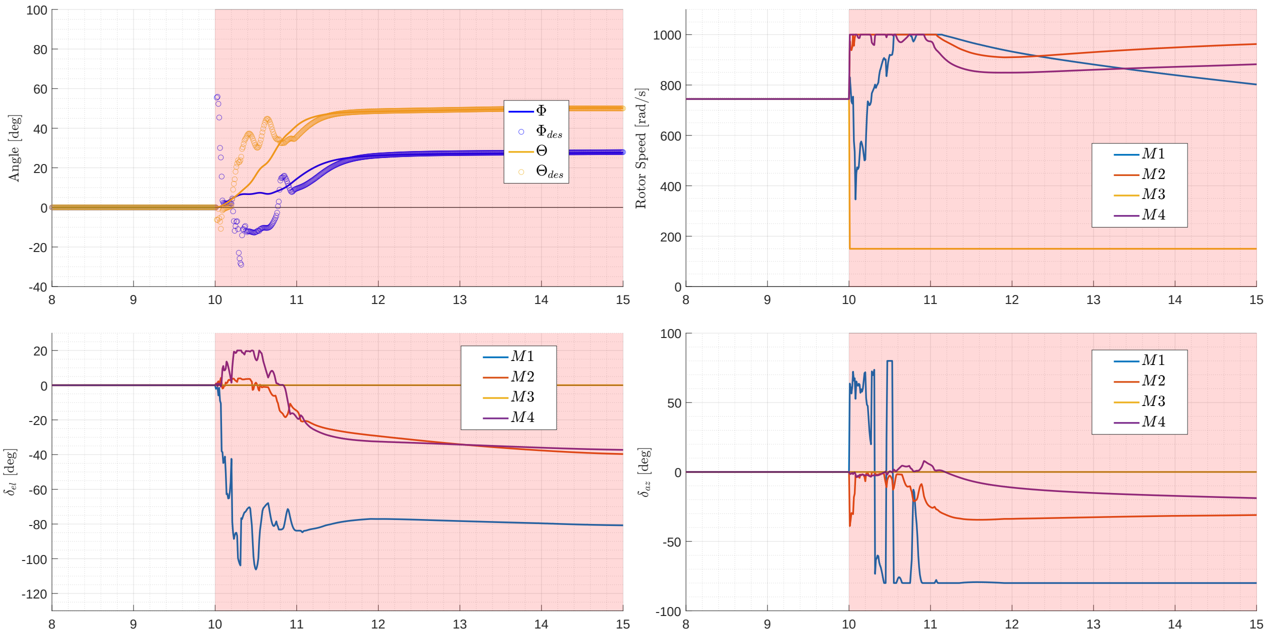


Fig. 10 Actuator and attitude commands as computed from the simulation run. Legends M1-4 denote commands. The angle plot has commands and response.

C. Flight Test Results

During the testflights the controller successfully stabilised the vehicle after failure was introduced on motor 3 and was able to follow waypoint changes. The resulting commands and other metrics logged by the onboard computer were analysed and are presented in this section. By analysing the flight test data from the repeated failure and the moving waypoint scenarios, the following observations can be made:

- Tracking of the desired waypoint position is consistent in both scenarios. In Figure 11 for the repeated failure and Figure 12 for the moving waypoint it can be seen that the drone attempts to match the desired position and succeeds with acceptable accuracy. The slight lack in precision can partially be attributed to the settings for the waypoint reference speed generation, which gets less aggressive the closer the vehicle is to the waypoint.
- With the desired thrust vector of motor 1 lying close to -90° servo elevation, it can be seen from Figure 13 in the first and third failure that the full range of azimuth 1 is utilised. From the cost function design, futile allocation to azimuth is prevented by the tertiary cost term.
- The servo angle responses during the repeated recovery test show a visible trend over the failures. As is evident from Figure 13, the response of the elevator servos shows a consistent pattern between failures. A similar trend can be observed for the azimuth servos in relation to the observed roll angle. For motor 1, the previously mentioned elevator servo state causes the patterns visible in the first and third failure as the thrust vector rotates in a tight cone.
- Looking at Figure 14, the continuous failure shows a similar stability in the solution during failure. The spikes in the servo angles can be traced to the waypoint location step changes visible in Figure 12, consistent with the way the acceleration reference is generated. There is no smoothing applied in case of a step change. The slight drifts, especially in azimuth of motors 2 and 4 can be connected to the changing body roll angle. The azimuth servos are compensating for the change.
- It can also be seen in Figure 14 that the controller is saturating the body pitch angle, despite the roll angle having a lower weight (Table 2). This and the initial overshoot indicate that a better attitude may yet be available beyond the limited pitch angle of 50°. The limitations on pitch and roll angles were established due to the current limitations of the controller internal reference frame conversion. As euler angle rotations are being used, the ZYX rotation order leads to undesired behaviour during pitch angles approaching the 90°. In this situation, the yaw and roll angle changes become synonymous, a highly undesirable ambiguity which the controller cannot handle.

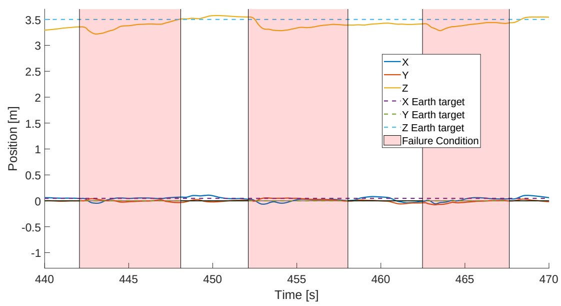


Fig. 11 Position tracking during repeated failure.

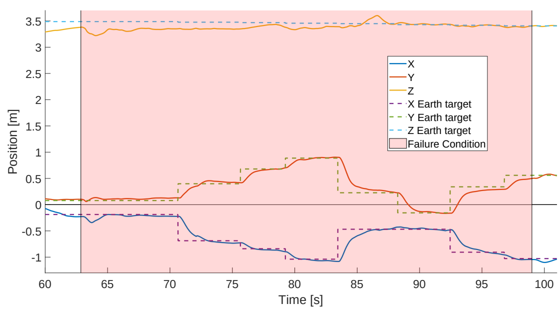


Fig. 12 Tracking position changes in XY-plane in failure.

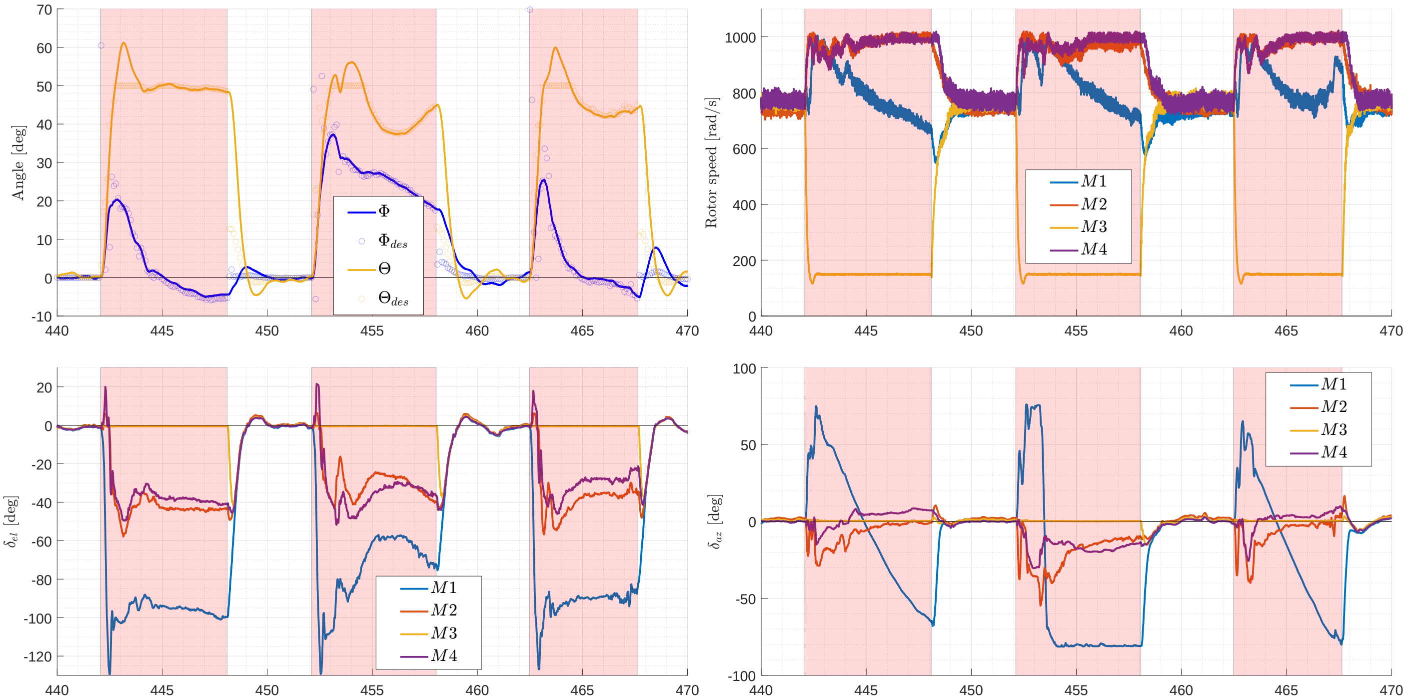


Fig. 13 Combined plot of actuator states and body angles during consecutive failures.

- It can also be seen from Figure 14 that the linear movement was mostly performed by the tilt servos as desired. There is still the indication of minor desired roll angle change but the cascaded CA limits excessive attitude changes. A major effect is also observed on the motor 1 speed in case of the largest position step. Changes in the waypoint position (Figure 12) cause temporary spikes, which is attributed to the additional required lateral thrust and changing body moments.
- The motor response in both scenarios has also been observed to be consistent. There is a slight outlier in the motor 1 speed during the last failure of the series in Figure 13, where rpm increases towards end of fault. It is visible that motors 2 and 4 are operating near complete saturation in both tracking and repeated failure scenarios. This is indicative of both: 1) a rather slim thrust to weight margin for the current condition; 2) a previously mentioned possible better orientation at body angles exceeding 50°. A less restrictive set of constraints could possibly utilise motor 1 to a greater extent, reducing overall motor saturation.

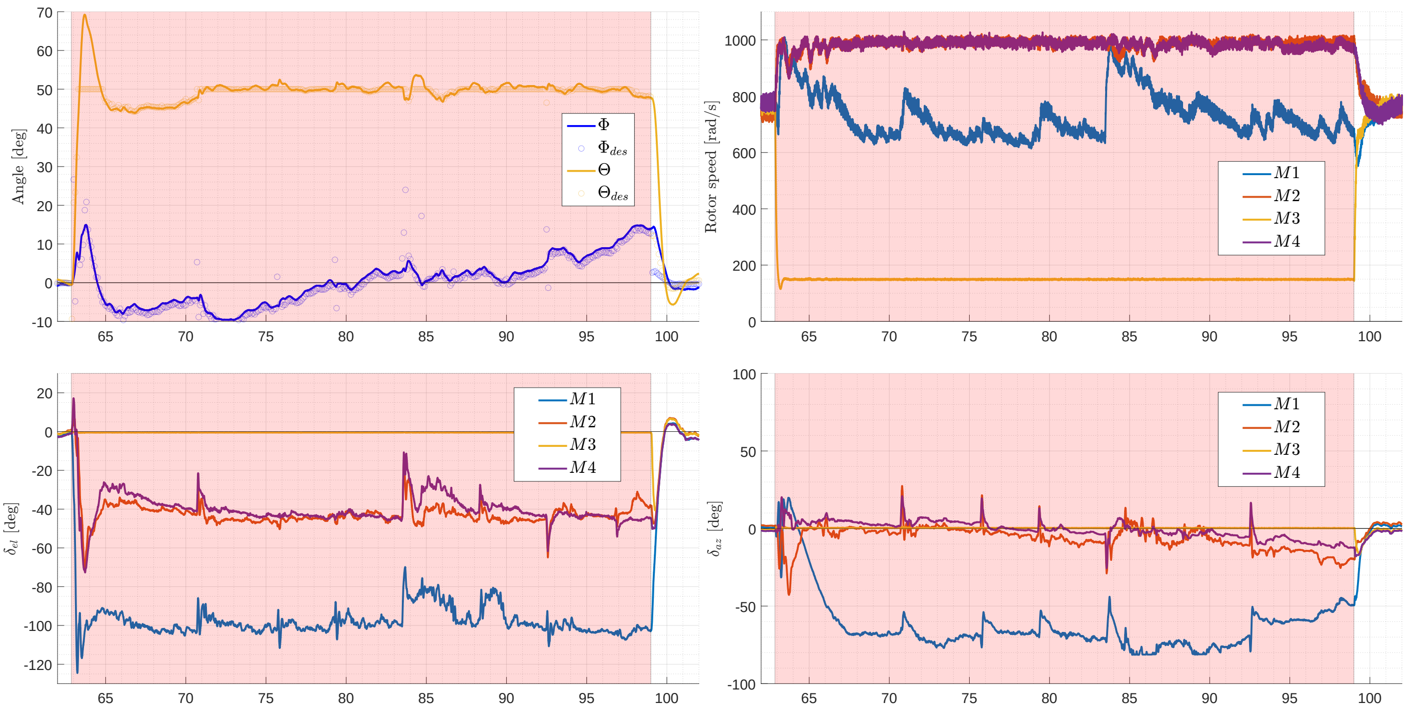


Fig. 14 Combined plot of actuator states and body angles during tracking task.

- In Figure 13 a difference between the desired and achieved body angles can be observed shortly after failures. An overshoot beyond the allowed 50° in pitch is observed. It can also be seen that during the transition to the new attitude and during the overshoot, the actuator commands show less steady behaviour. This is likely due to the dynamic nature of the maneuver dynamics and allocation problem compared to the established stable hover.
- It can be seen from 3D visualisations of the vehicle that motor 1 was used to both provide lift, and cancel out the yaw moment generated by the saturating co-rotating motors 2 and 4[†]. This is the primary reason for the difference in gimbal servo elevation when compared to the co-rotating pair.
- The thrust to weight ratio that was theoretically possible with the three remaining motors was calculated as 1.35. During the tracking maneuver, the vehicle used thrust in excess of the required 1.0 ratio. This may partially be influenced by inaccuracies in the propulsion model, or downwash interaction. The trend was expected due to the need to counter motor torque and balance of the cg according to attitude limitations. The largest step change in desired waypoint location caused a spike in motor 1 speed close to the maximum at 84 seconds into the flight.

D. Log Offline Analysis

Additional analysis was performed on the logged flight data to investigate possible model inaccuracies. The recorded state variables and control settings were fed to the offline optimization routine to generate offline commands based on the recorded vehicle data.

It was observed, that motor and tilt commands computed by the offline optimizer did diverge from the flight test, especially during reoriented flight and particularly for motor 1 Figure 15. The attitude did not show this problem. It is theorised that these inaccuracies may be caused by inaccurate estimation of the resulting vehicle dynamics within the model, as a similar trend in the computed actuator controls can be seen in Figure 10. Preliminary testing showed that changing the controller inertia estimates to a lower value improved the simulation accuracy compared to the flight data. Future work may focus on investigating the issue further and improving simulation accuracy. The combination of this points to a significant stabilising effect of the error controllers and acceleration filter on the aircraft.

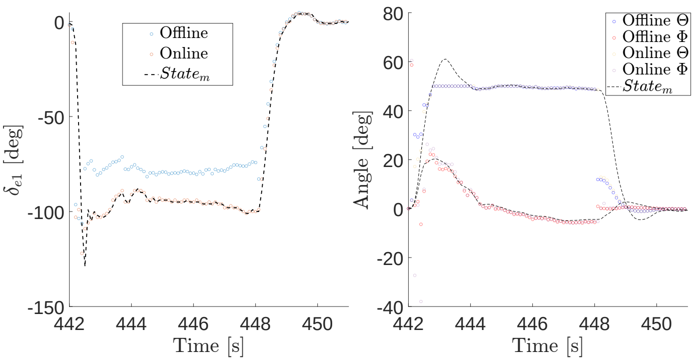


Fig. 15 Motor 1 elevation and angles from feeding the flight logs to offline routine.

Additionally, the power statistics from the test flight were plotted in Figure 16 to highlight the difference between the motor speed in Figure 13 and power efficiency. For a previous study it was suggested to make use of the current power consumption instead of speed values for the motor cost component [15]. This has proven to smooth out control allocation behaviour at higher motor speed as the resulting non-linear increase in power is taken into account instead. Future research should implement this metric instead to more accurately represent minimizing power consumption in hover.

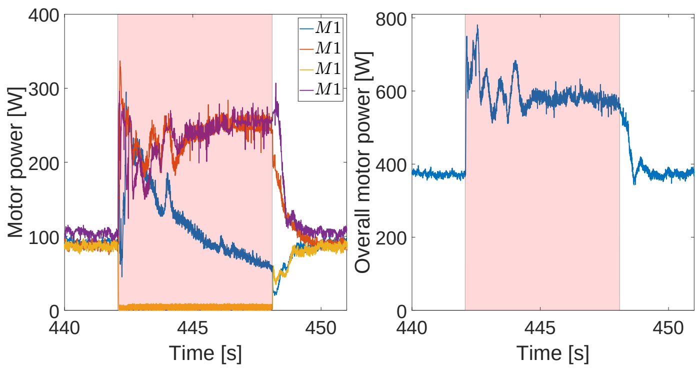


Fig. 16 Motor power consumption during failure.

IV. Conclusion & Recommendations

This project set out to investigate the capabilities of the proposed unified non-linear controller with SQP CA on the dual-tilt quadplane under actuator faults. Due to the nature of the vehicle, the fault of an engine in hover was chosen to be the prime failure case to be considered. Failure dependent constraints on the faulty actuator were introduced to inform the CA of the changed operating conditions. The addition of a control-effort tertiary cost term to counter excessively large steps, especially in early transition improved the stability of the solution and proved successful in preventing large jumps in the computed control solution at the cost of artificially slowed dynamics. With the separation of attitude and actuator command optimization the previously problematic feedback loop between desired attitude and desired angular acceleration increment was removed and the actuator orientation is now always computed with respect to the actual vehicle attitude instead of the desired one. This change removes undesired resulting accelerations from the difference between actual and desired attitude.

With these changes the vehicle has successfully recovered from repeated failures. Successful tracking of a changing reference waypoint was demonstrated with one failed motor. This paper highlights the potential of over-actuated tilt-rotor configurations in establishing new ways of recovery by utilizing the available thrust vectoring to re-orient the vehicle into a new hover configuration.

A. Limitations & Recommendations

The tests have been performed in a controlled indoors environment with limited disturbing factors that make a test in outside conditions more unpredictable. While the test flight campaign demonstrated repeated recovery, it is hard to guarantee the robustness for any optimization based system. A longer and more detailed flight test campaign may be required to map the recovery performance under a sufficiently large number of conditions.

Additional limitations of the project include the following points:

- No actuator dynamics model is considered when computing the

[†]https://www.youtube.com/playlist?list=PLleZWIsVvmSBqVSoune02p6FpacddIxAU

optimal configuration. While the added γ_{du} cost component and the cascaded architecture mitigate undesired acceleration components due to desired/actual mismatch, future iterations may investigate the possibility of more direct rate limits.

- It is possible that there are still more efficient hover attitudes available under failure. The available angles in pitch and roll were limited due to possibility of euler angle gimbal lock. With the ZYX rotation order, pitch angles close to or exceeding 90° cause instability in the controller. A refactor to a quaternion based system, or switching roll and pitch order could open up new possibilities.
- The tests were limited to the critical failure case of motor 3, more tests should be performed on different (types and combinations of) actuator failures. Future test campaigns should also include failure scenarios in forward flight mode or transition.
- For future simulations of the vehicle it may be insightful to include the OrangePi in a Hardware-in-the Loop (HIL) setup. This and testflight logs could be used to further improve simulation accuracy and identify potential model improvements.
- With the current controller setup, it is not possible to assign actuators to certain control objectives. In the future it might be investigated how i.e. the desired yaw has to be achieved with only gimbal azimuth.
- The control allocation on the motors was chosen for indirect control of gimbal alignment. It was previously mentioned that the cost of motor use should in the future be based on consumed power instead of achieved motor speed.
- This study focused heavily on the mitigation of a motor failure, but stuck servos and other failure scenarios should be looked at in real test-flights in the future.

References

[1] Kamel, M., Verling, S., Elkhatib, O., Sprecher, C., Wulkop, P., Taylor, Z., Siegwart, R., and Gilitschenski, I., "The Voliro Omniorientational Hexacopter: An Agile and Maneuverable Tiltable-Rotor Aerial Vehicle," *IEEE Robotics amp; Automation Magazine*, Vol. 25, No. 4, 2018, p. 34–44. https://doi.org/10.1109/mra.2018.2866758, URL http://dx.doi.org/10.1109/MRA.2018.2866758.

[2] Myeong, W., and Myung, H., "Development of a Wall-Climbing Drone Capable of Vertical Soft Landing Using a Tilt-Rotor Mechanism," *IEEE Access*, Vol. PP, 2018, pp. 1–1. https://doi.org/10.1109/ACCESS.2018.2889686.

[3] Zheng, P., Tan, X., Kocer, B. B., Yang, E., and Kovac, M., "TiltDrone: A Fully-Actuated Tilting Quadrotor Platform," *IEEE Robotics and Automation Letters*, 2020. https://doi.org/10.1109/lra.2020.3010460.

[4] Junaid, A. B., Sanchez, A. D. D. C., Bosch, J. B., Vitzilaios, N. I., and Zweiri, Y., "Design and Implementation of a Dual-Axis Tilting Quadcopter," *Robotics*, 2018. https://doi.org/10.3390/robotics7040065.

[5] Mousaei, M., Geng, J., Keipour, A., Bai, D., and Scherer, S., "Design, Modeling and Control for a Tilt-rotor VTOL UAV in the Presence of Actuator Failure," *IEEE/RJS International Conference on Intelligent RObots and Systems*, 2022. https://doi.org/10.1109/iros47612.2022.9981806.

[6] Mancinelli, A., Smeur, E., Remes, B., and Croon, G., "Dual-axis tilting rotor quad-plane design, simulation, flight and performance comparison with a conventional quad-plane design," *ICUAS 2022*, Institute of Electrical and Electronics Engineers (IEEE), United States, 2022, pp. 197–206. https://doi.org/10.1109/ICUAS54217.2022.9836063.

[7] Jiang, J., Zhang, Y., and Yu, X., "Fault-tolerant control systems: A comparative study between active and passive approaches," *Annual Reviews in Control*, 2012. https://doi.org/10.1016/j.arcontrol.2012.03.005.

[8] Jain, T., Yame, J. J., and Sauter, D., "Active Fault-Tolerant Control Systems: A Behavioral System Theoretic Perspective," *null*, 2017. https://doi.org/10.1007/978-3-319-68829-9.

[9] Abbaspour, A., Mokhtari, S., Sargolzaei, A., and Yen, K. K., "A Survey on Active Fault-Tolerant Control Systems," *Electronics*, Vol. 9, No. 9, 2020. https://doi.org/10.3390/electronics9091513, URL https://www.mdpi.com/2079-9292/9/9/1513.

[10] Wang, X., and Sun, S., "Incremental fault-tolerant control for a hybrid quad-plane UAV subjected to a complete rotor loss," *Aerospace Science and Technology*, 2021, p. 107105.

[11] Bouabdallah, S., and Siegwart, R., "Full control of a quadrotor," 2007, pp. 153 – 158. https://doi.org/10.1109/IROS.2007.4399042.

[12] Sun, S., Wang, X., Chu, Q., and de Visser, C. C., "Incremental Nonlinear Fault-Tolerant Control of a Quadrotor with Complete Loss of Two Opposing Rotors," *arXiv: Robotics*, 2020. https://doi.org/10.1109/tro.2020.3010626.

[13] Zhang, W., Mueller, M. W., and D'Andrea, R., "Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space," *Mechatronics*, Vol. 61, 2019, pp. 117–130. https://doi.org/https://doi.org/10.1016/j.mechatronics.2019.06.004.

[14] Mancinelli, A., Remes, B. D. W., de Croon, G. C. H. E., and Smeur, E. J. J., "Unified incremental nonlinear controller for the transition control of a hybrid dual-axis tilting rotor quad-plane," , 2023.

[15] Wechtler, N., "Implications of Propeller-Wing Interactions on the Control of Aerodynamic-Surface-Free Tilt-Rotor Quad-Planes," , 2024. URL http://resolver.tudelft.nl/uuid:3193131c-6b68-46a2-afe1-964a044dd6f9.

[16] Murillo, M. H., Limache, A. C., Fredini, P. S. R., and Giovanini, L. L., "Generalized nonlinear optimal predictive control using iterative state-space trajectories: Applications to autonomous flight of UAVs," *International Journal of Control Automation and Systems*, 2015. https://doi.org/10.1007/s12555-013-0416-y.

[17] Islam, M., Okasha, M., Okasha, M. E. A. A. E., and Sulaeman, E., “A Model Predictive Control (MPC) Approach on Unit Quaternion Orientation Based Quadrotor for Trajectory Tracking,” *International Journal of Control Automation and Systems*, 2019. https://doi.org/10.1007/s12555-018-0860-9.

[18] Schittkowski, K., “On the convergence of a sequential quadratic programming method with an augmented lagrangian line search function,” *Mathematische Operationsforschung und Statistik. Series Optimization*, Vol. 14, No. 2, 1983, pp. 197–216. https://doi.org/10.1080/02331938308842847, URL https://doi.org/10.1080/02331938308842847.

[19] *Sequential Quadratic Programming*, Springer New York, New York, NY, 2006, pp. 529–562. https://doi.org/10.1007/978-0-387-40065-5_18, URL https://doi.org/10.1007/978-0-387-40065-5_18.

Nomenclature

List of Abbreviations

ARX	Autoregressive and Exogeneous
FDD	Fault Detection and Diagnosis
FDI	Fault Detection and Identification
FTC	Fault Tolerant Control
HVTOL	Hybrid Vertical Takeoff and Landing
INDI	Incremental Nonlinear Dynamic Inversion
MMS	Multi Model Switching
MPC	Model Predictive Control
MPPI	Model Predictive Path Integral
PIM	Pseudo Inverse Method
SoC	System-on-Chip
SQP	Sequential Quadratic Programming
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Takeoff and Landing

List of Figures

List of Tables

Part II

Literature Review

2
Executive Summary

With the ever increasing advancement of drone technology, creating configurations that combine the advantages of vertical take-off and landing and efficient forward flight is a field of ongoing research. Whilst regular fixed configuration multicopters have high maneuverability, fault tolerance (depending on arrangement) and overall high versatility, they lack the passive lift generation of conventional aircraft and thus their endurance and speed. Overactuated hybrid Vertical Takeoff and Landing (VTOL)s are thus of interest for several applications and the options and their capabilities and high potential for fault tolerance are to be investigated.

Preliminaries

For the specifics of the vehicle, please refer to Mancinelli's earlier reports [1]. We consider a quadplane with 12 inputs resulting in a 2-axis tilt quadrotor. The motors can each be tilted by two servos and change their attitude in elevation and azimuth. The front two propellers are arranged to tilt into a pull configuration, whereas the two rear rotors are pusher props. This layout has a total of 12 actuators $(\Omega_1, \Omega_2, \Omega_3, \Omega_4, b_1, b_2, b_3, b_4, g_1, g_2, g_3, g_4)$ representing the rotational speeds of the 4 rotors and elevation/azimuth of their mounts respectively, to independently control the 6 degrees of freedom of the quadplane. Having more actuators than degrees of freedom to be controlled makes the system overactuated. The crux of overactuated systems is their control allocation, especially under fault conditions. Whilst some preliminary research exists for fully actuated quadcopters and a single axis tilt quadplane [2, 3, 4], there is a fairly distinct lack of existing investigations compared to fixed rotor configurations. This literature research was thus focused on three main aspects:

- Fundamental layouts for control schemes in Fault Tolerant Control.
- Different implementations of control allocation for overactuated vehicles.
- Methods for Fault Detection and Identifications.
- Identification of the gaps for such vehicles control implementations to retain control under failure. These shall ideally make used of the over-actuation to improve the fault tolerance compared to classical fixed layout configurations.

Fault Tolerant Control schemes

Fault tolerant control establishes the basis of accommodating degradation and failure of actuators, sensors and other system inherent components that effect the nominal system behaviour. In the field of overactuated drones with non-linear actuators, the focus of current research lies on **active** fault tolerance, in which the system relies on a means to timely and correctly identify faults. This information is then passed on to the active controller which adjusts the control strategy according to the informed fault. Passive fault tolerance by contrast does not rely on an additional supervisor and is instead concerned with robust and reliable controllers that work across a large region of possible failures.

Control Allocation

Control allocation is concerned with distributing the required forces and torques across the available actuators to achieve the desired action. The way that the allocation is done can be different, depending on the system. For the quadplane under consideration, methods involving linearised control effectiveness matrices have proven to be only of limited viability due to their limitations concerning non-affine systems. For some combinations of vehicle and actuator state, the control effectiveness matrix may contain small values, causing the actuator solution to be far away from the current actuator state, invalidating the system linearisation. As there is no information about the non-linearities of the actuators in the effectiveness matrix, the implementations have led to sub-optimal control solutions for static hover and oscillations for movement/rotation commands causing system instability. Limiting the maximum actuator control steps can improve stability, but introduces intolerable delays in the tracking response [1].

Several indirect approaches use optimisation based methods, which have proven to be the most powerful implementations making use of dynamic and nonlinear control allocation schemes. The current drone directly computes the control input from on optimised costs function imposed on actuator inputs and control objective, where the control objective takes much higher priority. The resulting constrained nonlinear optimisation problem is then solved by Sequential Quadratic Programming. The way the optimisation is done has a high impact on overall system performance and different norms can be applied to designing the cost function. These include $l1$, $l2$ and l_∞, which try to optimise for the total sum of input vector elements, the square root of the summed square elements and the magnitude of the largest input element respectively. Overall $l2$ has been most commonly applied as it gives the most naturally distributed solutions.

Fault Detection & Identification

As previously mentioned, the active approaches most commonly applied to problems in literature require reliable and robust Fault Detection and Identification (FDI) supervisors to work. If a system experiences a failure, but is not informed of it, the fault will still result in (catastrophic) failure of the entire system. Many different approaches exist to detect failures in actuators which can be roughly broken down into model based approaches and signal processing approaches. Model based methods compare a predefined or learned model against the system output and perform failure detection via residual thresholds. On the other hand signal processing aims to identify failures without a dedicated mathematical system model. Analysis of system vibration features and deep learning approaches have been successfully applied in previous studies, successfully localising errors within the sensor system [5].

Benchmarking & Future Research

The different implementations that have thus far been run on the prototype have been tested in simulation and the results were verified in flight tests. The comparison of the two allows to reflect on the accuracy of the model and possible future performance. To prevent damage to the aircraft the tests were performed whilst suspended from a security harness [1].

To assess the real world performance of the Fault Tolerant Control (FTC) implementation, a similar approach will be used. With the literature study complete and a preliminary research plan established, a starting point for the thesis is set. The research questions that shall be answered are presented and revised in the main body, as are the rough planning and work packages for the thesis. Investigation of additional literature will continue alongside the development of the prototype controllers, with the initial stages being the conceptualisation of different possibilities and coding and testing of them in software. Once the simulations yield desirable performance, the tests will be extended to the overactuated quadplane for flight tests. The gathered data and developed code, as well as steps taken, conclusions drawn and adjustments made are to be well documented for the purpose of delivering a structured and organised report including the codebase.

3
Introduction

As an introduction of why transforming configurations such as the considered vehicle are an important field of past and ongoing research, this chapter will explore the development of VTOLs and the change towards overactuated eVTOLs historically, with a mention of the developments in the area of drones and fault tolerance. An initial set of questions for the research project will be presented and revisited in later chapters.

3.1. History of VTOLs

Despite the history that helicopters and aircraft share, it has been a long standing challenge to develop systems combining their advantages. A good overview of earlier attempts is given by a NASA report from the 80s [6]. The idea of hybrid aircraft has been around for decades as stakeholders quickly realised that combining the advantages of helicopters and fixed wing aircraft in one machine would lead to a highly versatile platform. Despite this versatility, few concepts have made it into production, with big limiting factors being the inherent complexity and cost of such systems that also require intensive maintenance and training for crews versed in helicopter and aircraft operations.

Several different concepts have previously been considered to generate the thrust components for both vertical lift flight and propulsion under aircraft configuration. Interestingly they have gone through a similar evolution as can currently be seen in the drone sector, as the earlier designs make use of dedicated or limited reconfiguration lift solutions. The lower complexity of two effectively separate systems making them much easier to implement. Several configurations feature one or several sets of engines that only operate in hover mode and do not contribute to airplane mode flight. The more efficient configurations can be seen in what has eventually made it into production in the form of the tiltrotor V-22 Osprey in Figure 3.1. The advantages of combining high-speed, high efficiency flight with the capability of vertical take-off and landing in small or otherwise restrictive areas that conventional aircraft could not reach are of high value to military operations, but as can be seen by the development of the AW609 and several start-ups, there is spillover into the civilian and emergency service sector.

Figure 3.1: V-22, Credit: NASA/Sean Smith

3.2. The Rise of Drones

In the scope of UAVs, the trend to combine hover and classical aircraft flight has also spiked interest and research is underway to optimise both their layout and control. Due to their smaller scale and previously proposed concepts, the idea of a tiltrotor system is expanded to fully actuated thrust generation, enabling the vehicle to use thrust vectoring. This capability of individually thrust vectored rotors increases the maneuverability of the vehicle and allows to control orientation and translation independently, within the limits of the actuators. This over-actuation necessitates the implementations of more efficient and goal oriented control allocation schemes, which form one of the pillars of this field of research.

As for the additional capabilities, the vehicle can change its orientation in space such that the aerodynamic forces and moments generated by the lifting body are mitigated by realigning the lifting body or changing the thrust vectoring. This enables these vehicles to exhibit increased performance in one of the shortcomings of more conventional Hybrid Vertical Takeoff and Landing (HVTOL) systems, which tend to struggle more in the presence of gusts and windy environments in general [1]. This is especially interesting for vehicles that have to perform precision landings in turbulent environments, such as the aft deck of a ship, where the wind conditions tend to be very turbulent in the wake of the superstructure [7]. It also opens up possibilities for payloads that would not, or only to a lesser degree work on a classical multicopter platform. Other studies aim to increase the capabilities of copters to traverse tight spaces by matching their body orientation to the available space [8]. Whilst speculative, it is likely that the increased versatility of such systems may give rise to their popularity in the future, especially considering the fact that such configurations can be realised much simpler for comparatively small scale drones. If the development takes off, it is thus likely to see it first implemented in drones with a potential trickle into larger scale systems later on.

3.3. Fault Tolerance

With the increase in system complexity and performance, modern control systems face the problem that malfunctions may cause the entire system to show unsatisfactory or even unstable behaviour. These malfunctions are commonly termed *faults* and defined as unpermitted deviation of at least one characteristic property or parameter of the system from acceptable [...] condition [9]. In order to accommodate for such shortcomings in the system, it is necessary to design control systems capable of reacting to faults in the system.

Several projects have considered FTC for more classical multicopter configurations, but few have investigated the application to overactuated systems, especially drones with tilting rotors. The existing research on this will be investigated and form the focal point of this literature study.

3.4. Preliminary Questions and Goals

Going into the project, the goal for the thesis was already set to *develop an automatic rotor failure and mitigation strategy for the dual-axis tilt-rotor quad plane*. The means to do this are the supplied simulation models and code in Matlab/Simulink and later flight testing on the existing vehicle. The data generated by the simulation and flight tests shall be used to establish a performance rating of the considered control approaches and evaluate them according to a defined scheme similar to methods proposed by Edwards in [10]. The major idea is to investigate existing approaches both in overactuated vehicles and more conventional multi-rotor layouts, general theory on fault tolerant control and projects which apply the different methods to relevant configurations. After presenting the reviewed literature and background a bit more in the following chapters, the research questions will be revised, or rather specified to more accurately outline the research after completion of the literature study.

The preliminary questions are as follows and will be revisited in Chapter 6 to be shaped into adequate research questions:

- What control methods can be applied to the 2-axis tilt quadplane to implement fault tolerant control?
  - How well do the different methods deal with highly non-linear systems?
  - Are there better FDI approaches available, that can be adapted to the vehicle?
- What viable methods can be implemented in Matlab/Simulink and tested on the vehicle?
  - How much effort does integration take, i.e. can it be build on the existing control framework or require heavy re-structuring/programming?
  - Does the control hardware have sufficient computation reserves to run the candidate approach?
- What is the performance of the selected method(s)? (How do they compare to each other, and the simulation results vs. flight test?)
  - What kind of testing framework and weights can be used to evaluate the performance of the different approaches if several are tested?
  - How can one best visualise the controller performance (deviations, control actions, flight paths etc.)?

4

Fault Tolerance in Drones

Having previously established the benefits of VTOLs and HVTOLs specifically, this section aims to identify past and ongoing research in the major investigative point of this review: Fault Tolerant Control. Many past research papers exist for quadrotors and other fixed configuration systems, in addition to books and papers on fundamental theory and underlying frameworks of fault tolerant control. This chapter will recap this fundamental theory, investigate some of the projects and approaches in more detail and close with an outlook towards the field that this knowledge is later to be applied to: Over-actuated drones. In order to better understand FTC, it may be advantageous to first distinguish between the two major and generally agreed upon methods used to achieve fault tolerance and what characterises a fault.

4.1. FTC - an introduction

The literature suggests that FTC can be broken down in two major categories: **active FTC** and **passive FTC** methods [11]. The major differences lie in the way these approaches deal with an occurring failure. Active FTC is reliant on additional FDI mechanisms to register when and where a fault occurrs and reconfigure the flight controller accordingly. Good introductions to the topic are given by Boskovic [12] and Yu [13]. Mazeh et al. investigate the performance of online vs. offline FDI solutions, with significantly less computational effort for the offline solution, but much smoother results and recoveries for on-line identification [14].

Passive FTC is fundamentally a branch of robust control. Suboptimal performance across the flight envelope is accepted in order to design a single controller, that is capable of dealing with different failure modes. These controllers have to be tuned such that they have enough margins to account for the different possible failures, an approach that may quickly problematic for over-actuated systems such as the one under consideration. As such, and because the focus of current research papers mostly lies on optimising the function of FDI and active FTC, the passive methodology will take a backseat to the active approaches.

4.1.1. Types of Faults

There are several types of faults that have to be considered for these approaches and though possibly obvious, for the sake of completeness and to paint a more complete picture in the readers mind, a quick overview is to be given. Again, Jain [15] discerns three categories that are often used conventionally in the FTC community:

- *Actuator Faults* (af): Any type of partial or complete loss of control actions.
- *Sensor Faults* (sf): Incorrect sensor readings, can also be partial or complete.
- *Component Faults* (cf): Summarizes all other faults that cannot be assigned to either of the two previous categories, an example would be structural damage. Faults that fall into this category are generally regarded to be the hardest ones to accommodate. These are also referred to as Plant Faults [16].

Faults occurring in systems can have different characteristics. In time domain, they can be characterised such that they fit in either of the following categories: **abrupt**, **incipient** or **intermittent**.

Abrupt Faults, as the name suggests are characterised by a sudden change in system parameters, which happen faster than the nominal dynamics and are commonly hard to detect via residual based methods [16]. Examples of these can be vibrations, electrical faults, or structural breaks [17].

Incipient Faults are characterised by a failure over time nature, instead of suddenly introducing a major change, the fault occurs gradually, degrading system performance over time. The causes for this tend to lie in sensor or actuator inaccuracies or partial failures and may be hard to detect by residual based methods.

Intermittent Faults effect the system at irregular intervals and can be caused by i.e. faulty connections between sensor and actuator hardware and controller. These types of faults are a fairly common occurrence in any system and their likelihood increases with system complexity. Their inconsistent time characteristic poses a challenge to most FDI systems.

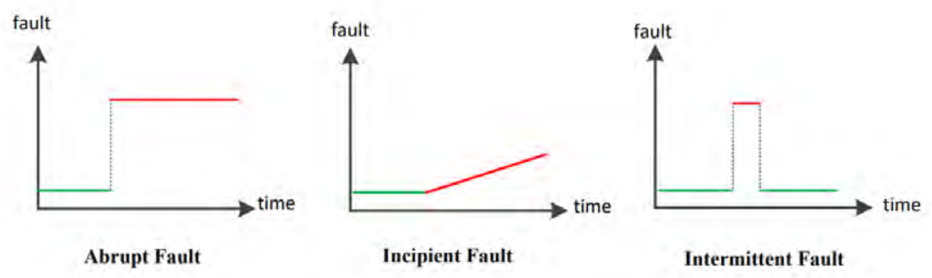


Figure 4.1: The different time characteristics of system faults [15, 16].

4.1.2. FDI Methods

As previously mentioned, active FTC control systems rely on different methods of identifying faults. These FDI mechanisms are also sometimes termed *supervisors* and are of high importance to the correct functioning of the system. If a fault is not detected and communicated, a controller that attempts to make regular use of a faulty actuator may very well lead to complete loss of the system. This is obviously undesirable, so efforts to develop fast, reliable and lightweight FDI implementations is a field of ongoing research.

Chen et al. have investigated the use of neural networks. Their methods makes use of neural networks as inputs for a residual comparison and thus error detection. According to them, their work can be adapted to non-linear systems due to the versatile nature of neural networks [18]. Their approach utilises the system states and inputs and feeds them to a neural network generating residuals for fault detection. Neural networks have also previously been used in the identification of certain maneuvers a drone performs from historic data [19], proving the merit it may have for direct fault detection implementations. Going one step further, Sadhu et al. [20] made use of a set of trained nets to perform anomaly detection and classification in a layered manner, achieving 90% detection and 85% classification accuracy respectively.

Genreally, FDI techniques can be divided into two major groups:

- **Quantitative Model-Based Approaches** make use of state or parameter estimation and parity space approaches.
- **Qualitative Model-Based Approaches** instead use AI based methods such as pattern recognition.

It can be said that FDI tends to be fairly case specific and requires knowledge about the system and possible approaches to successfully integrate a full fault tolerant control scheme. In the octo-quad example below Mazeh et al. present a parity space based approach [14]. A parity space is in essence a multidimensional space that spans the residuals of the detection algorithm [21].

Similarly, Miksch et al. have investigated different active FTC schemes in regards to their real-time performance [11]. Summarily, the different approaches of Model Predictive Control (MPC), Fault Tolerant Least Squares and Pseudo Inverse compared such that MPC does yield the highest control performance, but at the highest computation cost and intensive use of inputs due to its step-wise optimising nature and potential model/pant mismatches. According to them, whilst being the most computationally expensive method, it is also the method applicable to the most failure scenarios, with especially Pseudo Inverse Method (PIM) not being able to accommodate nonlinear faults such as saturation.

Example of Quad Layout FDI

For the detection and identification of faults in classical multicopters, several different approaches have been proposed and tested in literature. Mazeh et al. propose a residual based observer for their octo-quad, that identifies the failed engine by means of an identification table [14]. Each rotor failure leads to a unique combination of the three Euler angle residuals, generated from internal measurement unit readings and a sliding mode observer, which can be used to identify which of the actuators has failed. This does not take into account any type of sensor failure, but the identification has proven reliable for real world test flights.

Application to the quadplane: *Whilst it may be possible to create some kind of lookup table, due to the different orientations in space that the individual rotors can take, the identification may not be possible with equal certainty. However, considering that the flight condition for the flight tests and simulation will be a hover, the elevation and azimuth angles of the individual rotors will still be such that an identification via this method seems feasible to implement!*

Table 4.1: The Identification Table as presented by [14]

Motors	Error Sign		
	Roll	*Pitch*	*Yaw*
No. 1	+	+	-
No. 2	+	+	+
No. 3	+	-	+
No. 4	+	-	-
No. 5	-	-	-
No. 6	-	-	+
No. 7	-	+	+
No. 8	-	+	-

For this table, the errors are defined as the actual measurement minus the predicted output of the supervisor. In order to understand better the layout of the system, the schematic layout of the drone is presented in Figure 4.2.

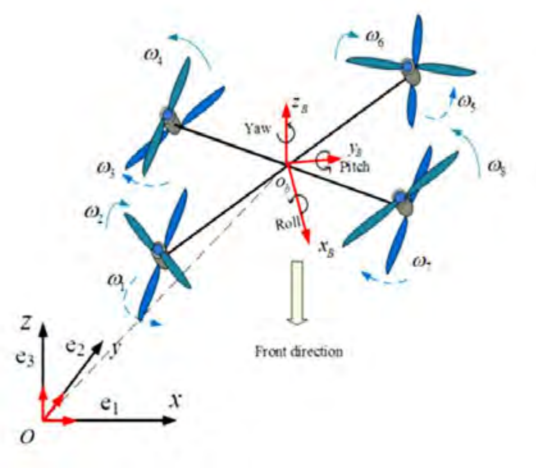


Figure 4.2: The motor layout of the octo-quad as presented by Mazeh et al. [14]

As mentioned previously an approach such as this could be used on the quadplane in hover mode, pending reasonable variations in thrust vectors. A possible issue however is the detection of faults in the servo hardware responsible for rotor tilt. Their impact would likely not be as prominent in a ”stuck serve” scenario and detection of their failure is unlikely to be easily possible by a similar categorisation method.

Other options for this type of failure identification would need to be considered, which could include a more complex model based approach.

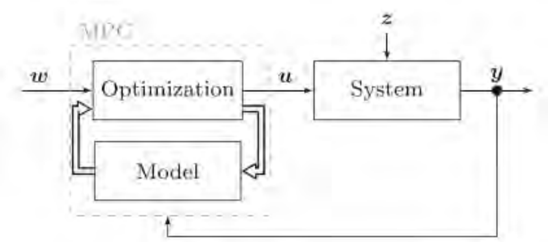
4.1.3. Thesis Relevance of Faults

For the consideration of the thesis, the considered failure cases will have to be limited to a certain set of cases for testing. As with previous master study subjects for drone FTC and recovery by Sun, Baert et al. [22], the prime case to be considered is the complete failure of one of the motors. This will effectively eliminate 3 degrees of control, as the elevation and azimuth servos for an ineffective rotor become irrelevant. The additional failure of one of the rotors thrust vectoring leads towards the second bigger topic of constructive re-orientation of the vehicle. Further investigations will then show how far considered controllers can be pushed and whether edge cases like upset recovery are possible. In the following sections a more detailed overview of some of the fault tolerant control methods will be given.

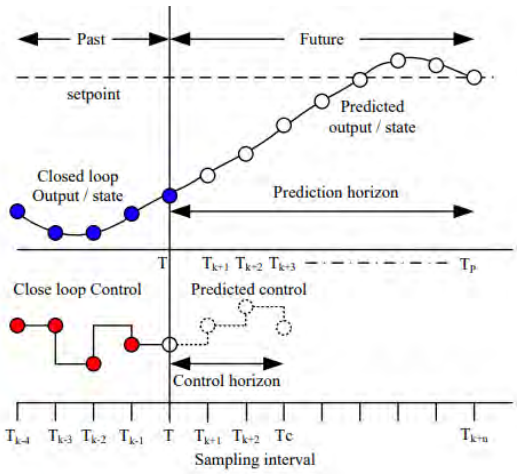
4.1.4. Model Predictive Control

Model Predictive Control is an optimisation based technique with several sub-categories in problem approaches, one of them being sampling based Model Predictive Path Integral (MPPI) [23]. Fundamentally, it describes a set of control methods, which make use of representative system models to predict the future behaviour of the system. By constraining this optimisation problem, MPC can implicitly generate the control law and lays a heavier focus on properly identifying and modeling the system [24] to improve control performance. The approach can be adjusted to handle constraints by giving limitations to the available variables for the optimisation problem, which is the underlying control allocation method of MPC. It computes a minimum cost input sequence that tracks the desired output as closely as possible whilst trying to minimise the attached cost function. It is important to pay attention to the computation times and termination conditions for the optimiser to not induce undesired behaviour in the system due to excessive computation times.

The computation of control inputs is done depending on different horizon settings. In general three different timesteps are considered: the control horizon N_u, the prediction horizon N_2, and sometime the lower prediction horizon N_1. N here indicates sampling based indexing of time. The control horizon determines how far into the future the control sequence is computed, the prediction horizon does the same for the system output and the lower prediction horizon is responsible for shifting the control horizon a certain number of timesteps to account for delays and computation time.



(a) A Block Diagram of MPC control logic [24]



(b) Time history and horizons in MPC [25]

Figure 4.3: MPC control block diagram (left) and control time scheme (right).

MPC has been proven as a computationally rather expensive, but highly adaptable and well performing controller. It can also fairly easily be adapted to fault tolerant control by informing the controller of actuator failures, adapting the input constraints of the optimisation.

Controllers that utilise MPC have the benefit of reacting to future control objectives, set-points or obstacles compared to purely reactive approaches. Their optimisation problem can also be set up to account for a variety of system constraints, such as input(-rate) limits and constrained states. The biggest issue that MPC faces is the reliance on high quality system models and long computation times for the optimisation at each time-step. Oftentimes other means like chained Incremental Nonlinear Dynamic Inversion (INDI) controllers are used to correct for flaws in the solution that arise from model inaccuracies. As a subfield, MPPI relies on the random sampling of control sequences instead of optimising one actively, which tends to cause noisy computed input signals.

4.1.5. General Optimisation Based Approaches

Model Predictive control is only a very specific implementation of an optimisation based controller. The beauty of this type of controller is that it can be applied to any properly formulated problem and come up with a constrained objective optimised solution. Often this involves hand tuning the weights of more complex systems and making smart choices about cost priorities. Additionally it is essential to have a good mathematical representation of the system, as usually the weights are assigned to balance between control effort and reached state accuracy.

Another focus area of these type of controllers is the choice of proper optimisation algorithms, focusing on lightweight yet powerful implementations to be able to run them on the respective on-board systems. Previously this has been a large limiting factor, especially for very (parallel) computation intensive approaches such as MPPI control. With the advance in high performance small form-factor chip-sets, parallel computing hardware (CUDA, etc.) and the means to properly program them, these techniques benefit significantly.

4.1.6. Multi Model Approach

The Multi Model Approach or MMS is one of the simpler concepts in theory, but its complexity scales quickly with increased number of possible faults. Essentially, this approach pre-defines a set of considered faults and triggers, which each scenario making use of a specially adjusted system model and controller. One can compare this to a gain scheduling approach, as used in other problems, and it can be used to good effect. A standardised approach to designing such systems for overactuated drones have been proposed by Ahmed and Katupitiya [26], who successfully applied it to a 12 DOF quadcopter and propose a standardised design approach.

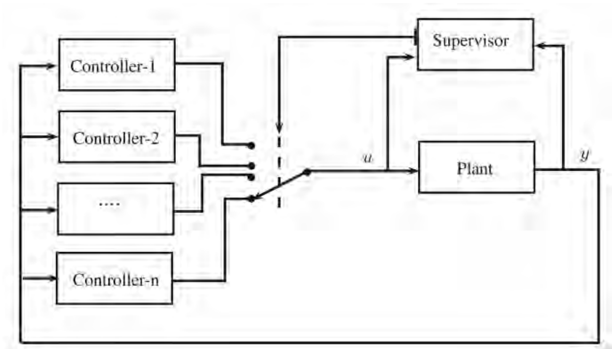


Figure 4.4: A schematic of an MMS based controller [15]

This method is especially sensitive to inconsistencies in the supervisor. This kind of problem is addressed by Allerhand et al. [27], who discuss a method to guarantee system stability. In the worst case, a fault is wrongly identified and the system crashes due to a wrong control architecture being applied to a normally functioning system. The biggest and most common issue however is if the detector is to sensitive and often forces the controller to switch between different control strategies leading to a jittery control system, and worst case also causing a crash due to the different controller constantly fighting each other to correct the system.

4.2. FTC and Overactuated Drones

An effort was made to find literature related to the fundamentals of fault tolerant control, but also their applications to drones and HVTOLs of different configurations. Simple quadrotors have previously been modified to make use of dual tilt actuators, and builds an additional basis for research and developments.

In the light of quadrotors, one of the major problems of the platform is the fact that the system is inherently underactuated. Control about all six degrees of freedom is not possible independently and only realised by rotor layout and control mixing [28]. In case one of the 4 rotors fails, the vehicle cannot remain in a fixed position and orientation. Fault tolerant controllers instead allow for the drone to spin about the yaw axis, which is considered to be the least important control axis and make use of the remaining control authority to balance the drone and allow for trajectory following. This relaxed hover state has been the foundation of many studies for rotor failures on multicopters [29]. Studies have been performed on using as few as two rotors for a commercial drone [30] and one rotor for a specially designed vehicle [31] both of which still retain control.

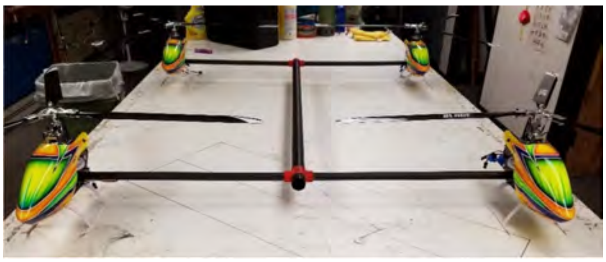
There has been a fair amount of investigation performed on different approaches to retaining degraded control of a quadrotor under failure, but few have investigated the fault tolerance of single or dual axis tilt capable drones. Their overactuated nature theoretically allows them to successfully mitigate failures without losing any angular control (compared to the yaw that is introduced in conventional quadcopters). Given the nature of these systems however, the non-linear nature and use of control allocation and optimizer introduces their own level of complexity and besides suggested design approaches to MMS [32] no standardised approaches exist. Optimisation based controller however also offer a certain advantage in that the control can be implicitly optimised for, accounting for constraints.

Some of the previously considered configurations can be seen in Figure 4.5

(a) Fully actuated Quadrotor [33]

(b) Quadrotor HVTOL independent single axis tilt [3]

(c) 2nd fully actuated Quadrotor [8]



(d) Quadrotor with collective and cyclic conntrol [34]

Figure 4.5: Previous research configurations of thrust vectoring Quadrotors.

Of the configurations presented above only the AirLab's HVTOL in Figure 4.5b has been investigated for FTC in the event of actuator failure [3, 4]. The vehicle makes use of an active FTC approach with on-board real-time FDI, which is performed by an ARX model continuously comparing predicted to actual system states and alarming upon violation of an accuracy threshold. The ARX internal transfer functions allow for identification of the failed actuator to be communicated to the control allocator, which accommodates the change by adjusting the control effectiveness of the failed actuator to zero. They have successfully demonstrated the viability of a feasible control space allocation and optimisation method, making use of control wrenches, contained trimmed states and allocation cost optimisation.

The investigation of fault tolerance for fully actuated vehicles with thrust vectoring thus far has not reached the depth of research performed for fixed rotor configurations. Also, fixed rotors can provide a fully actuated configuration by making used of several motors fixed at different angles around the frame, but lack the hybrid aspect of using the same motors for efficient forward flight after transformation.

4.3. Nemati Quadcopter

In this research project, the successful detection of faults is assumed to be provided by a perfect FDI mechanism and the focus lies on manually designing a control law for a single motor loss scenario for the single axis tilt quadcopter. In this case the tilting happens about the struts connecting the motors to the central frame as can be seen from Figure 4.6. As far as the literature reviewer is aware, this is one of the first papers specifically exploring the possibilities for fault tolerance offered by tilt-actuated multicopters. The paper constrains itself to investigate a predefined failure condition that allows for the explicit formulation of a new control law after failure.

4.3.1. Hand Designed Control Law

In the paper, the failure of rotor number 2 is the considered failure scenario. Instead of making use of any kind of control allocation or optimisation, Nemati et al. choose to apply a hand-tailored approach. In this, the tilt of motors No. 1 and 3 is fixed to zero, controlling the rotation about the vehicle x-axis (roll) and No. 4 and the respective tilt are used to control yaw and pitch. The initial set of equations can be seen in Equation 4.1 and Equation 4.2 where the disappearing terms are marked in red, the Forces generated by the rotors are denoted as F_i, the Moments from the motors as M_i, the angles of the motors as Θ_i, the rotational speed of the motors ω_i, dashed variables denote the failure variant of that variable, sin and cos are denoted as s and c. The C_i denote rotational drag coefficients. Post failure, the solution to the equilibrium is derived such that the torque of the number 4 engine also cancels its own moment generation about the pitch axis.

$$
\begin{aligned}
m\ddot{x} &= F_1 s\theta_1 c\psi c\theta - F_3 s\theta_3 c\psi c\theta - F_4 s\theta_4 c\psi s\theta s\phi \\
&\quad + F_4 s\theta_4 s\psi c\phi + 0 F_2 s\theta_2 c\psi s\theta s\phi - 0 F_2 s\theta_2 s\psi c\phi \\
&\quad + F_1 c\theta_1 c\psi s\theta c\phi + 0 F_2 c\theta_2 c\psi s\theta c\phi \\
&\quad + F_3 c\theta_3 c\psi s\theta c\phi + F_4 c\theta_4 c\psi s\theta c\phi + F_1 c\theta_1 s\psi s\phi \\
&\quad + 0 F_2 c\theta_2 s\psi s\phi + F_3 c\theta_3 s\psi s\phi + F_4 c\theta_4 s\psi s\phi - C_1 \dot{x} \\
m\ddot{y} &= F_1 s\theta_1 c\psi c\theta - F_3 s\theta_3 s\psi c\theta - F_4 s\theta_4 s\psi s\theta s\phi \\
&\quad + 0 F_2 s\theta_2 s\psi s\theta s\phi - F_4 s\theta_4 c\psi c\phi + 0 F_2 s\theta_2 c\psi c\phi \\
&\quad + F_1 c\theta_1 s\psi s\theta c\phi + 0 F_2 c\theta_2 s\psi s\theta c\phi \\
&\quad + F_3 c\theta_3 s\psi s\theta c\phi + F_4 c\theta_4 s\psi s\theta c\phi - F_1 c\theta_1 c\psi s\phi \\
&\quad - 0 F_2 c\theta_2 c\psi s\phi - F_3 c\theta_3 c\psi c\phi - F_4 c\theta_4 c\psi s\phi - C_2 \dot{y} \\
m\ddot{z} &= -F_1 s\theta_1 s\theta + F_3 s\theta_3 s\theta - F_4 s\theta_4 c\theta s\phi \\
&\quad + 0 F_2 s\theta_2 c\theta s\phi + F_1 c\theta_1 c\theta c\phi + 0 F_2 c\theta_2 c\theta c\phi \\
&\quad + F_3 c\theta_3 c\theta c\phi + F_4 c\theta_4 c\theta c\phi - mg - C_3 \dot{z}
\end{aligned}
\tag{4.1}
$$

$$\begin{aligned} I_x\ddot{\phi} &= l\left(F_3 c\theta_3 - F_1 c\theta_1 - C_1'\dot{\phi}\right) \\ &\quad + (M_1 s\theta_1 - M_3 s\theta_3) + (0M_2' + M_4') \\ I_y\ddot{\theta} &= l\left(F_4 c\theta_4 - 0F_2 c\theta_2 - C_2'\dot{\theta}\right) \\ &\quad + (M_4 s\theta_4 - 0M_2 s\theta_2) + (M_1' + M_3') \\ I_z\ddot{\psi} &= l\left(F_1 s\theta_1 + 0F_2 s\theta_2 + F_3 s\theta_3 + F_4 s\theta_4 - C_3'\dot{\psi}\right) \\ &\quad + (M_1 c\theta_1 - 0M_2 c\theta_2 + M_3 c\theta_3 - M_4 c\theta_4) \end{aligned} \tag{4.2}$$

Eliminating the effects of motor No. 2 from the equations yields a reduced set of equations. A new control law is derived to compute the new tilt angle of the 4th motor such that yaw and pitch remain in steady state. Additionally, to remain in a stable hover, the loss of thrust has to be compensated by adjusting the rotational velocity of the remaining engines. This has to be done whilst conforming with the previously setup equation for the tilt of motor No. 4. This new law can be expressed as in Equation 4.3. Here the respective rotational velocities of the rotors still need to follow the newly required thrust due to the loss of one engine.

$$\theta_4 = c^{-1}\left[\omega_4^2 / \left(\omega_1^2 c\theta_1 + \omega_3^2 c\theta_3\right)\right] \tag{4.3}$$

The resulting control laws are implemented via PD control to stabilise the vehicle after the rotor loss. Some loss of altitude was observed during the test flights, as well as an additional offset in the new yaw angle. The control after failure is realised by feeding the PD controllers new desired pitch and roll angles.

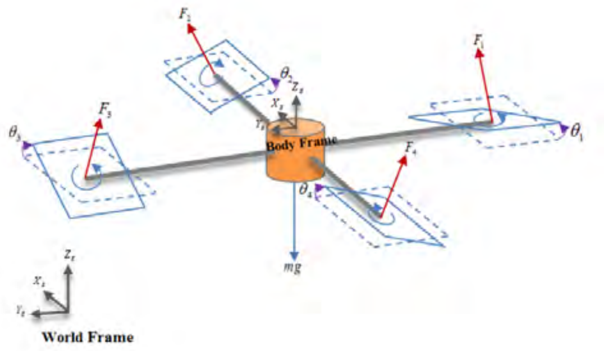


Figure 4.6: Actuator layout of the Nemati Quadcopter [2]

Application to the quadplane: *To summarize the paper, this approach is likely the simplest possible implementation of a fault tolerant controller for a sufficiently powerful vehicle. However, if a method like this would be applied to the dual tilt quadplane, a lot of potential for optimising the control scheme would be lost, as several actuators would have to be artificially locked to allow for explicit allocation of control, further showing that this method would be very inefficient at making the best use of available controls. In contrast to this older more fundamental paper, the approach taken by the Airlab in the next chapter bears more resemblance to the control allocation problem as faced by the MAVlab quadplane.*

5

FTC and HVTOL Drones

This section will look into the work that has been done specifically in the field of HVTOL drones, as opposed to conventional quadrotors. Whilst they share comparable performance, especially for the critical hover phase, their lifting bodies introduce a number of differences that have to be observed. In gusty hover and especially in failure, their lifting bodies may cause disturbances to the control system from aerodynamic forces generated from body motion or the environment. The separation is mostly done to get an idea of what research has been done for configurations similar to the 2-axis tilt vehicle to be considered for the thesis project.

Most research and industry solutions still makes use of dedicated vertical lift solutions, which reduces system and control complexity at the cost of unnecessary components for forward flight modes. The exception to this are various hybrid helicopters configurations, such as the X-50A like vehicle presented by Wang et al. [35]. These systems have been proven promising in the past and retain the effectiveness of the main rotor for other phases of flight. In the case of Wang et al. the main rotor employs elliptical rotor blades which can be locked as a third main wing in addition to two canards. In forward flight the propulsion is taken over by a separate puller motor. Despite research into their fault tolerance as presented by the sliding mode approach in [35], the system is not overactuated and the paper investigates more classical problems such as locked control surfaces instead of the failure of a engine.

The majority of current drones do however make use of a classical quadrotor layout with beam-like extensions from the lifting body and shut these lift engines off after transition to forward flight. Subsequently, this combination of single purpose (vertical and forward flight) systems lack the advantages of fully actuated thrust generation in that a motor failure in hover still necessitates yaw spin to retain altitude and pitch/roll attitude. For quadplane layouts this is complicated by the aerodynamic forces generated by the wing during rotation, which has been investigated by Wang and Sun in [36], who managed to stabilise and control the vehicle after complete loss of one rotor via a incremental adaptive sliding mode control approach. Additionally, the dedicated lift systems do not contribute to any other phase of flight and instead tend to degrade flight performance in regimes other than hover.

Other systems, such as the PteroDynamics Drones [37], do possess reconfiguration capability but have the same problems with actuator failures, especially in hover configuration. Unless the number and position of motors is scaled such that full controllability can be preserved under failure and a proper controller is implemented, these systems are subject to the same failure modes and effects as classical quad or multicopters during hover, compounded by the aerodynamic interference of the wing.

Many current commercial drones and eVTOLs do not incorporate such tilt-mechanisms, but there is projects from many manufacturers that do consider hybrid setups. Lilium is working on a tiltwing like system that utilises a multitude of ducted electric fans [38], Airbus and their Vahana feature a more classical tiltrotor system and are experimenting with fixed rotor configurations that still allow for efficient foward flight with their CityAirbus NextGen. In either case, their drone-like electric propulsion makes it much easier to implement motor arrangements that favour fault tolerance, as no complex gearboxes and transmissions need to be designed as is the case for more conventional combustion based systems [39]. These more distributed propulsion concepts can make easier use of FTC and have increased safety margins based on the fact that a single motor failure degrades systems performance much less than is the case for the previously mentioned tilt rotors like the V-22.

Utilising a classical quadrotor layout and fully (2-axis tilting) actuated rotors, a high degree of fault tolerance is possible, which a standard layout cannot provide [40]. Additional benefits are the proper angling of the lifting body irrespective of translation. This is desirable for vertical landings in gusty situations such as shipborn operations. The actuators can tilt the vehicle without causing linear motion, allowing for reorientation in space without changing the position. This grants these configurations to ability to mitigate aerodynamic forces from the wing that would otherwise effect normal operations and require quick tilting of the entire vehicle. Their increased capability in terms of over-actuation and resulting freedom of orientation enables even systems that have experienced rotor failure to remain in a steady orientation and thus not face the complex aerodynamic of a spinning lifting body. Vourtsis et al. [41] investigated this from a slightly different angle of morphing aerodynamic surfaces instead of fully steerable rotors, as presented by Mancinelli et al. [1].

Other studies have investigated landing on and adhering to angled surfaces and shown them to be feasible via real time trajectory planning [42], complex maneuvers to do these landings could be simplified by translation independent rotation as investigated in [8] to transverse narrow spaces. Having established some of the advantages of fully actuated tiltrotors over conventional quadrotors, some existing investigations into their fault tolerance will be reviewed.

5.1. Airlab Quadplane

The most comparable quadplane to the vehicle under development in the MAVLab is the quadplane developed by the Airlab. As visible from Figure 5.1 the motors can independently tilt around the connecting beams axis, generating forward or backwards components of thrust. In the following sections the FDI and FTC methods used will be presented in more detail.

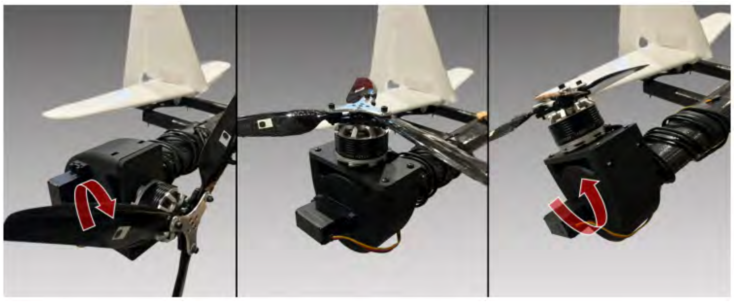


Figure 5.1: A closer look at the tiltmechanism of the individual rotors [3]

5.1.1. ARX Model for Failure Detection

An ARX model is a combination of an Autoregressive (AR) and an Exogenous (X) model which is used to analyse and represend dynamic systems. By their nature, they are defined as linear models, but for the application to the considered vehicle, many signal pairs can be estimated with a linear model. This means, that the relations between these actuator signals and outputs have to be designed such, that they can be approximately modeled by linear systems. Once the ARX model has reached a point where the coefficients successfully estimate the input output relations within a reasonable margin, large deviations from predicted behaviour can be used to detect the presence of faults.

In general an ARX model is defined as follows:

$$y(t) = c + a_1 y(t-1) + a_2 y(t-2) + ... + a_p y(t-p) + b_1 u_1(t) + b_2 u_2(t) + ... + b_q u_q(t)(+e(t)) \tag{5.1}$$

Or with indexing as a time series model with single in and output:

$$y_{k+1} = \sum_{i=1}^{n_a} a_i y_{k-i+1} + \sum_{i=1}^{n_a} a_i y_{k-i+1} \tag{5.2}$$

This model can successfully detect sudden changes in the model dynamics via large sudden changes in the output residuals, **but lacks the function to identify where the fault originated**. It also cannot detect sufficiently incipient failures, as the model simply adapts to smaller changes as the fault progresses without crossing the residual threshold. As stated in their paper, the identification of the failure for the testing of the fault tolerant controller is assumed to be known. This also ties back to a statement made in an older article by Zhang et al. [43], in which is mentioned that the research performed on Fault Detection and Diagnosis (FDD)/FDI schemes and FTC and re-configurable controllers is largely performed separate from each other, but the availability of perfect FDD/FDI schemes is oftentimes assumed by the latter.

5.1.2. Wrench Space Analysis

In order to get an idea of the achievable forces and moments to be generated by the quadplane, a wrench space analysis was performed. This also allows for a very intuitive visualisation of the available reachable forces and moments as can be seen in Figure 5.2

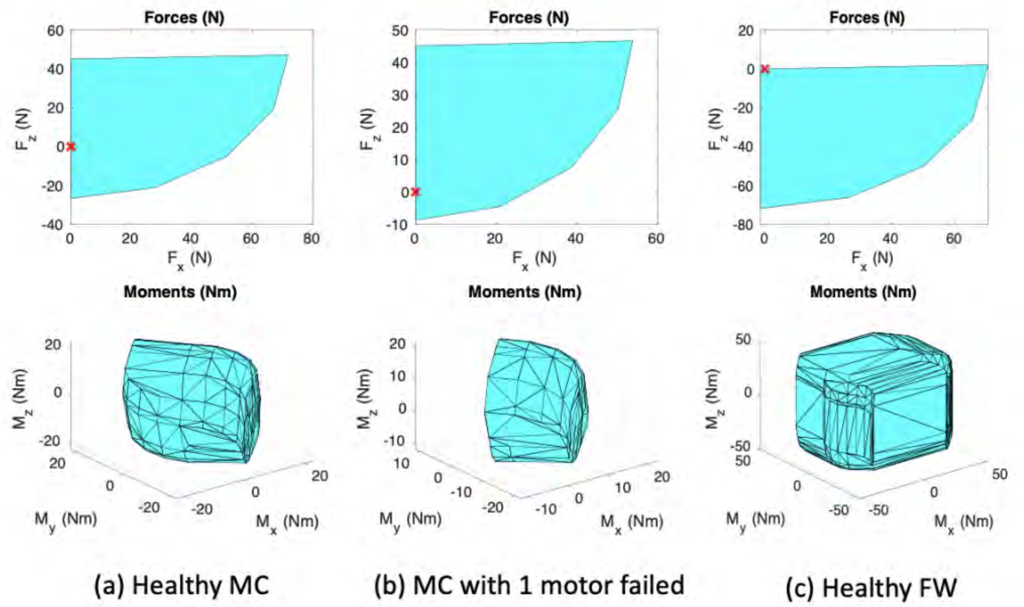


Figure 5.2: The feasible wrench sets as described in the respective sub-captions. It can be seen from b) that the failure of one rotor has significant impact on the feasible space.[3]

The wrench space is also used as a means to set up the control allocation problem, in that the controller provides a reference control wrench in the shape of three desired forces and three desired moments about the vehicle axes. The vehicle dynamics are linearised about the trim condition and applying small pertubation Theory:

$$F^B = F^B_{trim} + F^B_\Delta, \qquad M^B = M^B_{trim} + M^B_\Delta \tag{5.3}$$

In which a first order pertubation is used to approximate the pertubation:

$$F^B_\Delta = \frac{\partial f(u)}{\partial u}|_{x_0,u_0}\Delta u, \qquad M^B_\Delta = \frac{\partial g(u)}{\partial u}|_{x_0,u_0}\Delta u \tag{5.4}$$

This is already in a form that can be optimised for and that is done by first computing the least norm solution by utilising the pseudo inverse of the A matrix. Subsequently actuator limits are accounted for by optimising a constrained optimisation problem that computes additional coefficients to maintain non-saturated actuators. This method can be fairly easily adjusted to work in failure cases, as the respective column in A can simply be set to zero with the same optimisation method applied.

The control method has proven to successfully stabilise the system after informed motor failure in hover with a relatively short transition time of 8 seconds in which the controller converges to the optimal solution. The vehicle does experience some rotation about yaw in this transition period, but overall remains under control.

6 Research Planning and Revised Questions

The main research question was already predefined, in that the goal of the Thesis is to develop and implement a fault tolerant controller on the existing prototype. In more detail, the specifics of how many types of failure can be mitigated etc. were not specified directly, but a higher performance controller is the goal.

To reach that goal, several sub-steps have to be taken. First it has to be identified which types of failure detection and mitigation can successfully be implemented on the existing hardware. Constraints exist in the shape of the power of onboard hardware, and the way the current control system is set up. As the project progresses further, certain areas may arise that show the most potential for application. It was mentioned in Chapter 4, that the prime failure case to be considered for the duration of the thesis is the complete failure of one of the drones rotors. In that sense, FDI will also not take priority, but shall be thought about to improve the controllers performance and response to faults.

Secondly, the goal that has to be reached with the new control architecture is to successfully have the aircraft fly and survive (controllable) this rotor failure and other potential failure scenarios. These scenarios would first be run in simulation and later on the real prototype to avoid unnecessarily damaging the prototypes.

6.1. Identifying Gaps

In order to specify the research questions and positively contribute to the research, it should first be identified where there are gaps in the current state of research. As they are identified, these gaps open up the opportunities to apply the existing theory to the vehicle and improve its function, whilst creating research value for the field of overactuated quadplanes.

In the the comparative report from Al-Ali et. al [40] it was noted, that most studies up until 2019 only covered single tilt-quadrotors with no research yet performed beyond the work of Nemati [2]. This research deficit is compounded by a lack of investigating fault impact on system dynamics, simplified rotor models, a lack of variety and performance comparison of different control techniques and the gap between FDI and FTC.

To the knowledge of the author, the only paper that has specifically dealt with single axis tilt reconfigurable hybrid drones is the AirLab of Carnegie Mellon University. Their approach to analysing and mapping the remaining available control authority of a vehicle after actuator failure may be used to assist in finding viable control strategies for the MavLab vehicle. Their papers introduce an ARX model based residual FDI scheme and an optimisation based controller with constraints based on wrench space analysis [3, 4]. A similar mapping of the available control space may be valuable for the vehicle under consideration. As has been noted in Section 5.1, their approach still lacks the functional combination of an FDI and FTC scheme, instead assuming informed failure for the considered test cases. As has been presented in Section 4.1.2, quadrotors make use of the parameter based parity-space method, whose potential for application to quadplanes may be investigated in this project, if time allows. In case a parity-space based observer can be successfully implemented, the problem remains of detecting faults in the servos. Their seizure or malfunction may not be easily mapped to the parity-space model, so further investigations have to be performed into what can be done to detect and identify such fault scenarios. A possible way to achieve this may be the investigation of acceleration residuals in addition to the rotational behaviour. At a glance this does not seem able to yield distinct information on which actuator is experiencing faults, but it should be further investigated whether identification is possible via vehicle states and residuals, or whether additional means are required.

However, the thesis shall focus on investigating the existing controller for the potential of fault tolerant control and search for methods that automate and implement informed failure fault tolerance. After some discussion it was decided to not prioritise a MMS approach due to the high complexity of switch dynamics, and instead favour the investigation of different control approaches to retain control under failure. The existing controller and optimiser framework give a good baseline to test the current systems resilience and adaptability to failure, but different control approaches shall be considered for failure.

Thus the main point is to focus on the development and implementation of the actual fault tolerant controllers and methods to reconfigure after failure, instead of identifying them. This on its own holds enough potential for a thesis project and shall thus be the first priority. Potential different approaches include the reorientation of the vehicle in space, which is currently still trying to maintain a certain horizontal reference state and introduce different means of retaining partial actuator authority after failure. For example if a servo fails on one of the motors, it may be possible to start spinning the vehicle slowly about one of the body axis to prevent the complete loss of available thrust from that motor. Reorientation of the vehicle may reduce the control effort and make more efficient use of the remaining available thrust. Methods to identify such orientations and a means to control the transition to and movement under these new trim states is to be the core of the research. The possibilities that are open here are rather vast and will thus be explored in as wide and detailed a scope as possible, with the reservation that FDI may be added should time be sufficient.

The failures that are to be considered should be limited to certain failure scenarios for the purpose of testing and evaluating control performance. One of the most important ones would be the failure of a motor. This scenario is to be considered both in simulation and flight tests, as it effectively eliminates 3 control inputs from the system (the azimuth and elevation control on that motor become ineffective). Besides that, there are the possibilities of stuck or degraded actuators and as has happened during a test-flight, the detachment of an engine. Especially for those scenarios it would be very valuable to investigate potential schemes for FDI in the future as the motor feedback was still reporting a functional actuator.

In the light that the research project shall yield new and own work to contribute to research and the active project, the existing controller should be seen as more of a baseline. It can also be modified, but shall not simply be adapted to include a change to the existing optimisation weights. Instead, the research project shall take a more top-down approach to finding solutions to the stated problem and research questions.

The revised research questions are then defined as follows:

- How can the 2-axis tilt quadplane implement fault tolerant control?
  - Assuming informed failure, what can be done to maximise the remaining performance?
  - What are possible ways to identify a new vehicle orientation that makes more efficient use of the remaining thrust in hover?
  - What kind of controllers can be used for this, and how well do the perform in the face of the non-affine system?
  - Reserve: Are there better FDI approaches available, that can be adapted to the vehicle?
- What is the performance of the selected method(s)?
  - How do they compare to each other, and the simulation results vs. flight test?
  - What are the results in terms of computation times, system complexity and tracking performance?
  - Can upset recovery be realised? Is it possible to first establish a stable hover and then re-orientate for optimal performance?
  - Reserve: How does this additional control mode compare to simply adapting the existing optimiser?

It can be seen that the questions have been changed significantly to both narrow down the scope and encourage a more open approach to the control methods under failure and their implementation. The second and third question have been omitted as they were poorly phrased and touched more upon the planning of the coming research work, rather than the questions that shall be answered by it. The questions marked with ”Reserve” are related to the implementation of an additional method for FDI, but as with previous research, the project will first assume perfectly informed fault scenarios, potentially coming back to this if the scope of the project allows for it.

6.2. Research Planning

As previously stated, the thesis project already has the more accurately defined goal to develop and integrate a fault tolerant controller on the tiltrotor prototype. To that extent, a somewhat coarse Gantt chart has been created that aims to give a timeline to the Literature Study and Thesis work. It can be found in the Appendix A. The large work packages are the writing and presentation of the literature review, familiarisation with the Simulink model and in-software testing of controller architectures. Once it can be said with adequate certainty that the controller(s) performs desirably, the control architecture can be compiled and tested on the hardware prototype under safe conditions. Partially parallel to this are the efforts of collecting, organising and evaluating the data from the simulations and flight tests. From this data, comparisons and conclusions are to be drawn with potential adjustments to the controller implementations. The extent of the flight experiments will depend on the successful implementation first, but (upset) recovery from failure and tracking of a simple hover trajectory shall be considered. The work during this time shall be well documented, such that the progress and intermediate results can be presented well in the final report.

Lastly, with all the data generated the report is to be finalised with the gained results and insights to answer the research question and aims to close the identified gap in existing theory and applications. A thematic overview of the different work areas is given in Figure 6.1, with a Gantt chart to be found in Appendix A. A rough timeline for the different action items within the Thesis work has been created. The writing of the reports for the literature study and the actual thesis will, as previously mentioned, both run alongside the research work as an ongoing process. The testing, first in simulation, is represented in the scheduling, risk mitigation for the prototype shall be maximised. As data from simulation and test flights is collected, performance can be analysed and adjustments made if necessary.

As for managing the code and data during the project: Sadly git management of Simulink files is flawed, and does not allow for proper practice. There is a central repository with the most up to date version of the code, that access is provided to. Personal developments will be backed up on a personal mobile hard drive and home PC with an additional mirror to a cloud to provide additional safety in case of hardware loss or failure. The data will be backed up after major changes and in case of the cloud backup may also be encrypted before upload, such that sensitive data is not unintentionally leaked. The collected data and related control architectures can then be evaluated and compared in the report as well as detailing the background and development work. This will then lead to a conclusion to the research project in which the research objective will be recapped and the formulated research questions answered. An outlook and recommendation on future work shall be included.

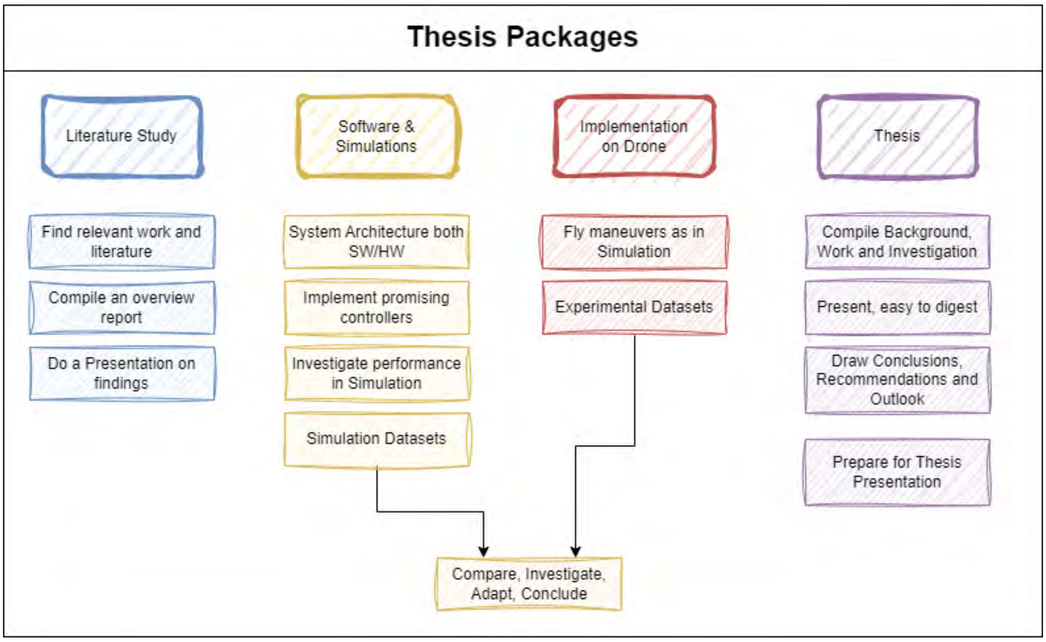


Figure 6.1: Overview of work packages

Part III

Additional Results

7

Other Results

This section serves to highlight some of the additional work that was done during the project but has not made it into the final article. It deals with the problems faced during the development of the controllers and elaborates on the analysis of the logs after the respective test flights. Some issues were seen easier than others and the inclusion of some of this development content serves to provide a more complete picture of the process. The final section deals with a short review of different visualisation methods.

7.1. Before Flight Tests

The proposal for this thesis came to be because the controller in its original configuration does not support Fault Tolerant Control. In case a failure is simulated on one of the motors, the optimiser would still attempt to allocate to that that motor which causes the system to eventually crash as the behaviour of the control allocation remains unchanged. Shutting down the considered engine reduces its control effectiveness and that of the connected servos to near zero, which needs to be implemented into the controller design. The first tests attempted to force the shutdown of the engine by significantly increasing the failed motors cost function weight. This approach has have several shortcomings: Simply increasing the weight did not guarantee a certain actuation range, and using a weight that was high enough to accomplish minimal use did have a negative effect on the evaluation of the cost function as a whole. Instead it was decided to limit the range of actuation for actuators in which a fault has been detected (Table 7.1). Upon failure the optimisation constraints are adjusted such that their usage is forced to a set value.

Table 7.1: Qualitative comparison of forcing methods.

Type of Forcing	**Advantage**	**Disadvantage**
Weight	*Comparatively simple implementation, straightforward cost change*	*Does not guarantee command values to lie within capability of failed actuator. Disturbs cost function with high weight influence*
Range constraint	*Direct forcing of possible actuator values after failure occurs, no disturbance of cost function gradients*	*Requires more detailed changes to optimisation constraints based on case*

In order to trigger the failure an additional channel to the non-linear controller had to be created. This channel would be fed a switch value that either caused the change of the failed actuator weight or different optimisation constraints for the failed actuator. This enabled time based triggering of the failure in simulation. For the test flights, the switch was mapped to one of the RC controller switches.

Note on Model Predictive Control

In the early stages of the project a CasADi[1] based implementation was considered, however the increased computation times caused this to run slow to the point of in-feasibility on the test computer. Model Predictive Path Integral (MPPI) control with parallel sampling of trajectories and subsequent evaluation can be run much faster on e.g. CUDA accelerated hardware, but that may be left open for a future project, especially considering the already limited thrust to weight ratio.

[1] https://web.casadi.org/

7.2. Flight Tests - 1

As previously described in the paper, the considered scenario focused on motor 3 failure during regular hover. In the first set of test-flights, the drone did not succeed in the recovery task. Once the fault on motor 3 was introduced, the remaining motors all reduced speed to around 550 rad/s, which was too low to generate sufficient thrust for flight. The logs were copied and compiled and analysis started on the gathered flight data. First the focus was to compare the predicted result with the logged data and identify where the problem originates. During the data review, the motor anomaly that can be seen in Figure 7.1 was specifically investigated. It could be seen that the failed motor never reached the speed the controller asked of it, instead being at a constant offset from it. In simulation this did not show at the time, as the minimum speed limit was set to 0. The reason behind this was the nature of the on-board control framework: The motor speed threshold was not set correctly and violated the minimum framework-required 150 rad/s, instead the restricted value for the optimiser was defined to lie at 0. This caused the controller to seize up on the real vehicle when the supposedly feasible solution could not be reached.

Setting the respective settings in Simulink, it was possible to re-create the motor response in simulation and verify that the issue was caused by this unwanted gap.To fix the issue the post-failure motor speed has been adjusted such that it conforms to the framework limitations. This has in further flights proven to have solved this particular issue.

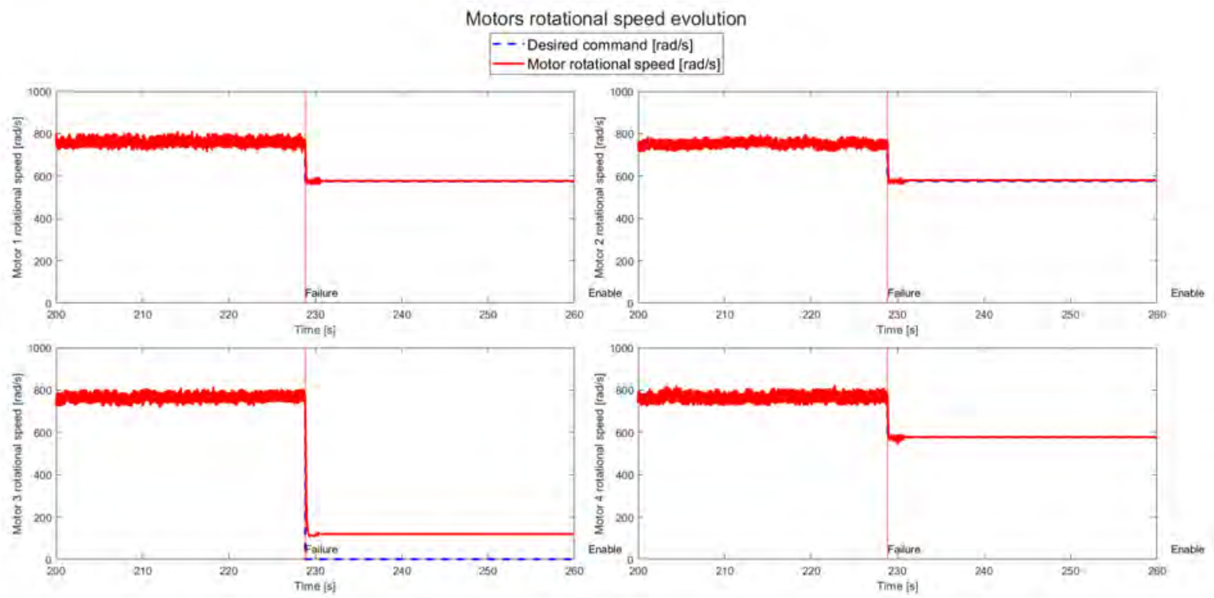


Figure 7.1: Motor speed limit issue identified in the first set of testflights.

7.3. Flight Tests - 2

The second flight test brought partial success. In this flight, the drone was first oriented into an attitude close to the one expected from Simulink predictions before the fault was triggered. By doing this, the transition phase that was the biggest problem throughout the project was bypassed and the controller only flew the vehicle from an already stable initial position. Even though shutting off motor 3 (back right) changes the thrust configuration, the controller stabilised the vehicle and small maneuvers were possible. After removing the failure switch from the controller, it was possible to recover the reference attitude and perform a controlled landing. The controls after introduced motor failure and the subsequent recovery in the actuator commands can be seen in Figure 7.2. In Figure 7.3 the tracking performance of the flight can be seen, showing that the drone successfully remained close to the hover point while accurately matching the desired attitude in pitch and roll. Visible in the yaw angle is a deviation, which can also be seen in other test flights. For the course of the project we considered a yaw rate commanded to 0, this did however not implement a yaw angle control to restore original heading.

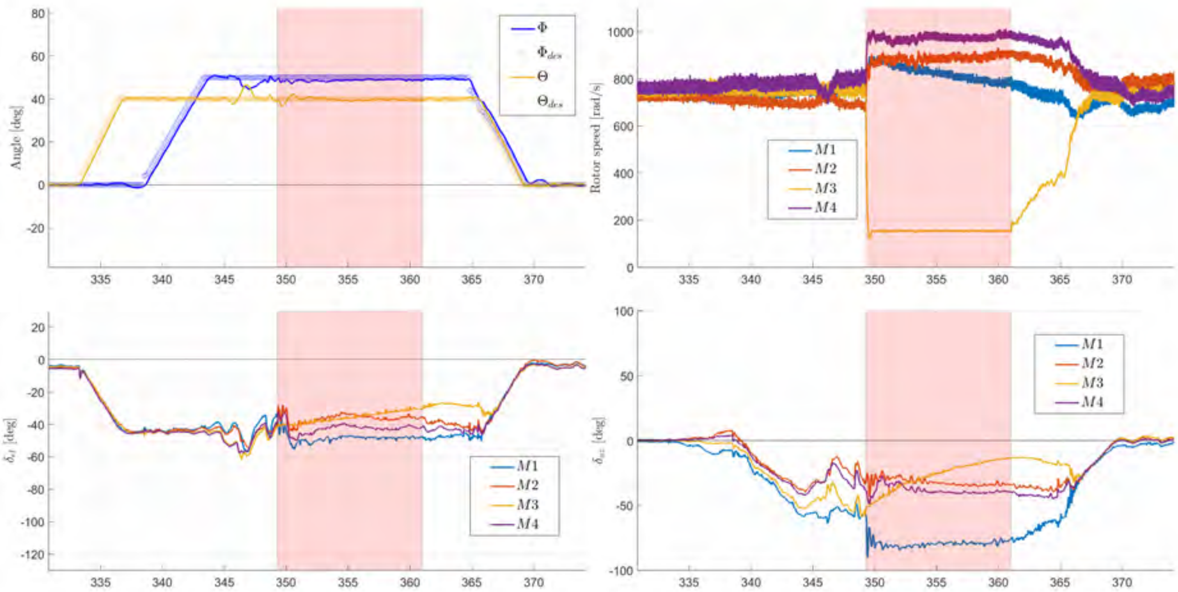


Figure 7.2: Combined plot of actuator states and body angles during prepared attitude failure scenario.

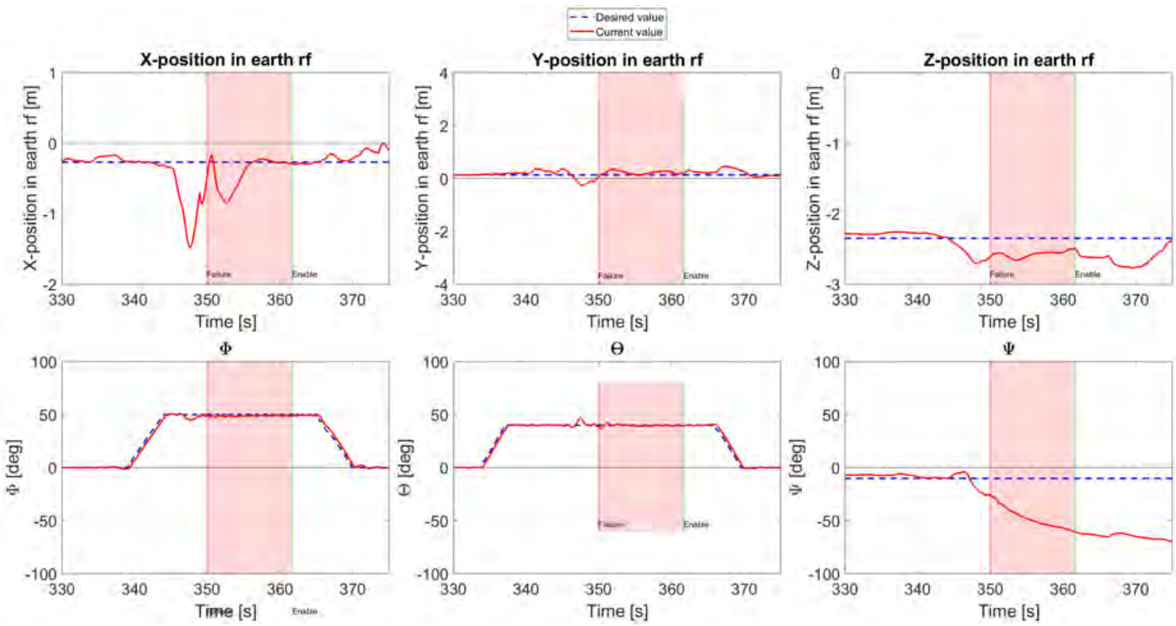


Figure 7.3: Ground track and body angles in prepared failure. Pilot position correction visible before failure introduction.

The commanded attitude was fed via the hover reference attitude channel. That is, the optimiser penalises actuator configurations that stray away from the specified reference. This makes the drone follow said reference closely, given sufficiently high weights. As the orientation was already proven feasible in simulation, it did not deviate far from it after motor failure. Beyond this, the second set of test flights did not yet yield a full transition success. The problem here was primarily the noise in the actuator commands and commanded attitude. After this flight it was decided to look at different methods to generate the angular references separately to the actuator solution to solve this problem of simultaneous optimisation. This approach ended up yielding the cascaded approach as presented in the paper above. A different solution was considered in which a one-shot optimiser would establish a new desired orientation once, which is then used by the original controller. This idea was dropped in favor of the cascade, as it would still not consider the dynamics during the transition and base the desired attitude on a snapshot of time only.

7.4. Flight Test - 3

Stepping over to tests performed with the full cascaded implementation, the concept of motor orientation forcing by only motor cost proved to be a complication. While a high motor cost is needed to generate the earth-normal orientation of the rotors in regular flight, during failure this would lead to problems in the first optimiser. It was observed that oscillations in commanded attitude were generated as the motor speed was constantly fighting to stay low. Inspecting Figure 7.4 it becomes clear that there is a connection in the behaviour of motor 1 speed and the computed optimal attitude angle.

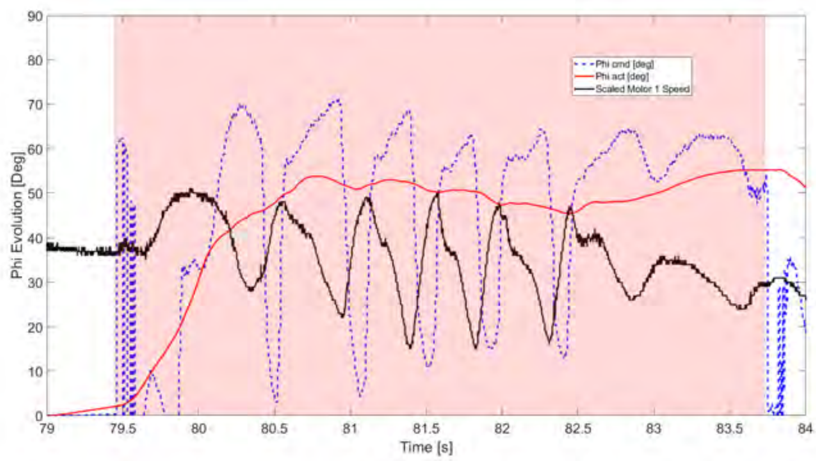


Figure 7.4: Oscillation between Motor Speed (black) and Roll angle (blue: desired, red: achieved) due to high motor cost.

While there was no large effect observed on the attitude response of the vehicle, the RPM oscillation should be avoided. The discrepancy between desired and actual body attitude should be minimised to avoid problems as presented in the paper, where the oscillation originally propagated into the angular acceleration reference. Even during the observed scenario of high motor cost under failure, the oscillation leaked into the servo commands via the generated acceleration reference.

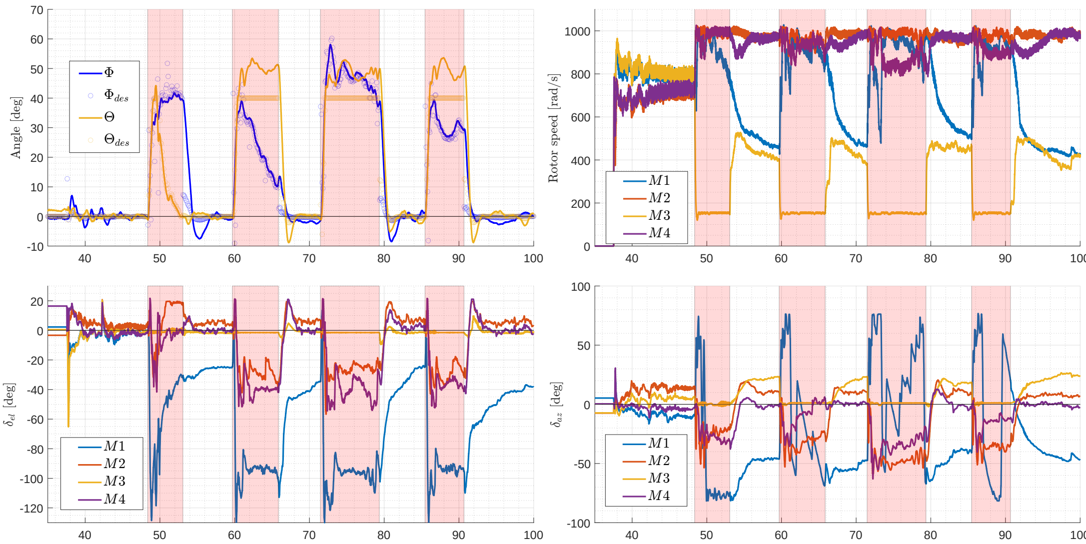


Figure 7.5: Using a lower cost without switch resulted in insufficient earth-normal reorientation during non-fault hover.

As previous tests did not show this behaviour, subsequent testing was performed in which the motor weight was lowered during the failure. During these scenarios, the oscillation was not present anymore proving the original suspicion correct. However, it was also noted during the flight that the lower fixed weight did not yield adequate rotor orientation. In non-fault condition the lower weight caused significant deviation from earth-normal alignment in regular hover, as can be seen in Figure 7.5. This also led to different configurations in recovery due to the varying initial conditions and insufficient forcing of gimbal orientation. In the end it was decided to lean on the already existing method of switching the constraints in the fault condition to switch the motor weight from high to low upon failure introduction and vice versa.

With weights adequate for both scenarios, the drone successfully recovered from repeated failure and performed the tracking task presented in the paper. As a visual support in addition to the videos linked in the paper, Figure 7.6 shows the recovered orientations.

Figure 7.6: The attitudes of the vehicle during repeated failures in final test flight series, as recorded by the test camera.

7.5. Visualisations

As can be seen from Figure 7.6 the resolution of the camera lacked detail to perform good visual inspection of the actuator behaviour. Different methods of visualising the system commands and response have been explored during the project. Matlab/Simulink supports a Unity Engine based visualisation of results, and can export video of a custom model. For the longest time the Unity view-port did however make use of a generic fixed-wing model to show the results. As it was desired to observe especially the orientation of the motors with respect to the body, this was not sufficient. The model has since been updated and now used the vehicle 3d model, indicating individual thrust magnitude and orientation via scaled 3d vectors.

In its place, a Matlab based visualisation was first implemented indicating the body angles, motor orientation and rpm and track: Figure 7.7 and Figure 7.8. These early models were later replaced with the Blender version below, or the direct Simulink-Unity visualisation.

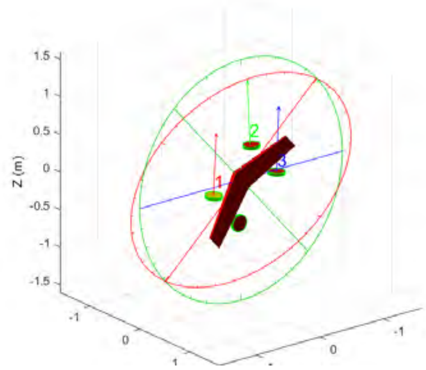


Figure 7.7: Early Matlab visualisation.

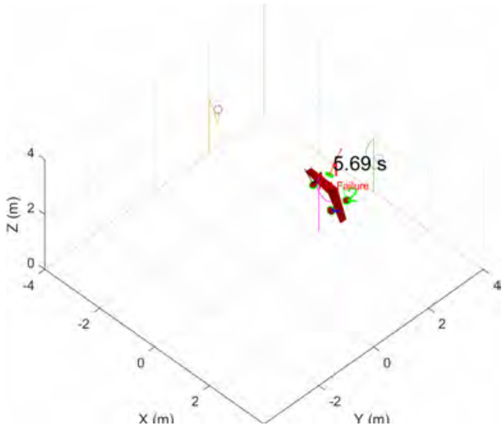


Figure 7.8: Still image of early animation.

For the flight test results, a custom python script was used to extract the data from the logs. This script automatically maps the RPM, attitude, servo commands and vehicle position to a 3d model ready for animation. Compared to the Mathworks implementation, the user can also simply "drag" the timescale to see the behaviour in time as desired without performing the full video generation. The animated videos in the linked playlist of the article were generated largely via the EEVEE render engine and a virtual rig of cameras[2].

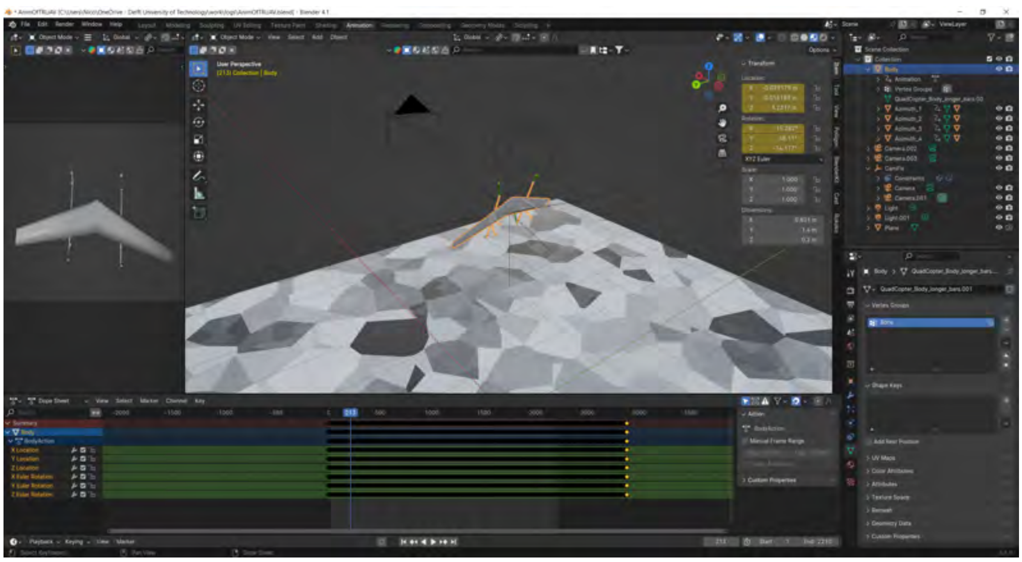


Figure 7.9: A screenshot of the Blender environment used for visualizing the flight test results.

[2]https://www.youtube.com/playlist?list=PLleZWIsVvmSBqVSoune02p6FpacddIxAU

References

[1] Alessandro Mancinelli et al. “Dual-axis tilting rotor quad-plane design, simulation, flight and performance comparison with a conventional quad-plane design”. In: *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2022, pp. 197–206. DOI: 10.1109/ICUAS54217.2022.9836063.

[2] Alireza Nemati et al. “Stabilizing and Control of Tilting-Rotor Quadcopter in Case of a Propeller Failure”. In: *ASME* (2016). DOI: 10.1115/dscc2016-9897.

[3] Mohammadreza Mousaei et al. “Design, Modeling and Control for a Tilt-rotor VTOL UAV in the Presence of Actuator Failure”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 4310–4317. DOI: 10.1109/IROS47612.2022.9981806.

[4] M. Mousaei et al. “VTOL Failure Detection and Recovery by Utilizing Redundancy”. In: *arXiv.org* (2022). DOI: 10.48550/arxiv.2206.00588.

[5] Xiaomin Zhang et al. “Fault Detection and Identification Method for Quadcopter Based on Airframe Vibration Signals”. In: *Sensors* 21.2 (2021). DOI: 10.3390/s21020581. URL: https://www.mdpi.com/1424-8220/21/2/581.

[6] Seth B. Anderson. “Historical Overview of V/STOL Aircraft Technology”. In: *NASA Technical Memorandum* (1981).

[7] James Forrest et al. “Evaluating ship superstructure aerodynamics for maritime helicopter operations through CFD and flight simulation”. In: *The Aeronautical Journal* -1 (July 2016), pp. 1–26. DOI: 10.1017/aer.2016.76.

[8] Peter Zheng et al. “TiltDrone: A Fully-Actuated Tilting Quadrotor Platform”. In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 6845–6852. DOI: 10.1109/LRA.2020.3010460.

[9] R. Isermann et al. “Trends in the application of model-based fault detection and diagnosis of technical processes”. In: *Control Engineering Practice* 5.5 (1997), pp. 709–719. DOI: https://doi.org/10.1016/S0967-0661(97)00053-1. URL: https://www.sciencedirect.com/science/article/pii/S0967066197000531.

[10] Christopher Edwards et al. “Fault tolerant flight control”. In: *Lecture notes in control and information sciences* 399 (2010), pp. 1–560.

[11] Tobias Miksch et al. “Real-time performance comparison of fault-tolerant controllers”. In: *International Conference on Computability and Complexity in Analysis* (2008). DOI: 10.1109/cca.2008.4629695.

[12] Jovan Boskovic et al. “Failure Detection, Identification and Reconfiguration in Flight Control”. In: *STAR* (2003). DOI: 10.1007/3-540-45737-2_5.

[13] Zhen Yu et al. “Fault Identification and Design of Fault Tolerant Control for a Certain Aircraft”. In: *IMCEC* (2019). DOI: 10.1109/imcec46724.2019.8984018.

[14] Hussein Mazeh et al. “Offline and Online Active Fault Tolerant System For Multirotor Drones”. In: *International Conference on Robotics and Mechatronics* (2021). DOI: 10.1109/icrom54204.2021.9663499.

[15] Tushar Jain et al. “Active Fault-Tolerant Control Systems: A Behavioral System Theoretic Perspective”. In: *null* (2017). DOI: 10.1007/978-3-319-68829-9.

[16] Alireza Abbaspour et al. “A Survey on Active Fault-Tolerant Control Systems”. In: *Electronics* 9.9 (2020). DOI: 10.3390/electronics9091513.

[17] Behrooz Safarinejadian et al. "Fault detection in non-linear systems based on GP-EKF and GP-UKF algorithms". In: *Systems Science & Control Engineering* 2.1 (2014), pp. 610–620. DOI: 10.1080/21642583.2014.956843. URL: https://doi.org/10.1080/21642583.2014.956843.

[18] Hongtian Chen et al. "Data-Driven Designs of Fault Detection Systems via Neural Network-Aided Learning." In: *IEEE Transactions on Neural Networks* (2021). DOI: 10.1109/tnnls.2021.3071292.

[19] Jasper van Beers. "Development of a system identification routine, with prediction reliability estimates, suitable for modelling high-speed and aggressive quadrotor flight outdoors". In: *TUDelft* (2021). URL: http://resolver.tudelft.nl/uuid:4677f001-bc4f-4990-be0e-4316ec9f8847.

[20] Vidyasagar Sadhu et al. *On-board Deep-learning-based Unmanned Aerial Vehicle Fault Cause Detection and Identification*. 2020. eprint: 2005.00336.

[21] R.J. Patton et al. "A Review of Parity Space Approaches to Fault Diagnosis". In: *IFAC Proceedings Volumes* 24.6 (1991). IFAC/IMACS Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'91), Baden-Baden, Germany, 10-13 September 1991, pp. 65–81. DOI: https://doi.org/10.1016/S1474-6670(17)51124-6. URL: https://www.sciencedirect.com/science/article/pii/S1474667017511246.

[22] Sihao Sun et al. "Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 4273–4279. DOI: 10.1109/ICRA40945.2020.9197239.

[23] Nico Voß. "Helicopter Flight Systems". In: *Internship Report* (2022).

[24] Max Schwenzer et al. "Review on model predictive control: an engineering perspective". In: *The International Journal of Advanced Manufacturing Technology* (2021). DOI: 10.1007/s00170-021-07682-3.

[25] Nopporn Patcharaprakiti et al. "Model Predictive Control Based on System Identification of Photovoltaic Grid Connected Inverter". In: *International Journal of Information Engineering and Electronic Business* (2012).

[26] Ahmed et al. "Modeling and Control of a Novel Vectored-Thrust Quadcopter". In: *Journal of Guidance, Control, and Dynamics* 44 (Apr. 2021), pp. 1–11. DOI: 10.2514/1.G005467.

[27] L. I. Allerhand et al. "Robust Switching-Based Fault Tolerant Control". In: *IEEE Transactions on Automatic Control* 60.8 (2015), pp. 2272–2276. DOI: 10.1109/TAC.2014.2375752.

[28] Samir Bouabdallah et al. "Full control of a quadrotor". In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2007, pp. 153–158. DOI: 10.1109/IROS.2007.4399042.

[29] Mark Mueller et al. "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles". In: *The International Journal of Robotics Research* 35 (Oct. 2015). DOI: 10.1177/0278364915596233.

[30] 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 (2021), pp. 116–130. DOI: 10.1109/TRO.2020.3010626.

[31] Weixuan Zhang et al. "Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space". In: *Mechatronics* 61 (2019), pp. 117–130. DOI: 10.1016/j.mechatronics.2019.06.004.

[32] Qandeel Fazal et al. "Robust fault tolerant control of an unmanned aerial vehicle in the presence of actuator faults". In: *IEEE PLANS* (2016). DOI: 10.1109/plans.2016.7479770.

[33] Ali Bin Junaid et al. "Design and Implementation of a Dual-Axis Tilting Quadcopter". In: *Robotics* 7.4 (2018). DOI: 10.3390/robotics7040065. URL: https://www.mdpi.com/2218-6581/7/4/65.

[34] Colin Alexander Sledge. "Design, Build and Analysis of a Multirotor Vehicle with Cyclic-Collective Pitch Control". In: *UC Irvine Electronic Theses and Dissertations* (2021).

[35] Ban Wang et al. “Adaptive Fault-Tolerant Control of a Hybrid Canard Rotor/Wing UAV Under Transition Flight Subject to Actuator Faults and Model Uncertainties”. In: *IEEE Transactions on Aerospace and Electronic Systems* (2023). DOI: 10.1109/taes.2023.3243580.

[36] Xuerui Wang et al. “Incremental fault-tolerant control for a hybrid quad-plane UAV subjected to a complete rotor loss”. In: *Aerospace Science and Technology* 125 (2021), pp. 1–9. DOI: 10.1016/j.ast.2021.107105.

[37] *PteroDynamics Homepage*. https://pterodynamics.com/. Accessed: 2023-05-15.

[38] Patrick Nathen et al. “Architectural performance assessment of an electric vertical take-o□ and landing (e-VTOL) aircraft based on a ducted vectored thrust concept”. In: 2021.

[39] Bowen Zhang et al. “Overview of Propulsion Systems for Unmanned Aerial Vehicles”. In: *Energies* 15.2 (2022). DOI: 10.3390/en15020455. URL: https://www.mdpi.com/1996-1073/15/2/455.

[40] Ismail Al-Ali et al. “State of the art in tilt-quadrotors, modelling, control and fault recovery:” in: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* (2020). DOI: 10.1177/0954406219878741.

[41] Charalampos Vourtsis et al. “Wind Defiant Morphing Drones”. In: *Advanced Intelligent Systems* 5.3 (2023). DOI: https://doi.org/10.1002/aisy.202200297.

[42] Sensen Liu et al. “Hitchhiker: A Quadrotor Aggressively Perching on a Moving Inclined Surface Using Compliant Suction Cup Gripper”. In: *IEEE Transactions on Automation Science and Engineering* (2022). DOI: 10.48550/arxiv.2203.02304.

[43] Youmin Zhang et al. “Bibliographical review on reconfigurable fault-tolerant control systems”. In: *Annual Reviews in Control* 32.2 (2008), pp. 229–252. DOI: https://doi.org/10.1016/j.arcontrol.2008.03.008. URL: https://www.sciencedirect.com/science/article/pii/S1367578808000345.