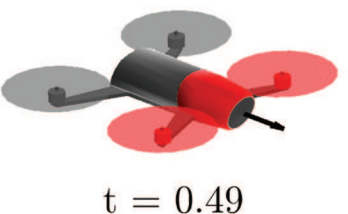
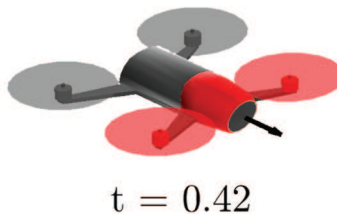
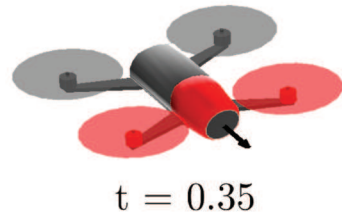
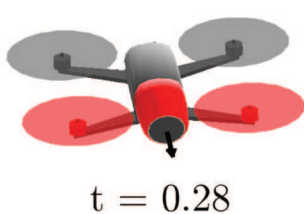
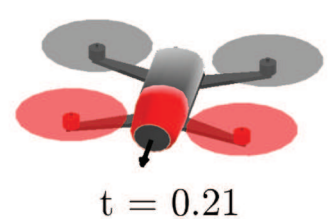
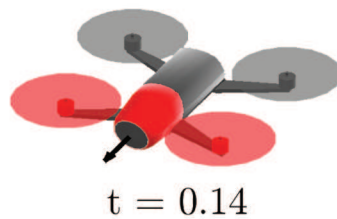
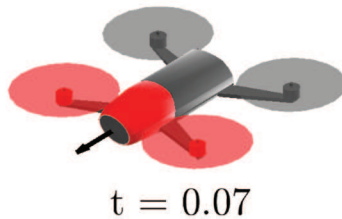
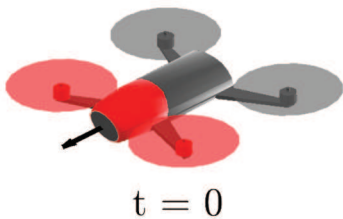


Actuator Saturation Handling using Weighted Optimal Control Allocation Applied to an INDI Controlled Quadcopter

Master of Science Thesis

D.C. Höppener



Actuator Saturation Handling using Weighted Optimal Control Allocation Applied to an INDI Controlled Quadcopter

Master of Science Thesis

by

D.C. Höppener

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday October 27th, 2016 at 14:00

Student number:	40402387
Project duration:	September 1, 2015 – October 27, 2016
Thesis committee:	Prof. dr. ir. M. Mulder, TU Delft, Committee Chair
	Ir. C. de Wagter, TU Delft, supervisor
	Ir. E.J.J. Smeur, TU Delft, supervisor
	Dr. Q. Chu, TU Delft
	Dr. O.A. Sharpans'kykh TU Delft, External committee member

This thesis is confidential and cannot be made public until defended.

Abstract

Incremental Nonlinear Dynamic Inversion provides a high performance attitude controller for multi-rotor Micro Aerial Vehicles by providing very good disturbance rejection capabilities. Flights conducted with a quadcopter revealed undesired pitch and rolling motions which occurred simultaneously with actuator saturation for instantaneous yaw angle reference tracking commands. Constrained control allocation methods can increase the system's performance by providing an effective strategy to prioritize control objectives, and redistribute control effort accordingly. Weighted Least Squares control allocation makes the constrained control allocation problem a quadratic optimization problem. An iterative solver based on the computationally efficient active-set algorithm finds the optimal control distribution for a weighted control objective. In this paper the Weighted Least Squares control allocator is used to overcome two challenges 1) increase performance by applying prioritization between control objectives and redistribute control effort accordingly, accounting for the actuator limits 2) enable flight when flying with severely compromised actuator(s). Real-world flight experiments are performed and show a significant increase in performance for high load yaw maneuvers, and enabled a quadcopter to perform controlled flight with a severely compromised actuator.

Preface

Personally I am very interested in inner loop attitude control. The first time this topic sparked my curiosity was when I saw a video of a Saab Gripen jet aircraft which was landing. All the control surfaces were moving very rapidly, and I wondered what algorithms were running to keep this aircraft stable. I really liked the idea of computers, intelligent algorithms, and controllers enabling high performance flight. When the time came to choose a graduation research topic, this was the type of topic I was searching for. Not only that, but I wanted to actually see physical results of my work. Luckily, at the TU Delft, Faculty of Aerospace Engineering, Control and Simulation department we have the MAVlab which practices research with quadcopters and other Micro Aerial Vehicles.

The topic of this research originated from a need: The superb INDI attitude controller is very good at rejecting disturbances. Naturally, to have a high performance controller, also aggressive maneuvering is desired. When this was done with an INDI controlled quadcopter for rapid yaw maneuvers, undesired coupled effects were observed. This generated the need to find a structural, root cause, solution to tackle this problem, which spawned this research.

Other than academic challenges associated with doing high quality research, for this research also practical challenges were present. Working at the MAVlab you need to learn to code in *C* and become more familiar with software development and the Paparazzi open source framework. I had little experience in both these fields, and this graduation enabled me to learn much about it. Admittedly, sometimes my mood was ruined because I would spend days fixing a small piece of code. But as time progressed and experience was gained, I became much faster and better in working with Paparazzi and *C*, which makes it a much more fun and rewarding task.

But I could not have done that all by myself and I would like to acknowledge Erik van der Horst and Anton Naruta specifically. Erik helped me a lot with Paparazzi and practical issues. I am really thankful for that. Anton learned me the basics of *C* programming and always helped me when I had (not always intelligent) coding questions. Furthermore I would like to thank my supervisors Christophe de Wagter and Ewoud Smeur for guiding me in the right direction and answering my (not always intelligent) questions.

This document, which now lies before you in its completed condition, is the culmination of my academic achievements so far. I hope it provides you with a good scientific read.

D.C. Höppener
Delft, October 2016

List of Figures

A.1	Block diagram of control allocator implementation	23
B.1	Roll and pitch Control in G_1	28
B.2	Yaw in G_1 and G_2	28
C.1	Coupling of the actuators	30
C.2		31
C.3	Attainable Control Set Visualization, with multiple solutions	31
C.4	Two dimensional intersection	33
D.1	Visualization of the QP solution, using the solution facet on an Attainable Control Set . .	36
E.1	Overview of the model architecture	40
E.2	Linear Controller, with feedback gains on the angular rates and attitude angle	40
E.3	Contents of the MAV block	40
E.4	Thrust Curve of Simulated Actuator	43
E.5	Reference input to linear controller, pitch doublet input	43
E.6	Reference input to linear controller, yaw step input	44
F.1	Response of Control Allocators	46
F.2	RPI CA (unconstrained) actuator response for Pitch Doublet Reference	47
F.3	RPI CA actuator response for Pitch Doublet Reference	47
F.4	WLS CA actuator response for Pitch Doublet Reference	48
F.5	WLS pq CA actuator response for Pitch Doublet Reference	48
F.6	linDCA CA actuator response for Pitch Doublet Reference	48
F.7	Computation time of Control Allocators in the second Experiment	49
F.8	Response of Control Allocators	51
F.9	RPI CA (unconstrained) actuator response for Yaw Step Reference	52
F.10	RPI CA actuator response for Yaw Step Reference	52
F.11	WLS CA actuator response for Yaw Step Reference	52
F.12	Weighted WLS CA actuator response for Yaw Step Reference	52
F.13	lin DCA actuator response for Yaw Step Reference	53
F.14	Computation time of Control Allocators in the second Experiment	53
G.1	Response of RPI and unconstrained RPI Control Allocators for pitch doublet simulation	57
G.2	Response of WLS and WLSpq Control Allocators for pitch doublet simulation	58
G.3	Response of SLS and gDCA Control Allocators for pitch doublet simulation	59
G.4	Response of lin DCA Control Allocator for pitch doublet simulation	60
G.5	Response of RPI and unconstrained RPI Control Allocators for yaw step simulation . . .	61
G.6	Response of WLS and WLSpq Control Allocators for yaw step simulation	62
G.7	Response of SLS and gDCA Control Allocators for yaw step simulation	63
G.8	Response of lin DCA Control Allocator for yaw step simulation	64

Nomenclature

Acronyms

<i>CA</i>	Control Allocator
<i>DCA</i>	Direct Control Allocation
<i>gDCA</i>	Geometric Direct Control Allocation
<i>INDI</i>	Incremental Non-Linear Dynamic Inversion
<i>linDCA</i>	Simplex Direct Control Allocation
<i>LMS</i>	Least Mean Squares Adaptive Filter
<i>MAV</i>	Micro Aerial Vehicle
<i>NDI</i>	Nonlinear Dynamic Inversion
<i>PCH</i>	Pseudo Control Hedging
<i>PD</i>	Proportional Derivative
<i>QP</i>	Quadratic Programming
<i>RePI</i>	Redistributed Pseudo Inverse
<i>RPI</i>	Regular Pseudo Inverse
<i>SLS</i>	Sequential Least-Squares method
<i>WLS</i>	Weighted Least-Squares method
<i>WLS pq</i>	Weighted Least-Squares method, prioritized for roll and pitch

Symbols

Subscripts:

f	Filtered
p	Roll axis
q	Pitch axis
r	Yaw axis

ω	Actuator angular rate vector
ϵ_{CO}	Control Objective Error metric
η_{ref}	Reference Attitude
η_{MAV}	MAV Attitude
$\dot{\Omega}$	Angular Acceleration in body frame
$\Delta\omega$	Incremental actuator angular rate vector
$\Delta\dot{\Omega}$	Incremental Angular Acceleration

c	(Incremental) Control objective
c_s	Realized (incremental) control objective
G	Control Effectiveness Matrix
T_s	Sampling time
W_v	Control objective weighting matrix
W_u	Actuator distribution weighting matrix

Contents

Abstract	iii
List of Figures	vii
Nomenclature	ix
General Introduction	1
Layout of Research Paper	1
Layout of Appendices	1
I Part 1: Scientific Paper	3
II Part 2: Appendices	21
A Implementation in Paparazzi	23
A.1 Structure and files	23
A.2 File description	24
A.2.1 Airframe file	24
A.2.2 (INDI) Attitude Controller	24
A.2.3 WLS Control Allocator	24
A.2.4 Motor Mixing	24
A.2.5 File Logger	25
B Control Effectiveness Matrices Estimate During the Compromised Actuator Experiment	27
C Applied Control Allocation Algorithms	29
C.1 Control Allocation in the INDI Controller.	29
C.2 Direct Control Allocation	29
C.2.1 Geometric DCA.	29
C.2.2 Attainable Control Objective set in R^3	31
C.2.3 Geometric DCA Solution	31
C.3 DCA: Linear Programming	33
C.3.1 Sequential Linear Programming	33
D Quadratic Programming using the Active-Set solver	35
D.0.1 Visualization of QP solution	35
D.1 Sequential Least Squares (SLS).	35
D.1.1 Weighted Least Squares Method	38
E Setup of Experiment using a Dynamic Model	39
E.1 Dynamic Model	39
E.1.1 Assumptions	40
E.1.2 Model Discrepancy	41
E.2 Performance Metrics	41
E.2.1 Error Vector	41
E.2.2 Weighted Error Vector	41
E.2.3 Magnitude Error	42
E.3 Experiment Set-Up	42
E.3.1 First experiment: Doublet Input on the Pitch Axis.	42
E.3.2 Second experiment: Yaw step input	42
E.3.3 Control Allocators.	43

F	Analysis of Experiment using a Dynamic Model	45
F.1	First Experiment: Doublet Input on Pitch axis	45
F.1.1	Analysis of Attitude Response	45
F.1.2	Analysis of Actuator Response	47
F.1.3	Computational Performance	47
F.2	Second Experiment: Yaw Step Input	49
F.2.1	Analysis of Attitude Response	49
F.2.2	Analysis of Actuator Response	50
F.2.3	Computational Performance	50
F.3	Conclusions from Simulation& Recommendations	51
F.3.1	Prioritization in Constrained Control Allocation	51
F.3.2	Limited Fidelity Model	53
F.3.3	Conclusions from the Experiments	53
F.3.4	First Experiment: Doublet Pitch Angle Input	54
F.3.5	Second experiment: Yaw Step Input.	54
F.3.6	Computational Performance	54
F.3.7	Experiment Conclusions	55
G	Simulation Output Graphs	57
G.1	Control Allocator Output Experiment 1 (Pitch Doublet Input).	57
G.2	Control Allocator Output Experiment 2 (Yaw Step Input)	58
	Bibliography	65

General Introduction

This report contains two distinct parts: a scientific paper and supporting appendices. Both parts contribute to providing an answer to the central research question for this M.Sc. Thesis:

How can control effort be prioritized to increase performance, for conflicting control objectives due to actuator constraints and coupling of control effectiveness, applied to an INDI controlled QuadCopter?

Most important in this document is the research paper presented in part I. The paper can be seen as a standalone document.

Layout of Research Paper

In chapter *I* of the paper, the introduction and problem statement are presented. The introduction moves on with a comprehensive literature review encompassing relevant research in the Control Allocation field and its application to Micro Aerial Vehicles. In chapter *II* of the paper Control Allocation within the INDI structure and the main product of this research: the Weighted Least Squares control allocator, are both presented in detail. In chapter of the paper *III* differences between control objective realization of relevant control allocators for an identified quadcopter model are presented. The quadcopter yaw and fault tolerant experiment setup is introduced in chapter *IV* followed by presentation and analysis of both flight experiment's data in chapter *V*. The results are discussed in section *VI* with the presentation of the research's conclusion in section *VII*.

Layout of Appendices

The WLS control allocator was implemented in Paparazzi. In appendix A details about the implementation and associated software structure are presented. During the flight experiments described in the paper, the control effectiveness matrices needed to be estimated with an adaptive filter. Especially interesting is the adaptation of the control effectiveness matrix during the compromised actuator experiment. The adaptive filter and the control effectiveness data is presented in appendix B.

For this research promising Constrained Control Allocation techniques were identified. A more broad overview of control allocators which were not all presented in the paper are featured in in appendix C. In this chapter, the selected Constrained Control Allocation methods are applied to the INDI controlled quadcopter.

Before executing the flight experiments, a dynamic simulation model was established of the Bebop quadcopter and control allocators to investigate which of the control allocators was most promising, and could be used to execute real flight experiments. Two experiments are executed with the simulation model in appendix E. The data from the experiments with the simulation model is thoroughly analyzed in appendix F. These experiments lead to conclusive results, which were used to select a control allocator (the WLS algorithm) which was implemented in Paparazzi. The conclusions from the simulation experiment are presented in appendix F.3. In appendix more graphs are shown which support the simulation experiments.



Part 1: Scientific Paper

Actuator Saturation Handling using Weighted Optimal Control Allocation Applied to an INDI Controlled Quadcopter

Daan C. Höppener, Ewoud J. J. Smeur, Christophe de Wagter

Abstract—Incremental Nonlinear Dynamic Inversion provides a high performance attitude controller for multi-rotor Micro Aerial Vehicles by providing very good disturbance rejection capabilities. Flights conducted with a quadcopter revealed undesired pitch and rolling motions which occurred simultaneously with actuator saturation for instantaneous yaw angle reference tracking commands. Constrained control allocation methods can increase the system's performance by providing an effective strategy to prioritize control objectives, and redistribute control effort accordingly. Weighted Least Squares control allocation makes the constrained control allocation problem a quadratic optimization problem. An iterative solver based on the computationally efficient active-set algorithm finds the optimal control distribution for a weighted control objective. In this paper the Weighted Least Squares control allocator is used to overcome two challenges 1) increase performance by applying prioritization between control objectives and redistribute control effort accordingly, accounting for the actuator limits 2) enable flight when flying with severely compromised actuator(s). Real-world flight experiments are performed and show a significant increase in performance for high load yaw maneuvers, and enabled a quadcopter to perform controlled flight with a severely compromised actuator.

NOMENCLATURE

p	Roll axis
q	Pitch axis
r	Yaw axis
G	Control Effectiveness Matrix
ω	Actuator angular rate vector [rad/s]
$\Delta\omega$	Incremental actuator angular rate vector Δ [rad/s]
c	Control objective Δ [rad/s ²]
$\hat{c}$	Realized control objective Δ [rad/s ²]
T_s	Sampling time
W_v	Control objective weighting matrix
W_u	Actuator distribution weighting matrix

I. INTRODUCTION

OVER the past decade, "Drones" or Micro Aerial Vehicles (MAVs) have gained enormous popularity. Multi-copters specifically, are very popular. Multi-copters control their position in space by directing the thrust vector to facilitate lateral and vertical accelerations. The platform is inherently unstable, and relies on controllers for stability. This conventionally, is a linear (PID) controller [1]. While robust and relative easy to implement, the overall performance of MAV can be increased by implementing a nonlinear attitude controller.

A popular nonlinear controller is Non-Linear Dynamic Inversion (NDI). This method utilizes an on-board model to cancel out non-linearity in the dynamics. To cope with un-modeled dynamics found in NDI, Incremental Nonlinear Dynamic Inversion (INDI) assumes slow dynamics of the vehicle states within an incremental time-step. Modeled and un-modeled dynamics are replaced by sensory feedback which removes the need to establish a detailed non-linear model of the dynamics. The control law relies on an effector model to generate control inputs to the control effectors.

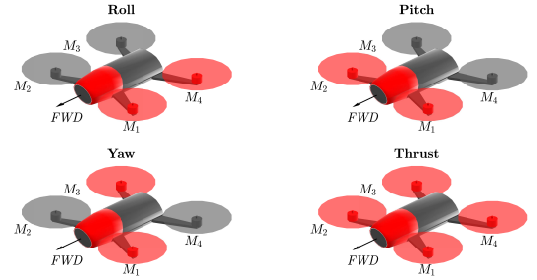


Fig. 1: Motor numbers and control signs for attitude control and total thrust of a quadcopter. A red propeller indicates a positive actuator increment, black a negative actuator increment.

In Smeur *et al.* [2], an adaptive INDI control law is derived for multi-rotor MAVs and experiments were performed with a quadcopter. Angular acceleration feedback is made possible by differentiating gyro rate measurements and filtering these. Due to incorporation of the spin-up torque of the actuators, faster yaw control is possible. The INDI controlled MAV has been shown to exhibit superior disturbance rejection capabilities compared to a PID attitude controller. This can be put to use in real world capabilities: a MAV which is better in rejecting disturbances, can fly in rougher aerodynamic disturbance conditions.

During flights conducted with an INDI controlled quadcopter, undesired pitch and rolling motions were observed for instantaneous yaw reference tracking commands of a large magnitude ($\psi > 45$ [deg]). Analysis revealed this phenomenon occurred simultaneously with actuator saturation. The system is not able to generate the control effort required to satisfy the stabilizing pitch and roll control objective as well as the yaw

control objective within the limits of the actuators. This gives rise to the constrained control allocation problem.

To illustrate this, consider the quadcopter MAV in Figure 1. The four actuators (motors), at each instant in time, provide control over four states: the roll, pitch and yaw attitude and the total thrust. The commands for each actuator are the sum of separate state commands, with the sign for the attitude states displayed in Figure 1. If the total sum of commands exceeds an actuator's limit, it is not possible to realize the full control objective and the total command needs to be limited. It is not implicit which set of state command(s) need to be limited and doing so for one actuator, directly affects the control objective realization of other actuators as each actuator provides control over all of the states. Constrained control allocation can provide a solution by providing an effective strategy to redistribute actuator effort in accordance to (prioritized) control objectives.

Available control allocation methods are briefly discussed based on existing analyses from literature[3] [4]. Generally this research focuses on control allocation applied to aircraft. First a general description of the control allocation problem is given:

For (over-actuated) systems, control allocation solves a under-determined system of equations. For $(m \times n)$ actuator effectiveness matrix G , control objective (virtual control input) $c \in \mathbb{R}^m$ and true control input to the actuators $\omega_{\min} \leq \omega \leq \omega_{\max} \in \mathbb{R}^n$, control allocation is the mapping from $\mathbb{R}^m \mapsto \mathbb{R}^n$ with: $n > m$. The primary objective is to satisfy the expression in equation 1. How different control allocators realize unattainable control objectives in combination with the use-case affects the performance of the system.

$$G\omega = c \text{ Subject to: } \omega_{\min} \leq \omega \leq \omega_{\max} \quad (1)$$

A common solution for the over actuated control allocation problem is the More-Penrose Regular Pseudo Inverse (*RPI*) [4]. The *RPI* however does not consider saturation, and does not guarantee that the realized control objective error is minimized for an unattainable control objective. The Redistributed Pseudo Inverse (*RePI*) [5] provides a solution for unattainable control objectives by iterating with the *RPI*. First a *RPI* solution is computed. For unattainable control objectives the *RPI* is recomputed with the saturated elements removed, and the control objective is updated by subtracting the control objective realized by the saturated actuators from the overall control objective. While the *RePI* is simple to implement, error minimization is not guaranteed with each iteration and no optimality criterion is present. The *RePI* generally performs poor compared to other Control Allocator (*CA*) [4].

Daisy chaining [6] groups control effectors, and calls and saturates these groups sequentially in order to achieve a control objective. This strategy is very easy to implement depending on the coupling of actuators and control objectives. Daisy chaining is highly impractical for a symmetrical quadcopter application, since the system is not over actuated and not using an actuator to achieve a control objective is not possible. Sequential satisfaction can also be applied to the control objectives. In *Buffington et al.* [7] a control allocator

is established which sequentially satisfies prioritized control objectives, applied to a tailless aircraft model.

Direct Control Allocation (*DCA*) was introduced by *Durham et al.* [8]. The main principle is to establish the Attainable Moment Set (*AMS*) which for attitude control consists of the convex hull in $\mathbb{R}^3$ containing all realizable control objectives given the actuator constraints. The attainable control set contains all combinations of actuator limits, which can be mapped to the *AMS* using actuator effectiveness matrices. For unattainable control objectives, Direct Control Allocation (*DCA*) scales the control objective such that $G\omega = \alpha c$, maintaining the ratio between individual control objectives for the solution (scalar: $0 < \alpha < 1$). In other words, the direction of the control objective vector is preserved. This is a large advantage for manually controlled systems, such as large transport aircraft. For unattainable control objectives the system moves in the expected direction. Challenges of *DCA* include establishing the full *AMS* and finding the solution, with rapidly increasing complexity for an increasing amount of actuators. Multiple geometric and linear program solvers are available [9], [10], [11], [12].

A powerful control allocation approach is to pose it as a constrained minimization problem. Quadratic Programming (*QP*) methods are applicable if the 2-norm between the realized control objective and control objective is minimized. Three solvers are commonly used, the active-set [13], fixed point [14] and interior point method [15]. The active-set method shows faster convergence and lower computational requirements for smaller control allocation problems. Posing control allocation in a *QP* formulation enables the usage of weighting matrices on the control objective and actuators, and subsequently the minimization of the weighted control objective error with the weighted actuator usage.

Control allocation methods have been researched in relation to multi-copters, focusing on fault tolerant control on larger (hexa- and octo-copter) configurations [16] [17]. *Schneider et al.* [16] makes use of parametric programming where pre-calculated solutions which correspond to the *DCA* solution are called by the *CA* for unattainable control objectives. The *DCA* solution is also used in *Frangenberg et al.* [17], but computed on-line with the redistributed pseudo-inverse. The focus in *Frankenberg et al.* is on minimizing the trajectory error on a pre-programmed flight when simulated failure of an actuator takes place. In both papers no prioritization of control objectives is present. For the complete failure of one, two actuators for a quadcopter MAV, equilibrium states and controllers are derived in *Mueller et al.* [18] and experimentally validated by exhibiting controllable flight by rotating the MAV freely about an axis fixed with respect to the body.

Although control allocation has been researched on multi-copter MAV, it's application to quadcopters may be of limited utility. Contrary to hexa- and octo-copters, quadcopters are not over actuated which makes previous research [16], [17] unlikely to provide comparable performance increase when used on a quadcopter platform. No research could be found where prioritized, optimal control allocation methods were applied to multi-rotor MAV. The results from *Mueller et al.* [18] show that full control over all of the states may be abandoned to enable

flight with reduced capabilities and suggest prioritization of states can be defined in control allocation of quadcopters.

This research aims to improve performance and stability of MAVs by prioritizing and optimizing control objective realization for unattainable control objectives using an effective control allocation strategy.

Control Allocation within the *INDI* structure and the Weighted Least-Squares method (WLS) control allocator are presented in detail in section II. In section III differences between control objective realization of relevant control allocators for an identified quadcopter model are presented. The experiment setup is introduced in section IV followed by presentation and analysis of experiment data in sections V. The results are discussed in section VI with the presentation of the conclusion in section VII.

II. METHODOLOGY

First, the *INDI* attitude controller is introduced in more detail. Consequently the WLS optimal control allocator is derived for implementation on an *INDI* controlled quadcopter. The Yaw Hedging (*YawH*) sequential control allocator is presented and used as a reference.

A. Control Allocation within *INDI* structure

In *Smeur et al.* [2] an adaptive *INDI* attitude controller is derived for multi-rotor type MAVs. An incremental relation between the angular rate of the actuators $\Delta\omega$ and the MAV's angular acceleration $\Delta\Omega$ is established.

$$\Delta\dot{\Omega} = G_1\Delta\omega + G_2(\Delta\omega - z^{-1}\Delta\omega) \quad (2)$$

In this section the control allocation within the *INDI* structure is observed. In Figure 2 the relevant parts for the control allocator are displayed. The $A(z)$ block contains a discretized model of the actuators, and the $H(z)$ block is a discrete second order filter which filters the actuator feedback. The external input to the control allocator is the control objective ν which is an incremental angular acceleration Δ [rad/s²] and follows from subtracting the measured angular acceleration of the MAV from a stabilizing angular acceleration reference [2].

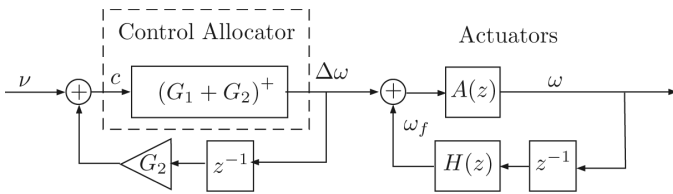


Fig. 2: Control allocation and actuators within the *INDI* attitude controller

The G_1 matrix is the inverse of the vehicle's inertia I_v^{-1} matrix multiplied by the moments originating from the actuators. The first row contains the rolling moments, the second row the pitching moments and the third row the yawing moments.

Each column in G_1 and G_2 relates to an actuator. The moments follow from the control sign, and the lateral b and longitudinal distance l to the actuators multiplied with force constant k_1 . The in-plane yaw moment follows from the moment constant k_2 [2]. Observe that roll and pitch moment generation is achieved using direct thrust of the propellers k_1 , whereas yaw control relies on the drag of the propellers to generate a yawing moment k_2 , which makes the control effectiveness in the G_1 matrix for yaw low compared to the control effectiveness in roll and pitch.

$$G_1 = 2I_v^{-1} \begin{bmatrix} -bk_1 & bk_1 & bk_1 & -bk_1 \\ lk_1 & lk_1 & -lk_1 & -lk_1 \\ k_2 & -k_2 & k_2 & -k_2 \end{bmatrix} \quad (3)$$

The G_2 matrix models the yaw moment due to changes in the angular momentum of the rotors (spin-up torque). The G_2 matrix contains the inertia of the rotors I_{rzz} around the z axis. Other inertia terms are neglected. The sampling time is denoted with T_s .

$$G_2 = T_s^{-1} I_v^{-1} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ I_{rzz} & -I_{rzz} & I_{rzz} & -I_{rzz} \end{bmatrix} \quad (4)$$

The control objective c follows from a feedback loop. The incremental angular rate vector $\Delta\omega$ of the previous time-step is multiplied with G_2 and is added to the external angular acceleration reference ν .

$$c = \nu + G_2(\Delta\omega_{z-1}) \quad (5)$$

By default control allocation is accomplished with the More-Penrose pseudo inverse denoted with $^+$. The output of the control allocator is the incremental angular rate vector $\Delta\omega$ [rad/s].

$$\Delta\omega = (G_1 + G_2)^+ c \quad (6)$$

The incremental angular rate vector $\Delta\omega$ is added to the filtered actuator feedback ω_f to deliver the actuator reference rate ω .

$$\omega = \omega_f + \Delta\omega \quad (7)$$

B. Addition of the Thrust Component

The *INDI* controller provides attitude control. Full control of the MAV requires a fourth degree: the total thrust. The thrust command follows from an external reference and is controlled outside of the *INDI* attitude control loop. For a quadcopter, the average angular rate ω_i of the actuators should be equal to the commanded rate for thrust:

$$\frac{\sum_{i=1}^4 \omega_i}{4} = c_{\text{thrust}} \quad (8)$$

C. Weighted Least Squares Control Allocator

In this section the WLS CA is derived for an INDI controlled quadcopter, using the active-set QP solver. Active-set methods are common in solvers for constrained quadratic programming and are proven to find optimal solutions in a finite number of iterations [19]. We start from the general active-set solver, displayed in algorithm 1 [19]. Here, the 2-norm of the realized control objective vector $\hat{c} = G\omega$ minus the control objective c is minimized:

$$\min_{\omega} \|G\omega - c\|^2 \quad \text{Subject to: } \omega_{\min} \leq \omega \leq \omega_{\max} \quad (9)$$

Algorithm 1: General Active-Set Method

```

for  $i = 0, 1, 2, \dots, n_{\max}$  do
  With suboptimal solution  $\omega_i$  the optimal perturbation
   $p$  is found considering the inequality constraints in
  the working set as equality constraints. Now solve:

      
$$\min_p \|G(\omega^i + p) - c\| \quad (10)$$

      
$$p_i = 0, i \in W \quad (11)$$


  if  $\omega^i + p$  is feasible: then
    Set  $\omega^{i+1} = \omega^i + p$  and compute the Lagrange
    multipliers.
    if all  $\lambda \geq 0$  then
       $\omega^{i+1}$  is the optimal solution.  $\omega = \omega^{i+1}$ 
    else
      Remove the constraint associated with the lowest
      negative value of the Lagrange multipliers  $\lambda$  from
      the working set  $W$ .
  else
    Determine the maximum increment  $\alpha$  such that
     $\omega^{i+1} = \omega^i + \alpha p$  is feasible. If this point is associated
    with a bounding constraint, add this to the working
    set.

```

With the definition from equation 9, the control allocation problem is formulated as a single quadratic optimization problem. In the WLS control allocator weights on the control objectives W_v and actuators W_u are introduced as well a reformulation of the control allocation problem. The following notations are used in algorithm 2:

- Control effectiveness: $G = G_1 + G_2$
- Control objective: $c = \Delta\Omega_{\text{ref}}$
- Control allocator output: $\omega = \Delta\omega$
- Minimum actuator increments: $\Delta\omega_{\min} = \omega_{\text{fb}} - \omega_{\min}$
- Maximum actuator increments: $\Delta\omega_{\max} = \omega_{\max} - \omega_{\text{fb}}$

Next to the primary control objective from equation 9, a secondary control objective is defined. The 2-norm of the control vector minus the desired control vector ω_d is minimized. Secondary control objectives affect the usage of actuators, for example to minimize drag originating from the control effectors [13].

$$\min_{\omega} \|\omega - \omega_d\|^2 \quad (12)$$

In equation 13 the WLS formulation of the control allocation problem is displayed. A weighting factor $\gamma \gg 1$ emphasizes that minimization of $G\omega - c$ is the primary objective. For very large ($\gamma > 10000$), the realized control objective of the WLS control allocator will approach minimizing the expression in equation 14. The matrix W_v contains the weights of the control objectives on the diagonal. The value of the weights affect the optimization and enable prioritization. A higher weight on a control objective with respect to other control objectives result in higher satisfaction of that control objective. The same holds for the secondary objective where the weights of the actuators in matrix W_u influence the usage of an actuator. A higher weight means more usage of that actuator.

$$\|W_u(\omega - \omega_d)\|^2 + \gamma \|W_v(G\omega - c)\|^2 = \left\| \begin{pmatrix} \gamma^{\frac{1}{2}} W_v G \\ W_u \end{pmatrix} \omega - \begin{pmatrix} \gamma^{\frac{1}{2}} W_v c \\ W_u \omega_d \end{pmatrix} \right\|^2 \quad (13)$$

$$\|W_v(G\omega - c)\|^2 \quad (14)$$

For convenience the constant matrix A and objective vector b are introduced.

$$A = \begin{bmatrix} \gamma^{\frac{1}{2}} W_v (G_1 + G_2) \\ W_u \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} \gamma^{\frac{1}{2}} W_v c \\ W_u \omega_d \end{bmatrix} \quad (15)$$

The general active-set algorithm 1 can now be called with the cost function from equation 15, initial working set W , initial solution ω_0 , incremental minima $\Delta\omega_{\min}$ and maxima $\Delta\omega_{\max}$ and the initial residual $d = b - A\omega_0$ to give the WLS CA in algorithm 2. The WLS algorithm minimizes the expression in equation 16.

$$\min_{\omega} \|A\omega - b\|^2 \quad \text{Subject to: } \omega_{\min} \leq \omega \leq \omega_{\max} \quad (16)$$

A large difference in the application of WLS control allocation compared to *Harkegard et al.* [13] are the maximum and minimum values for the actuators used in the optimization. In *Harkegard et al.* these are considered static whereas the incremental control law requires incremental minima $\Delta\omega_{\min}$ and maxima $\Delta\omega_{\max}$. The main difference compared to static limits is a rapidly changing attainable control set. This means that the previous solution of the control allocator might not be valid as initial solution ω_0 in the next iteration. It is possible that ω_0 is outside of the new attainable control set. To converge it is required that the initial solution ω_0 is within the attainable control set. This problem is circumvented by choosing $\omega_0 = [0 \ 0 \ 0 \ 0]^T$ and a initial working set W with one of the four constraints valid for each iteration at a cost of requiring more iterations to come to a solution.

Algorithm 2: WLS Control Allocator

for $i = 0, 1, 2, \dots, n_{\max}$ **do**
 Determine the active columns in the control effectiveness matrix: (defined by the working set):

$$A_{\text{active}} = A(:, h), \quad h \notin W \quad (17)$$

 Determine the optimal perturbation by solving:

$$p_{\text{active}} = A_{\text{active}} \mid d$$

 with: $p_{\text{active}} \notin W, p \in W = 0 \quad (18)$
if $\omega^i + p$ is feasible **then**
 $\omega^{i+1} = \omega^i + p$ and: $d = d - A_{\text{active}} p_{\text{active}}$
 The gradient and lagrange multipliers are computed with:

$$\nabla = A^T d \text{ and: } \lambda = W \circ \nabla \quad (19)$$

if all $\lambda \geq 0$ **then**
 The solution ω^{i+1} is optimal $\omega = \omega^{i+1}$;
else
 The constraint associated with the most negative λ has to be removed from the working set W .
 Re-iterate with this working set.
else
 The current solution violates a constraint which is not in W , or is on an extremum of the current set of constraints. Determine the factor $\alpha = \arg \min r$ such that ω^{i+1} is feasible, using r , the set of distances from the current solution ω^{i+1} to the set of constraints.

$$r_i = \frac{(\omega_{\min_i} - \omega_i^{i+1})}{p_{\text{active}_i}} \text{ if: } p_{\text{active}_i} < 0; \quad (20)$$

$$r_i = \frac{(\omega_{\max_i} - \omega_i^{i+1})}{p_{\text{active}_i}} \text{ if: } p_{\text{active}_i} > 0; \quad (21)$$

 Update the residual d and the solution ω^{i+1} :

$$\omega^{i+1} = \omega^i + \alpha p \quad (22)$$

$$d = d - A_{\text{active}} \alpha p_{\text{active}} \quad (23)$$

 Finally, update the working set: $W_{i_\alpha} = \text{sign}(p_{i_\alpha})$
 with: i_α the index corresponding to $\arg \min r$.
 The algorithm stops when the control objective is achieved or a maximum number of iterations $i_{\max}$ has been reached, or all $\lambda > 0$.

D. Yaw Hedging

Because of the high control effort associated to executing yaw maneuvers with quadcopters, a logical approach would be to limit the total yaw control objective, based on available capabilities of the actuators after the control effort from roll and pitch is subtracted. This can be considered sequential control objective prioritization [7]. Algorithm 3 computes the realizable yaw control objective from the actuator with the largest absolute overflow over the actuator limits of a temporary command ω_{tmp} .

The total thrust of the MAV is controlled outside of the attitude loop (from Figure 2). In order to not influence the total thrust control from equation 8, symmetrical control effectiveness matrices of the MAV are assumed for the YawH CA which are displayed in equations 24 and 25.

$$G_{1\text{symmetric}} = \begin{bmatrix} G_{1_p} & -G_{1_p} & -G_{1_p} & G_{1_p} \\ G_{1_q} & G_{1_q} & -G_{1_q} & -G_{1_q} \\ -G_{1_r} & G_{1_r} & -G_{1_r} & G_{1_r} \end{bmatrix} \quad (24)$$

$$G_{2\text{symmetric}} = [G_{2_r} \quad -G_{2_r} \quad -G_{2_r} \quad G_{2_r}] \quad (25)$$

Algorithm 3: RPI with Yaw Hedging

Compute the individual actuator contributions for the different control objectives:

$$\Delta\omega_p = G_{1_p}^+ \Delta\dot{\Omega}_p, \quad \Delta\omega_q = G_{1_q}^+ \Delta\dot{\Omega}_q, \\ \Delta\omega_r = (G_{1_r} + G_2)^+ \Delta\dot{\Omega}_r \quad (26)$$

Create a temporary actuator command containing the actuator commands originating from the attitude control objective, added to the current actuator state (actuator feedback ω_{fb}) and the total thrust requirement (ω_{thrust}):

$$\omega_{\text{tmp}} = \Delta\omega_p + \Delta\omega_q + \Delta\omega_r + \omega_{fb} + \omega_{\text{thrust}} \quad (27)$$

The absolute maximum overflow of ω_{tmp} , over either the lower or maximum limit of one of the actuators, determines the hedging of the yaw control objective.

$$\omega_{\text{hedge}} = \max \mid \min (\omega_{\text{tmp}} < \omega_{\min}), \\ \max (\omega_{\text{tmp}} - \omega_{\max}) > 0 \mid \quad (28)$$

if $\omega_{\text{hedge}} > 0$ **then**

Scale the yaw actuator commands with the absolute overflow ω_{hedge} :

$$\Delta\omega_r = \Delta\omega_r - \text{sign}(\Delta\omega_r) \omega_{\text{hedge}} \quad (29)$$

The actuator command is now:

$$\omega = \Delta\omega_p + \Delta\omega_q + \Delta\omega_r + \omega_{fb} + \omega_{\text{thrust}} \quad (30)$$

III. ATTAINABLE CONTROL OBJECTIVES

The constrained control allocation problem can be visualized using the Attainable Moment Set (AMS). The AMS defines the full set of attainable control objectives, constrained by the available actuator capabilities (for this research, we are discussing incremental accelerations, so AMS is not correct. This nomenclature is retained however, as it is standard in control allocation literature [8]). For n actuators, a n dimensional control space is considered with incremental actuator angular velocities $\Delta\omega \in \mathbb{R}^n$. The subset Θ contains the maximum and minimum increments, available to the actuators: $\mathbf{f}$

$$\Theta = \{\Delta\omega \in \mathbb{R}^n \mid \Delta\omega_{i,\min} \leq \Delta\omega_i \leq \Delta\omega_{i,\max}\} \subset \mathbb{R}^n \quad (31)$$

The subset of attainable values for the actuators is referred to as the attainable control set and can be mapped to the AMS in $\mathbb{R}^m$, with $n \geq m$. This linear mapping H follows from equation 32. The matrix D contains all combinations of maximum and minimum increments of the actuators. With H , as displayed in equation 33, the points which span a convex hull $d(\Phi) \in \mathbb{R}^m$ are defined. The convex hull $d(\Phi)$ is referred to as the AMS.

$$H : \mathbb{R}^n \rightarrow \mathbb{R}^m \quad (32)$$

$$H = \Phi \quad \text{where: } H = (G_1 + G_2) D \quad (33)$$

A. Visualization of Control Objective Realization

The AMS can be used to illustrate the differences between the control objective realization properties of the control allocators. We observe the realization of a mixed incremental control objective $c_p = -35$, $c_q = 28$, $c_r = 7 \Delta[\text{rad/s}^2]$, constrained by maximum and minimum actuator increments $\Delta\omega_{\max}$ and $\Delta\omega_{\min}$ of 1000 [RPM] and -3000 [RPM] respectively. The thrust requirement should also be satisfied, which entails $\Sigma\Delta\omega = 0$.

Two cases are introduced: in case 1, the MAV has perfectly symmetrical control effectiveness. In case 2, the control effectiveness of the 3rd column of G_1 and G_2 is reduced to 20%, corresponding to a severe actuator failure. As such, the control effectiveness matrix is (highly) asymmetric. The values in the control effectiveness resemble the Parrot Bebop quadcopter and were identified during a test flight using the Least Mean Squares filter and structure from *Smeur et al.* [2]. The realized control objectives of the following control allocators are presented:

- RPI (over saturation)
- Clipped RPI
- WLS ($W_p = 1$, $W_q = 1$, $W_r = 1$)
- DCA
- YawH

In Figures 3 and 4 the attainable moment set for attitude control is visualized for both cases. The convex hull for the quadcopter has the shape of a tetrahedron. The percentages of the realized control objective as well as the sum of actuator increments for both cases is presented in table I.

The RPI CA fully realizes the control objective, but violates the actuator constraints which makes this solution infeasible. If the RPI solution is clipped, in accordance to the actuator constraints, the clipped RPI solution is obtained. For this solution the sum of the actuator increments is larger than zero which causes it to be outside of $d(\Phi)$. The DCA solution is the intersection between the control objective and $d(\Phi)$. It satisfies the control objective as much as possible maintaining the control objective direction. The WLS control allocator searches for the weighted optimal solution over the convex hull. A colormap on the solution facet shows the WLS solution optimality over the facet.

B. Analysis of Realized Control Objectives

Clipping the RPI solution changes the direction of the realized control objective. This yields poor performance of the clipped RPI solution on control objective realization as well as failing to satisfy the thrust requirement. The DCA solution's performance improves over the clipped RPI solution, but is highly dependent on the shape of the AMS. Note the scaling on the axes in Figures 3 and 4. The range on the yaw ($\Delta\dot{\Omega}_r$) axis is much lower than the range on the roll ($\Delta\dot{\Omega}_p$) and pitch ($\Delta\dot{\Omega}_q$) axes due to the low control effectiveness in yaw (see section II-A). For mixed control objectives with unattainable yaw component and attainable roll and pitch components, the DCA solution doesn't converge to a solution which satisfies the roll and pitch component. The WLS and YawH solutions are very similar for the symmetric case, both satisfying the roll and pitch control objective. For the asymmetric case, the YawH also satisfies the roll and pitch control objective to a high degree. Compared to the WLS solution the control objective realization is lower. This can be attributed by the control effectiveness error the YawH from algorithm 3 inherently contains, because a symmetrical control effectiveness matrix is required.

From the introduced CAs the WLS CA is the only CA which defines an optimality criterion, minimizing the expression in equation 16. It shows that for the mixed control objective in both cases the highest *total* control objective realization can be achieved by having high roll and pitch and low yaw control objective realization. This translates to the WLS solution having the best overall score in table I. The control objective realization is the highest for both the symmetrical and asymmetrical cases compared to other CAs and the thrust constrained is not violated.

The YawH solution is closest to the WLS solution in both cases. For symmetrical cases, it is expected that the YawH provides performance close to the WLS control allocator. However, once more asymmetry is present in the G_1 and G_2 the differences between the realized control objective between the WLS and YawH CA will diverge. The WLS CA is able to maximize the realized control objective by optimizing the actuator usage. It is expected that the performance in any case for the WLS CA is superior to the other CAs.

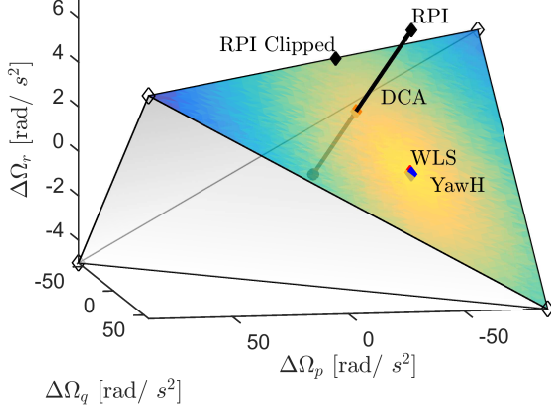


Fig. 3: Visualization of the AMS $d(\Phi)$ with symmetric control effectiveness matrix, with the *RPI*, *DCA*, *WLS*, *YawH* realized control objectives visualized. The colormap displays the *WLS* optimality over the solution facet and the control objective is displayed as the black arrow

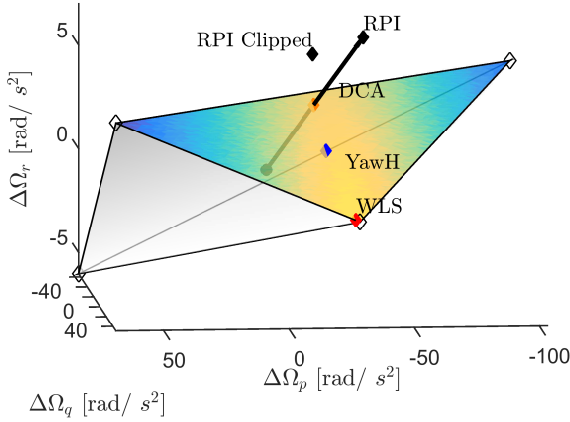


Fig. 4: Visualization of the AMS $d(\Phi)$ with asymmetric control effectiveness matrix, with the *RPI*, *DCA*, *WLS*, *YawH* realized control objectives visualized. The colormap displays the *WLS* optimality over the solution facet, and the control objective is displayed as the black arrow

IV. EXPERIMENT SETUP

To assess the performance of the *WLS* CA (implemented in the adaptive *INDI* attitude control loop) from section II-C a set of experiments are executed using a quadcopter. Two experiments will be conducted testing the key properties of the *WLS* CA: A) Performance and B) Stability. For comparison the clipped *RPI* and *YawH* control allocators are included in the experiments.

For the experiments a Parrot Bebop quadcopter is used equipped with Paparazzi open-source autopilot software. The Paparazzi autopilot contains modules with all the requirements

TABLE I: Control Allocator Control Objective Realization

	Symmetric				Asymmetric			
	$\Delta\dot{\Omega}_p$	$\Delta\dot{\Omega}_q$	$\Delta\dot{\Omega}_r$	$\Sigma\Delta\omega$	$\Delta\dot{\Omega}_p$	$\Delta\dot{\Omega}_q$	$\Delta\dot{\Omega}_r$	$\Sigma\Delta\omega$
RPI	100%	100%	100%	0	100%	100%	100%	-1436
RPI c	21%	34%	77%	-1289	47%	56%	85%	-2294
DCA	44%	44%	44%	0	49%	49%	49%	0
WLS	99%	98%	9%	0	93%	85%	-27%	0
YawH	100%	100%	8%	0	61%	68%	19%	0

for the implementation of the *WLS* algorithm. An important feature of the Parrot Bebop quadcopter is the availability of direct feedback from the actuators, to determine the maximum and minimum actuator increments available for control. The accelerometer, gyroscope, and control loops were running at 512Hz. The attitude control loop from Figure 2 is used. The CAs replace the output of the $(G_1 + G_2)^+$ block in the diagram. The *WLS* and other CAs are programmed in the C language.

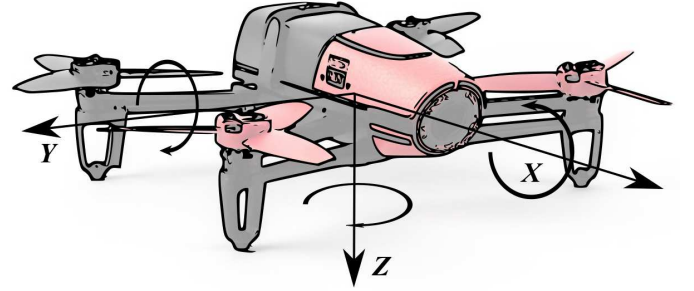


Fig. 5: Parrot Bebop quadcopter with axis definitions

A. Performance

In previous flights with the *INDI* attitude controlled Bebop it was observed that the *MAV* shows unfavorable rolling and pitching motion for high amplitude yaw rate changes. Due to the low control effectiveness of the *MAV* in yaw compared to the control effectiveness on the roll and pitch axes, yaw maneuvers require comparatively high control effort. It is discussed and visualized in section III. To put the CAs to the test an experiment is introduced where the *MAV* receives a 90 [deg] step heading reference while hovering 1 [m] over a reference position. Important for this experiment is the high instantaneous yaw rate reference.

The following control allocators are compared:

- Clipped *RPI*
- *WLS* ($W_p = 10, W_q = 10, W_r = 1$)
- *WLS* ($W_p = 1, W_q = 1, W_r = 1$)
- *YawH*

During the experiment, the *MAV*'s position, heading and height is sent to the *MAV* via WiFi and is obtained by

the Optitrack positioning system. OptiTrack triangulates the position of infrared markers placed on the MAV using a system of infrared cameras. The MAV's onboard guidance controller receives the position, height and heading reference and generates a reference attitude and thrust command for the MAV's attitude control loop. The input to the guidance control loop follows from a pre-configured flight plan. Waypoints in the flight plan can be called remotely with a laptop, which communicates to the MAV via WiFi. For the experiment, a fixed waypoint position is used, where the heading reference of the MAV on the position can be changed. Sensor data, reference input, and output variables are logged on-board the MAV using an integrated data logger.

From the example cases from III, we can get a feeling for the performance of different CAs. Depending on the symmetry of G_1 and G_2 matrices, the *YawH* and *WLS* CA will exhibit comparable responses. The clipped *RPI* CA may show undesired rolling and pitching motion, as the clipped actuator commands will distort achieving the desired control objective.

B. Stability

The goal of this experiment is to show that by using optimal, prioritized control allocation, a controllable and flyable stability of the MAV can be achieved, even when full control over the attitude and vertical acceleration is reduced. For this experiment, the blades of the right aft rotor were cut-off, reducing the total diameter of the rotor with 4.8 [cm], which can be seen in Figure 6. The normal diameter is 13.4 [cm] compared to 8.6 [cm] for the clipped rotor. The disk area is reduced from 141 [cm²] to 58.1 [cm²] which is a reduction of 58.9 %.

By severely compromising a single actuator on a quadcopter, the constrained control allocation problem becomes more pressing. There are four actuators and four degrees of freedom, over which each actuator has a similar control effectiveness (high symmetry in G_1 and G_2). As one actuator is severely compromised, G_1 and G_2 need to be re-identified, and are likely to become highly asymmetrical. The MAV may not be able to maintain full control authority over the full attitude and vertical thrust. For maintaining flight, the roll and pitch attitude enjoy a higher priority than the yaw attitude. The *WLS* CA in section III with unit weights on the roll, pitch and yaw axes, has been shown to already find an optimal solution (compared to the other CAs) by having high roll and pitch control objective realization compared to the yaw control objective realization. For this experiment, the roll and pitch control objective realization can be increased by having larger weights on the roll, pitch attitude relative to the weight on the yaw attitude.

V. RESULTS

In this section the results from experiments introduced in IV are presented. First the performance experiment is presented followed by the stability experiment with compromised aft right actuator. Repeating the experiments yielded comparable results.



Fig. 6: Reduced rotor diameter by [4 cm] on right-aft rotor

A. Performance: Yaw Step Reference Input

To maximize the performance of the *WLS* CA the control effectiveness matrices are estimated with the Least Mean Squares (*LMS*) filter which is also used in [2]. Note that some asymmetry is present, especially for the yaw effectiveness third column of the G_1 matrix, which is rather lower compared to the other columns. This might seem a big error, but when we add the G_2 yaw control effectiveness values to the G_1 matrix this discrepancy becomes less severe. The matrices were estimated during an estimation flight and are kept constant during the experiments:

$$G_1 = \frac{1}{1000} \begin{bmatrix} 20.57 & -20.13 & -20.07 & 19.83 \\ 11.75 & 12.40 & -12.40 & -12.71 \\ -1.774 & 1.461 & -0.3403 & 0.596 \end{bmatrix} \quad (34)$$

$$G_2 = \frac{1}{1000} \begin{bmatrix} -64.57 & 63.091 & -66.57 & 73.64 \end{bmatrix} \quad (35)$$

The *YawH* CA requires symmetrical matrices as discussed in section II-D. For the values of $G_{1p}, G_{1q}, G_{1r}, G_2$ the average absolute values from the identified matrices are taken.

Figure 7 shows the yaw angle response and reference for the 90 [deg] step input. In Figure 8, 10, 12 and 14 the pitch and roll angle response and reference is shown. The angular rates of the actuators are displayed in Figure 21, to 24. Note that *RF* denotes Right Front, *LR* Left Rear, etc. The achieved and reference angular accelerations of the CAs are presented in figures 16 to 18. For better comprehension the attitude of the MAV for each control allocator is displayed in Figures 9, 11, 13, 15.

By visually inspecting Figure 7 small performance differences are noticeable between the different CAs. The *RPI* and *YawH* CAs show the fastest yaw rate accelerating to the reference yaw angle. The overshoot of the *RPI* and *WLS* CA

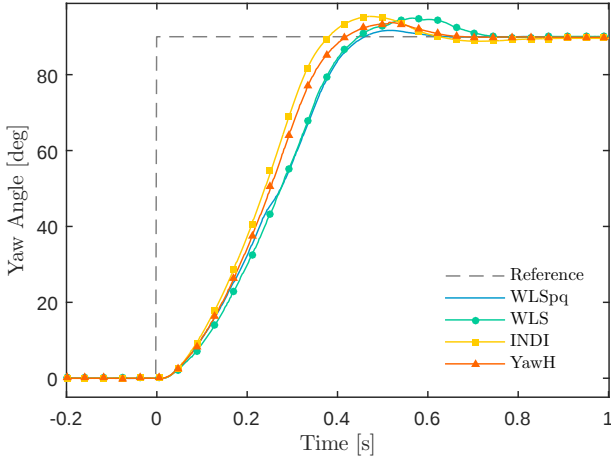


Fig. 7: Yaw angle step response

are similar and are larger than for the *YawH* CA. The prioritized *WLS* CA shows virtually no overshoot. The *WLS* and *RPI* CAs both converge in 0.76 seconds. The prioritized *WLS* and *YawH* CA converge faster, but due to the smaller overshoot of the prioritized *WLS* CA, the total convergence time is the less at 0.65 seconds. Both the overshoot as the convergence time are the smallest for the *WLS* prioritized for pitch and roll.

For optimal performance, the convergence towards the reference yaw angle should be as fast as possible, while also tracking the roll and pitch reference. The roll and pitch reference in this experiment follows from the guidance controller which maintains the position of the *MAV* while hovering. The *MAV*'s roll and pitch angle define the direction of thrust and poor roll and pitch reference angle tracking performance will yield undesired translational motion. The tracking performance of the clipped *RPI* CA shown in Figure 12 exhibits the largest departure from the reference. The pitch angle error exceeds more than 15 [deg]. In Figure 10 the *WLS* CA also shows a significant departure from the reference roll and pitch angle, with a roll error exceeding 10 [deg] and a maximum pitch error over 8 [deg]. Prioritization of the *WLS* CA for roll and pitch manifests in much better tracking performance visible in Figure 8. The *YawH* CA in Figure 14 performs better than the *WLS* and *RPI* CAs, but still shows a pitch tracking error exceeding 3 [deg].

Next to the effect of the CA's on the performance of the *MAV* we are interested how the CAs realize control objectives. The realized control objective $\hat{c}$ follows from multiplying the *realized* incremental actuator command $\Delta\omega_r$ with the control effectiveness matrices.

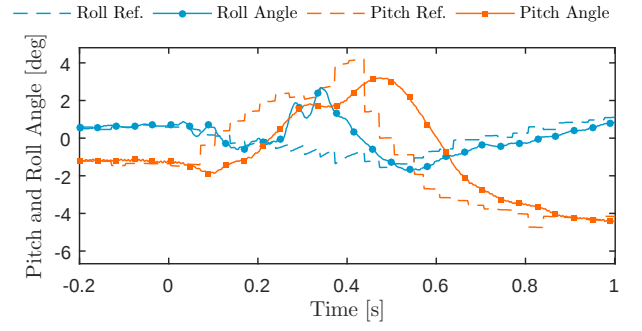


Fig. 8: Roll and pitch angle during yaw step input for the *WLS* CA prioritized for pitch and roll

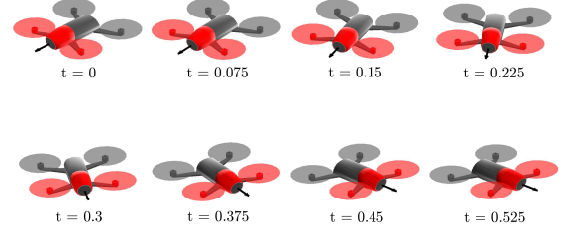


Fig. 9: Visualization of the attitude for the *WLS* CA prioritized for pitch and roll

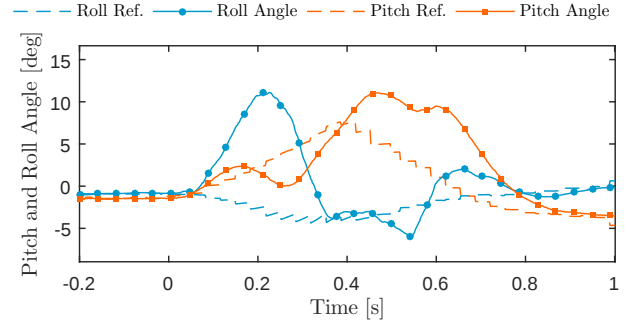


Fig. 10: Roll and pitch angle during yaw step input for the regular *WLS* CA

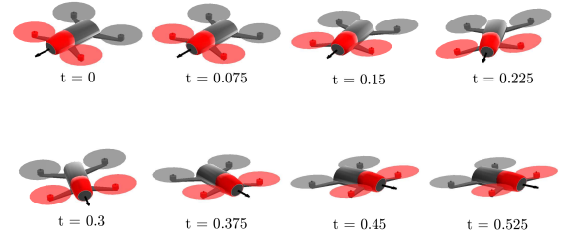


Fig. 11: Visualization of the attitude for the *WLS* CA

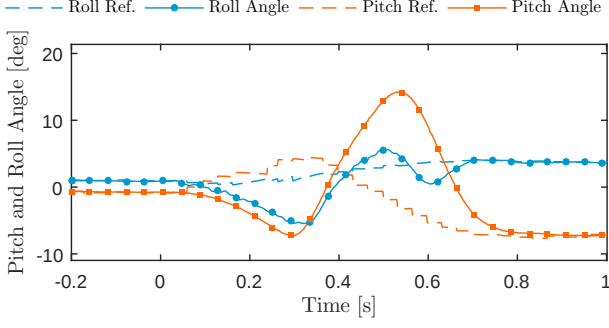


Fig. 12: Roll and pitch angle during yaw step input for the unconstrained *RPI CA*

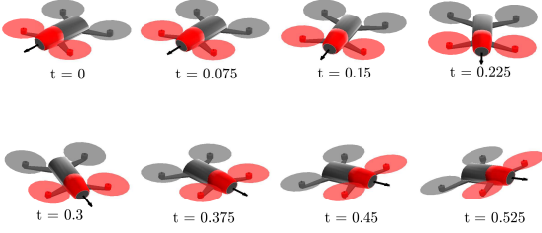


Fig. 13: Visualization of the attitude for the the unconstrained *RPI CA*

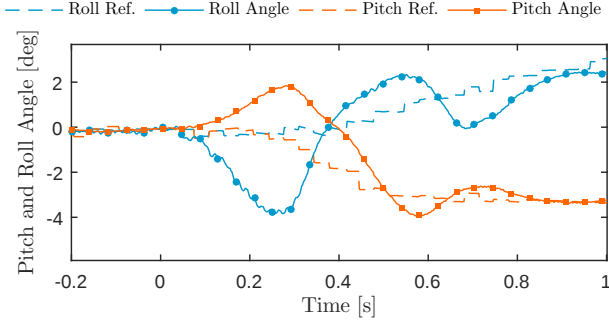


Fig. 14: Roll and pitch angle during yaw step input for the *YawH CA*

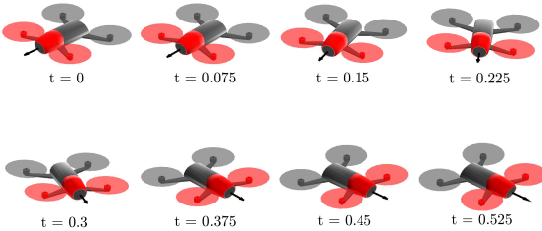


Fig. 15: Visualization of the attitude for the *YawH CA*

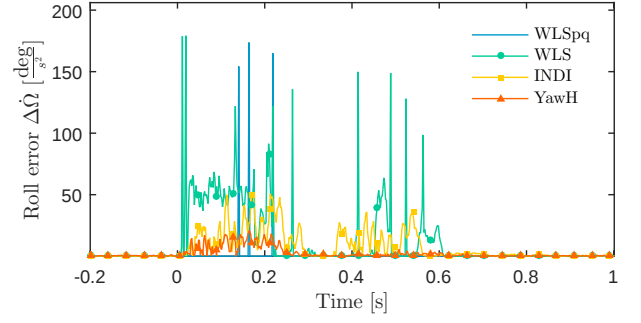


Fig. 16: Comparison of Roll realized control objective error

$$\hat{c} = [G_1 + G_2] \Delta\omega_r \quad (36)$$

For the prioritized *WLS*, normal *WLS* and *YawH CAs* $\Delta\omega_r$ is defined as the *CA* output. For the *RPI CA* the output is not clipped and the realized control objective $\hat{c}$ will be the same as the control objective c . Therefore the realized control objective for the *RPI CA* is defined as the rate send to the actuators ω_{ref} with the filtered actuator feedback ω_{fb} subtracted:

$$\Delta\omega_{r,RPI} = \omega_{ref} - \omega_{fb} \quad (37)$$

To present the realized control objective data compactly, the absolute error $\epsilon = |c - \hat{c}|$ is compared for each *CA*, for each control objective axis in figures 16, 17 and 18. Observing Figure 18 there is a large discrepancy between the prioritized *WLS* and *YawH CAs* and the *RPI* and regular *WLS CAs*. Contrary to what one would expect the errors for the prioritized *WLS* and *YawH CAs* are much smaller. This can be explained by observing the *INDI* control scheme (displayed in Figure 2) as the yaw control objective follows from $c_r = \nu_r + G_2\Delta\omega_{z-1}$. The $\Delta\omega$ output for the prioritized *WLS* and the *YawH CAs* are hedged in accordance with the *MAV's* capabilities and have a lower value, hence a lower $G_2\Delta\omega_{z-1}$ feedback is added to the yaw control objective in the next time step. To illustrate this, the yaw control objective c_r , realized yaw control objective $\hat{c}_r$ and the $G_2\Delta\omega_{z-1}$ feedback are compared for the prioritized *WLS* and *RPI CAs* in Figures 20 and 19. It can be observed that the G_2 feedback makes up the bulk of the yaw control objective. The unconstrained $\Delta\omega$ output of the *RPI CA* generates a very large yaw control objective which cannot be achieved. The prioritized *WLS* on the other hand generates incremental output $\Delta\omega$ in accordance with the actuator's capabilities, which leads to realistic, achievable yaw control objective c_r .

The rotational rates of the actuators show how different *CAs* use the available actuator capacities. The *RPI* and *YawH CAs* in figures 23 and 24 show the most pronounced saturation of actuators to execute the yaw maneuver, with periods where all

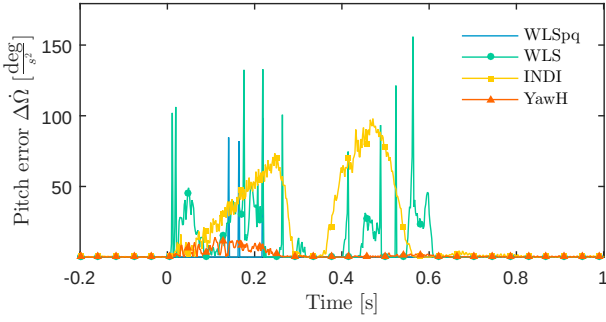


Fig. 17: Comparison of Pitch realized control objective error

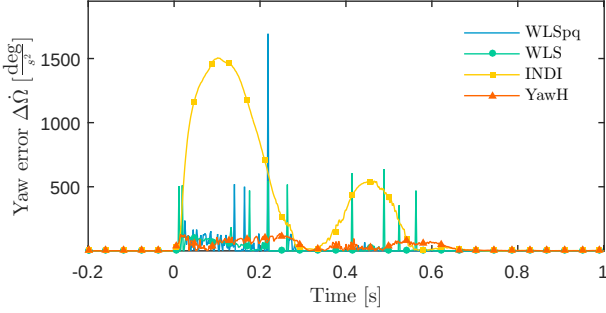


Fig. 18: Comparison of Yaw realized control objective error

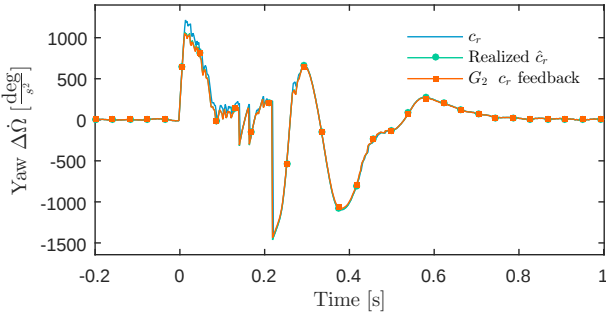


Fig. 19: Comparison of yaw control objective, realized yaw control objective and G_2 feedback yaw control objective for the WLS CA prioritized for pitch and roll

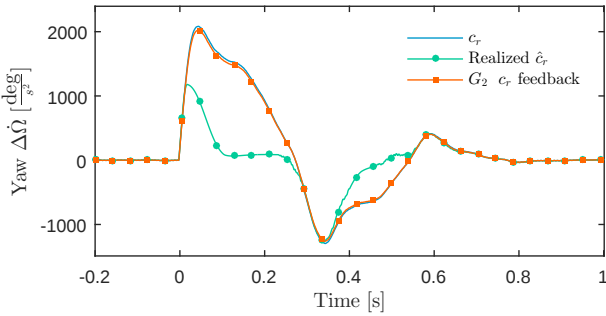


Fig. 20: Comparison of yaw control objective, realized yaw control objective and G_2 feedback yaw control objective for the RPI CA

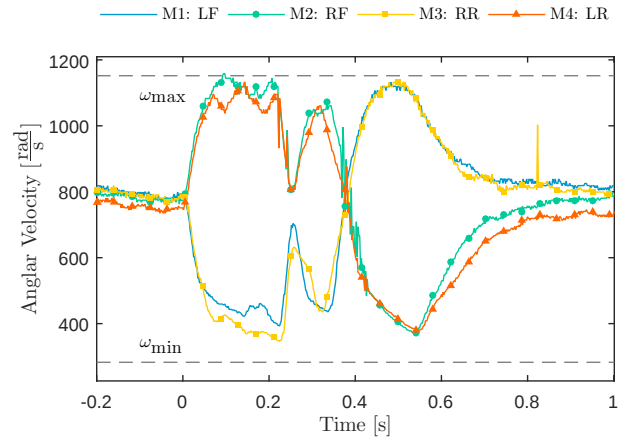


Fig. 21: Actuator angular rates during the yaw step input of the WLS CA prioritized for pitch and roll

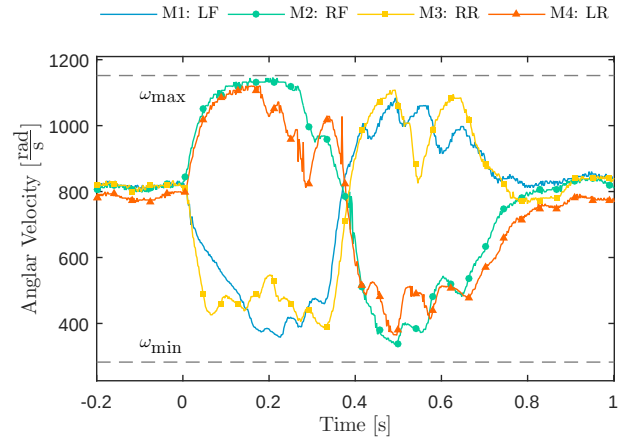


Fig. 22: Actuator angular rates during the yaw step input of the WLS CA

four actuators are saturated. The maneuver is characterized by two phases: first motors 2 and 4 rapidly spin up, while motors 1 and 3 decrease their rotational rate, to accelerate the MAV's yaw rate. After 0.55 seconds (for the RPI CA) the opposite happens, to decrease the yaw rate. This double phase pattern is also recognizable for the prioritized WLS and regular WLS CAs in figures 21 and 22. Here however, the double phase pattern is less pronounced when compared to the RPI and YawH CAs. The actuator rates oscillate within the dual phase pattern, and no saturation of all 4 actuators takes place.

B. Stability: Compromised Actuator Experiment

In the second experiment, we investigate the stability and performance of a quadcopter with a compromised rotor, as

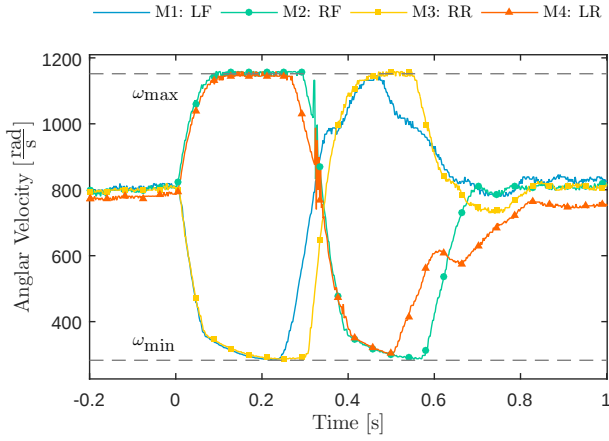


Fig. 23: Actuator angular rates during the yaw step input of the *RPI CA*

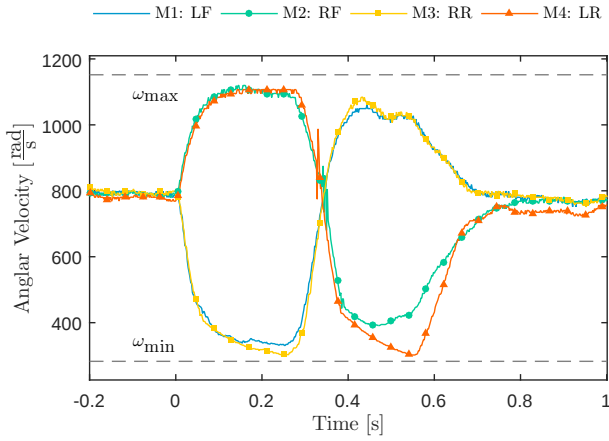


Fig. 24: Actuator angular rates during the yaw step input of the *YawH CA*

introduced in section IV-B. To achieve this, the actuator effectiveness matrices were re-identified:

$$G_1 = \frac{1}{1000} \begin{bmatrix} 17.43 & -33.48 & -13.73 & 35.43 \\ 9.660 & 8.980 & -1.605 & -10.37 \\ -3.790 & 2.755 & 0.4701 & 1.028 \end{bmatrix} \quad (38)$$

$$G_2 = \frac{1}{1000} [-129.9 \quad 79.16 \quad -10.08 \quad 68.77] \quad (39)$$

For re-identification, the *LMS* filter from [2] was used. The identification followed from a test flight with the prioritized *WLS CA*. Obtaining an actuator effectiveness matrix was challenging, as this was done without the “*Careless*” flight mode (explained further in this section), and a large variance of the values was present throughout the flight. The values which are presented followed from a (relatively) stable hover phase of the flight where the variance was lower. The identified control effectiveness were also used on the *YawH* and clipped

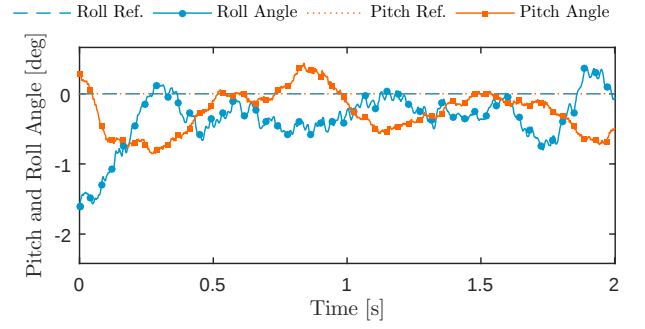


Fig. 25: Roll and pitch angle during compromised actuator experiment

RPI CAs, and the logged experiment. The values are somewhat surprising, as the control effectiveness of non-damaged actuators diverges from the undamaged values in equations 34 and 35. To be able to execute comparable experiments, it was chosen to use the values from equation 38 and 39, given that these might be flawed but a better solution was not available.

Before logging an experimental flight, the *WLS*, prioritized *WLS*, clipped *RPI* and *YawH CAs* were tested to show stability using a pilot controlled attitude controlled flight mode. The pilot controls the *MAV*’s roll, pitch and yaw reference angle as well as the total thrust manually using an input device. The *WLS*, *RPI* and *YawH CAs* did not manage to fly, crashing instantly at take-off before gaining height. The prioritized *WLS* did manage to take-off and was able to gain altitude and keep the roll and pitch angle bounded such that sustained flight was possible. While in the air, the *MAV* started spinning around the *MAV*’s z body axis, similar to what was demonstrated in [18].

In the attitude flight mode the pilot controls the roll and pitch reference angle in the vehicle carried reference frame. Due to the high yaw rate present it is very hard (close to impossible) to control the *MAV*’s position with attitude control mode, as the yaw angle is constantly changing. For a quadcopter it is possible to decouple the orientation from the thrust vector from the *MAV*’s yaw angle [20]. The Paparazzi autopilot contains a “*Careless*” flight mode, where the pilot controls the roll and pitch angle of the *MAV*’s thrust vector, relative to a fixed yaw reference, detached from the *MAV*’s yaw angle. This flight mode requires an accurate and high rate of the *MAV*’s heading angle for which the on-board magnetometer proved to be insufficient. The Optitrack system is used to provide *MAV*’s heading angle. In the second experiment flights the *WLS CA* optimizes for the roll and pitch control objective realization ($W_p = 100$, $W_q = 100$, $W_r = 1$ and $W_{thrust} = 10$).

The *MAV*’s position could be manually controlled as long as the pilot did not steer to aggressively. In Figures 25 and 26 the *MAV*’s attitude is presented, taken from a 2 [s] interval where the *MAV* was hovering very stable and no pilot input was required, except for maintaining the altitude

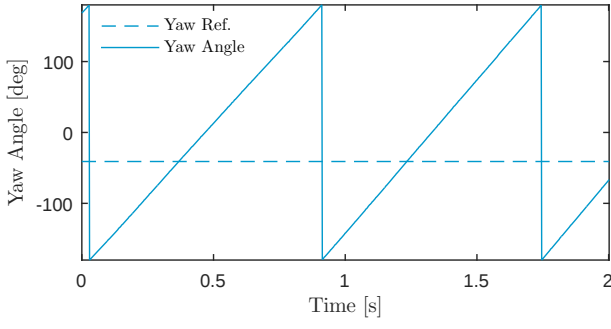


Fig. 26: Yaw angle during the compromised actuator experiment

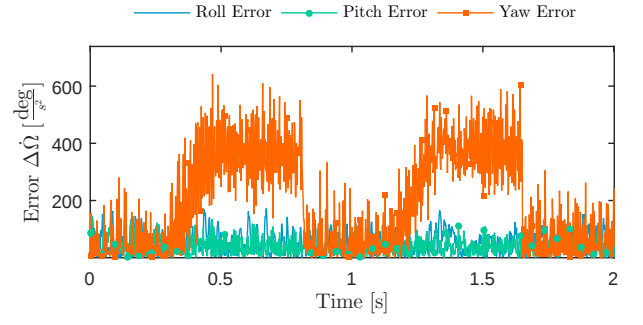


Fig. 28: Comparison of realized control objective error during the stability experiment for roll, pitch and yaw

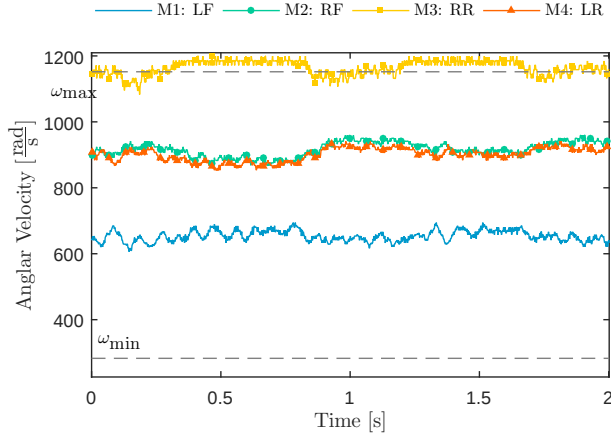


Fig. 27: Actuator angular rates during the compromised actuator experiment

by controlling the total thrust. As can be observed, the yaw angle control objective realization is largely abandoned by the *WLS CA*. The pitch and roll reference is still tracked by the *WLS CA* which is able to keep the absolute error within 1.5 [deg] of the reference. A more intuitive depiction of *MAV*'s attitude during the stability experiment for a 0.525 [s] interval is given in Figure 29. From Figure 27, we can observe that the damaged actuator is saturated, even slightly exceeding the maximum rate $\omega_{\max}$. The actuator diagonally opposite has the lowest angular rate to maintain a force equilibrium. The two actuators on the other diagonal have similar rates and make up the bulk of the thrust.

The absolute error between the realized control objective and the control objective $\epsilon = |c - \hat{c}|$ for the roll, pitch and yaw attitude is displayed in Figure 28. The data is very noisy, it is however distinguishable that the yaw control objective error is the largest, and cycles corresponding to the yaw attitude of the vehicle. The roll and pitch control objective is lower and shows a constant bandwidth for the experiment duration.

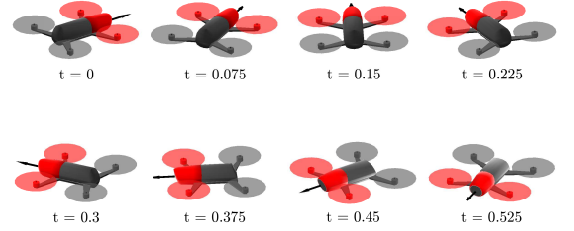


Fig. 29: Visualization of the attitude during the stability experiment

VI. DISCUSSION

This research aimed to improve performance of a quadcopter *MAV* equipped with *INDI* attitude controller by optimizing the use of actuators to prioritize and satisfy control objectives. This is achieved by providing an optimized solution to the constrained control allocation problem bounded by the available incremental rates to the actuators.

From the results of the first experiment, the *WLS CA* prioritized for roll and pitch shows the most optimal performance. Not only does the prioritized *WLS* show the smallest error on the roll and pitch reference tracking task, the convergence towards the yaw reference angle is also the fastest. In this research the weights for the trade-off have been taken such that $W_p, W_q \gg W_r$, in the first experiment by a factor 10. No justification for this number was made, except for the hypothesis that the roll and pitch angle which define the direction of the thrust vector for a *MAV* is more important than the heading of the *MAV*. Further research may lead to adaptive weights for W_v , or a derivation which justifies these weights.

In section III it was shown that for a symmetric control effectiveness matrix, the *WLS* solution for unattainable control objectives was almost identical to the *YawH* solution. However, the obtained results from the yaw reference experiment are different. One cause is that the identified quadcopter control effectiveness (displayed in equations 34, 35) is not perfectly symmetrical. The *YawH CA* does not optimize depending on the available actuator increments, instead hedging the yaw

command to the lowest absolute minimum available increment.

The *WLS CA* provides a solution for the constrained control allocation problem. However, when saturation occurs, the reference signal may also be adjusted. For this anti-windup schemes [21] and Pseudo Control Hedging [22] can be used.

The second experiment demonstrated the capability of the *WLS CA* to provide control over the pitch, roll and thrust of a quadcopter with a very asymmetrical control effectiveness matrix. Contrasting to the results from [18], no prior derivation of the adjusted dynamics was made. While the results show excellent capabilities of maintaining the attitude, the prioritized *WLS CA* relies on a reasonable estimate of the actuator effectiveness. With adaptive actuator effectiveness which was estimated using the *LMS* filter, the *MAV* was able to take off without prior knowledge of the failed actuator. If the actuator would fail while in flight, the *LMS* must be able to converge fast to a new estimate in order to avoid a crash. Control over a quadcopter spinning around its z axis introduces a new unmodeled effect which the *INDI* attitude controller does not take into account. The spinning rotation introduces gyroscopic moments. It can be established if this effect is relevant, and if so, derivation of an adjusted *INDI* attitude control loop.

To fully exploit the fault tolerant capabilities of the prioritized *WLS* control allocator, experiments may be conducted with a different airframe, as the Parrot Bebop does not allow the motors to go below 3000 [RPM] while enabled. Not having this constraint as a minimum reduces the thrust generating requirement for the damaged actuator as the actuator diagonally opposite does not generate the "static" thrust resulting from the minimum angular velocity. With the *WLS CA* various configurations of compromised actuators should be possible to still be able to maintain stable flight, for example two compromised actuators on the *MAV*'s diagonal or three compromised actuators. The damaged actuator should always be able to generate some thrust to maintain a zero pitch and roll angle. The *MAV* still relies on four actuators to maintain the attitude. For a fully fault tolerant flight system, the outer loop should also be considered, for example to limit the maximum rates which can be allowed as a reference.

VII. CONCLUSION

The Weighted Least-Squares method (*WLS*) Control Allocator (*CA*) provides a flexible, computationally efficient and optimized solution to the constrained control allocation problem, which occurs when a control objective is not realizable due to the actuator constraints. With the *WLS CA* weights can be assigned to control objective realization, which enables non-sequential prioritization of states. The *WLS CA* is implemented in an adaptive Incremental Nonlinear Dynamic Inversion (*INDI*) controlled *MAV*. Because of the low control effectiveness for yaw control, the actuators saturate rapidly for sustained yaw maneuvers and make it a very demanding control task. Using the default Regular Pseudo Inverse (*RPI*), the actuator control input has to be clipped, and can lead to high roll and pitch angle reference tracking errors. The *WLS CA* with high weights for the roll and pitch control objective compared to the weight for the yaw control objective will yield

a solution where the yaw maneuver has very little influence on the roll and pitch control objective realization. Experiments validated the ability of the prioritized *WLS CA* to limit roll and pitch angle tracking errors to a minimum while executing the demanding yaw reference control task. Performance was increased on both the roll and pitch reference tracking tasks, as well reducing the convergence time to the reference heading angle.

The prioritized *WLS CA* showed promising fault tolerant capabilities, by demonstrating a controllable and flyable *MAV* with a severely damaged actuator. The damaged actuator introduces a significantly asymmetric control effectiveness matrix for which full control over the four states (roll, pitch and yaw angle, and total thrust of the *MAV*) is not feasible.

It has been shown that *WLS* optimal control allocation can increase performance for multi-rotor type *MAVs* with typical highly coupled control effectiveness matrices. Prioritizing for the roll and pitch control objective increases the overall performance of the system for highly demanding maneuvers. Combined with prioritized *WLS* control allocation, adaptive *INDI* provides a robust and flexible controller which has the ability to fly with compromised actuators.

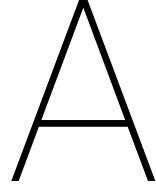
REFERENCES

- [1] B. Erginer and E. Altug, "Modeling and PD Control of a Quadrotor VTOL Vehicle," *2007 IEEE Intelligent Vehicles Symposium*, pp. 894–899, 2007.
- [2] E. J. J. Smeur, "Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Aerial Vehicles," *AIAA Guidance, Navigation, and Control Conference*, pp. 1–16, 2016.
- [3] T. A. Johansen and T. I. Fossen, "Control Allocation - A Survey," *Automatica*, vol. 49, 2012.
- [4] M. Bodson, "Evaluation of Optimization Methods for Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, pp. 703–711, 2002.
- [5] J. C. Virnig and D. S. Bodden, "Multivariable Control Allocation and Control Law Conditioning when Control Effectors Limit," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 572–582, 1994.
- [6] J. M. Buffington and D. F. Enns, "Lyapunov Stability Analysis of Daisy Chain Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 6, 1996.
- [7] J. M. Buffington, "Tailless aircraft control allocation," *AIAA Guidance, Navigation, and Control Conference*, pp. 737–747, 1997.
- [8] W. C. Durham, "Constrained control allocation," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 4, pp. 717–725, Jul. 1993.
- [9] W. C. Durham, "Constrained control allocation - Three-moment problem," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 2, pp. 330–336, Mar. 1994.
- [10] W. C. Durham, "Attainable moments for the constrained control allocation problem," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 6, pp. 1371–1373, Nov. 1994.
- [11] W. C. Durham, "Computationally Efficient Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, 2001.
- [12] K. A. Bordignon and W. C. Durham, "Closed-form solutions to constrained control allocation problem," *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 5, pp. 1000–1007, Sep. 1995.
- [13] O. Härkegård, "Efficient active set algorithms for solving constrained least squares problems in aircraft control allocation," *Decision and Control, 2002, Proceedings of the ...*, vol. 2, no. December, pp. 1295–1300, 2002.

- [14] J. J. Burken, P. Lu, Z. Wu, and C. Bahm, "Two Reconfigurable Flight-Control Design Methods: Robust Servomechanism and Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, pp. 482–493, 2001.
- [15] J. A. M. Petersen and M. Bodson, "Constrained Quadratic Programming for Control Allocation," *IEEE Trans. Control and Dynamics*, vol. 14, no. 1, pp. 91–98, 2006.
- [16] T. Schneider, G. Ducard, K. Rudin, and P. Strupler, "Fault-tolerant Control Allocation for Multirotor Helicopters using Parametric Programming," *International Micro Air Vehicle Conference and Flight Competition*, 2012.
- [17] M. Frangenberg, J. Stephan, and W. Fichter, "Fast Actuator Fault Detection and Reconfiguration for Multicopters," *AIAA Guidance, Navigation, and Control Conference*, no. January, pp. 1–24, 2015.
- [18] M. W. Mueller and R. D'Andrea, "Stability and control of a quadcopter despite the complete loss of one, two, or three propellers," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 45–52, 2014.
- [19] J. Nocedal, S. J. Wright, and S. M. Robinson, *Numerical Optimization*, 1999.
- [20] P. De Monte and B. Lohmann, "Trajectory tracking control for a quadrotor helicopter based on backstepping using a decoupling quaternion parametrization," *21st Mediterranean Conference on Control and Automation*, pp. 507–512, 2013.
- [21] M. V. Kothare, P. J. Campo, M. Morari, and C. N. Nett, "A unified framework for the study of anti-windup designs," *Automatica*, vol. 30, no. 12, pp. 1869–1883, 1994.
- [22] E. Johnson and A. Calise, "Pseudo-control hedging: A new method for adaptive control," *Advances in Navigation Guidance and Control Technology*, vol. September, 2000.



Part 2: Appendices



Implementation in Paparazzi

In this section a quick overview of the files which are used by the *WLS* control allocator, and

The files can be easily accessed by using the Github "*file finder*" on the writer's personal Github branch of Paparazzi: https://github.com/noreplacementfordisplacement/paparazzi/find/bebop_wls.

A.1 Structure and files

The following **C** files are of importance for the *WLS* Control Allocator. It is stated if the file is modified with respect to the Paparazzi Master branch, or a new file which has been generated for this research. A general overview of the system and relevant variables are given in figure A.1. The functionalities of each file are briefly described in section A.2.

- **tudelft_bebop_indiwls.xml** : airframe configuration file (generated for this research)
- **stabilization_indi.c** : (adaptive) INDI attitude control loop file (heavily modified)
- **motor_mixing.c** : file which generates true actuator input (modified)
- **wls_alloc.c** : WLS Active-Set constrained quadratic programming control allocator (non-standard paparazzi file, generated for this research)
 - **qr_solve.c** : external qr matrix solver library
- **file_logger.c** : file logger (modified)

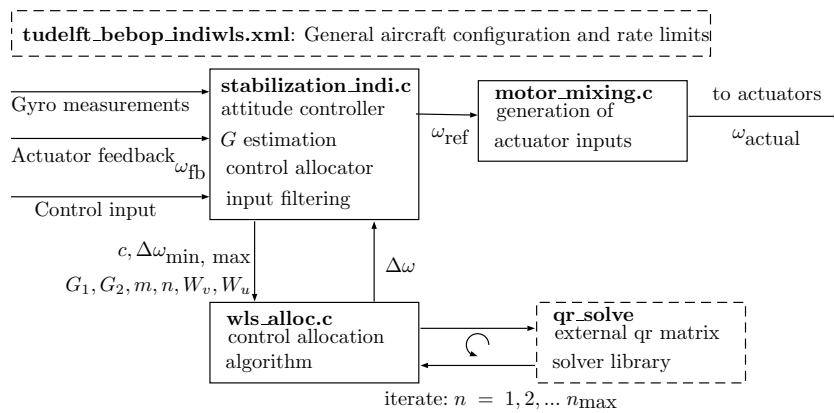


Figure A.1 Block diagram of control allocator implementation

A.2 File description

Here we observe how different files exchange information and summarize the key functionalities.

A.2.1 Airframe file

The airframe file "*tudelft_bebop_indi.wls.xml*" enables configuration of top level functionalities. Important are type of controllers used for the attitude and guidance control loop as well as the sensors and systems (Optitrack, GPS, Magnetometer) specified to generate input to key variables such as the heading and position. The default rate limit for yaw is set by default at a safe 120 [deg/s] and was increased to 600 [deg/s] equivalent to the limits on the roll and pitch axes.

A.2.2 (INDI) Attitude Controller

The "*stabilization_indi.c*" is the main file used in this research. It combines all the elements required for the adaptive *INDI* attitude control loop. Two types of inputs are received: measurements and control input. The measurements include in filtered gyro measurements and actuator feedback. Combined with the control input and the gyro measurements the incorporated *Proportional Derivative* linear controller generates an angular acceleration reference. The measured angular acceleration (differentiated gyro measurements which are filtered) are subtracted and are the control objective c for the *WLS* control allocator. The *WLS* control allocator is called by the **wls_alloc** function. The "*stabilization_indi.c*" also contains the *LMS* adaptive filter to estimate the actuator effectiveness matrices and the filters to filter the gyro and actuator feedback.

A.2.3 WLS Control Allocator

The file "*wls_alloc.c*" executes the pseudo-code from in Algorithm 3 (found in the paper). It is designed such that it can be very easily implemented on other platforms.

The function is called with the following syntax:

wls_alloc(u, v, umin, umax, Bwls, NICO, MARINUS, Wv, Wu, wd, gamma, nmax)

u Initial solution $\Delta\omega_0$ and output of the function (initial solution is overwritten)

v Control objective c

umin Minimum increment vector $\Delta\omega_{\min}$

umax Maximum increment vector $\Delta\omega_{\max}$

Bwls Control effectiveness matrix $G_1 + G_2$

NICO Column n dimension of control effectiveness matrix

MARINUS Row m dimension of control effectiveness matrix

Wv Control objective weighting matrix W_v

gamma Priority of primary control objective $\gamma > 10000$

nmax Maximum number of iterations $n_{\max} = 100$

Not used:

Wu Actuator weighting matrix W_u

wd Preferred solution ω_d

The matrix solver library "*qr_solve*" uses a *QR* decomposition to find the least squares solution to an overdetermined system of equations and is called each iteration by the *WLS* algorithm.

A.2.4 Motor Mixing

The file "*motor_mixing.c*" basically has no real function other than processing the ω_{ref} generated by the "*stabilization_indi.c*" file. A built in override function is used such that direct feed trough of the ω_{ref} command to the actuators is enabled.

A.2.5 File Logger

The Parrot Bebop possesses on-board memory, which can be used to write log files to. The file `"file_logger.c"` generates a ".csv" file. The following variables were logged:

- Measured attitude angles MAV Attitude (η_{MAV}) [deg]
- Reference attitude angles Reference Attitude (η_{ref}) [deg]
- Actuator feedback ω_{fb} [RPM]
- Filtered actuator feedback ω_f [RPM]
- Reference actuator rate ω_{ref} [RPM]
- Reference rotational acceleration (PD output) $\dot{\Omega}_r$ [deg/s²]
- Measured rotational acceleration (differentiated body rate, which has been filtered) $\dot{\Omega}_m$ [deg/s²]
- Incremental control objective ${}_c \Delta \dot{\Omega}$ [deg/s²]
- Incremental allocator output $\Delta \omega$ [RPM]

B

Control Effectiveness Matrices Estimate During the Compromised Actuator Experiment

During the second experiment with the compromised actuator, the control effectiveness matrices needed to be re-identified. This was done using a Least Mean Squares Adaptive Filter (*LMS*) estimator, similar to what was done in [1]. The control effectiveness depends on the moment of inertia of the vehicle and propellers as well as the type of propellers. In this case, the control effectiveness also takes into account the lateral distance to the propellers b and the longitudinal distance l . The *LMS* estimator works by observing the error between the achieved incremental angular acceleration and the angular acceleration which is expected from the incremental angular rate of the actuators.

For convenience the *LMS* equations are displayed below in equations B.1 and B.2. The element μ_1 is a diagonal, constant matrix with adaptation constants. The diagonal matrix μ_2 is also constant but the values differ for each control axis, due to different signal to noise ratios per axis. In the equations $\Delta\omega$ denotes the incremental output of the Incremental Non-Linear Dynamic Inversion (*INDI*) controller to the actuators and $\Delta\dot{\omega}$ the incremental actuator rate acceleration used to model the spin-up torque of the actuators. The achieved incremental acceleration is displayed with $\Delta\dot{\Omega}$. All time-variant values are in the same time step.

$$G(k) = G(k-1) \mu_2 \left(G(k-1) \begin{bmatrix} \Delta\omega \\ \Delta\dot{\omega} \end{bmatrix} - \Delta\dot{\Omega} \right) \begin{bmatrix} \Delta\omega \\ \Delta\dot{\omega} \end{bmatrix}^T \mu_1 \quad (\text{B.1})$$

$$G = [G_1 \ G_2] \quad (\text{B.2})$$

The adaptation of the control effectiveness matrices is displayed in figures B.1a to B.2b. Note that the adaptation is only made for 2[s] interval and the values at the start are already very asymmetrical. This was done performed during an earlier test flight, which was not logged. The Micro Aerial Vehicle (*MAV*) was controlled manually in attitude flight mode. This was done in a obstructed CyberZoo, such that the pilot was only able to fly for approx 1 second. In this second we see that the *LMS* adaption works, and converges to new values.

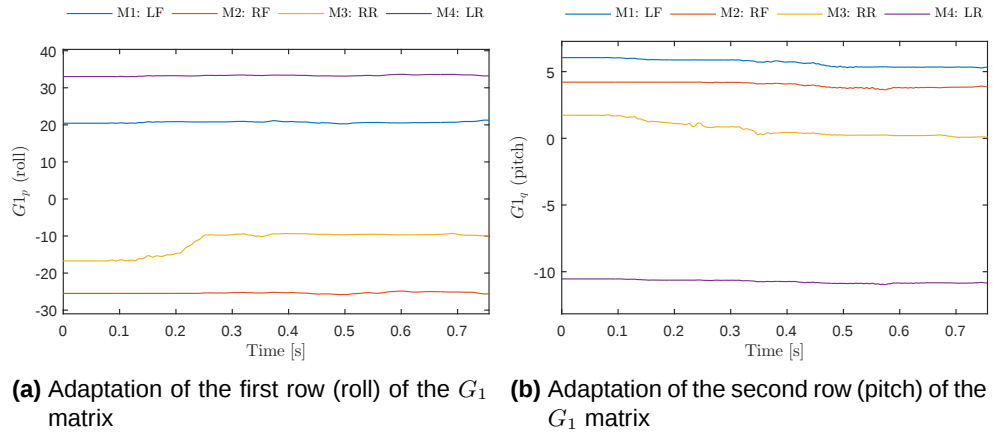


Figure B.1 Roll and pitch Control in G_1

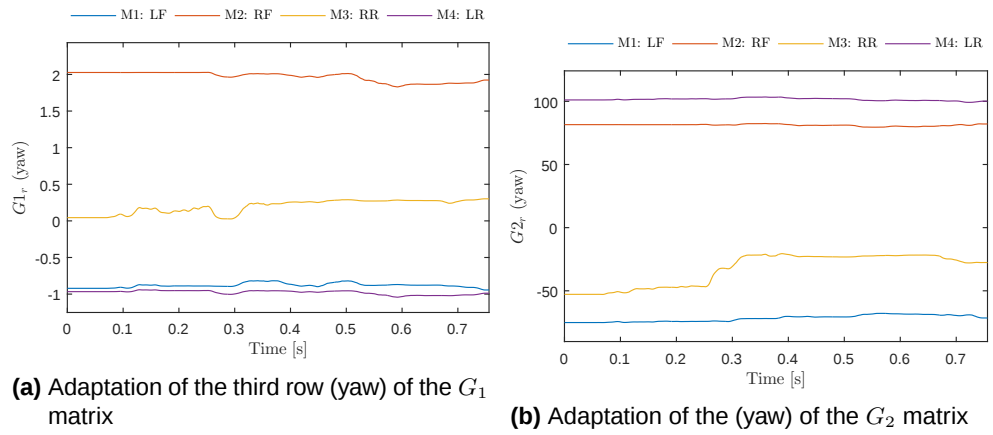
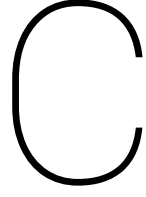


Figure B.2 Yaw in G_1 and G_2



Applied Control Allocation Algorithms

From available constrained control allocators we are interested in the Active-set based Quadratic Programming (*QP*) methods and direct control allocation. In this section, a derivation is made to apply these to the quad-copter case and implemented in the *INDI* control architecture.

C.1 Control Allocation in the INDI Controller

For the attitude *INDI* controller, the commands issued are incremental angular accelerations on all three axes: $\Delta\dot{\Omega}_p, \Delta\dot{\Omega}_q, \Delta\dot{\Omega}_r$. Multiplied with the (pseudo)-inverse of the Control Effectiveness Matrix (*G*), the incremental acceleration command yields the incremental angular velocity command for the actuators.

In the case of a quad-copter, there are four actuators. These actuators define fully control the attitude, and accelerations of the vehicle. With the *INDI* controller the angular accelerations and the acceleration in the z body axis are controlled. The limits on the actuator follow from the maximum incremental angular velocity increase and decrease which can be achieved. This is obtained by subtracting the angular velocity feedback from the actuator limits for the positive increment limit. The negative increment limit simply is the negative amount of the actuator angular velocity feedback. The assumption is made that the actuators are practically infinitely fast, such that there is no rate limit on the actuators. This can be identified later in the research.

$$\Delta\omega_{max} = \omega_{max} - \omega_{fb} \quad (C.1)$$

$$\Delta\omega_{min} = -\omega_{fb} \quad (C.2)$$

Actuator limits

The actuator limits are defined by the maximum and minimum angular velocity of the motors. This limit can be variable, depending on flight conditions, external disturbances and propellers used. [2]. It is assumed that there is a fixed, maximum angular velocity for the actuators.

C.2 Direct Control Allocation

Two separate branches of Direct Control Allocation are identified: Geometric Direct Control Allocation (*gDCA*) and Simplex Direct Control Allocation (*linDCA*). The implementation of the *linDCA* is much less complex.

C.2.1 Geometric DCA

First the attainable control set (all the attainable control-objectives) is derived for the attitude (achievable angular acceleration increments (in R^3)).

A m dimensional control space is considered with actuators $\delta\omega \in R^m$. The subset defines the maximum and minimum achievable values of the actuators:

$$\Theta = \{\Delta\omega \in R^m \mid \Delta\omega_{i,min} \leq \Delta\omega_i \leq \Delta\omega_{i,max}\} \subset R^m \quad (C.3)$$

The subset of attainable values for the actuators can be mapped to the attainable control space in R^n , where $m > n$. This mapping H follows from equation C.4. Using the linear mapping, as displayed in equation C.5, the points which span a convex hull $d(\Phi) \in R^n$ are defined.

$$H : R^m \rightarrow R^n \quad (C.4)$$

$$H\Delta\omega = \Phi \quad \text{where: } H = (G_1 + G_2) D \quad (C.5)$$

$$(G_1 + G_2) D \Delta\omega = \Phi \quad (C.6)$$

The mapping D is an important to $gDCA$. The matrix consists of all possible combinations of minimum and maximum increments.

Each of the actuators actively controls the 3 attitude angular accelerations as well as the translational acceleration in the MAV's z axis. The coupling of the actuators is displayed in figure C.1. A plus sign denotes a positive increment in angular velocity of the propeller is needed for a positive control objective. A negative sign means an angular deceleration is required to achieve positive control objective.

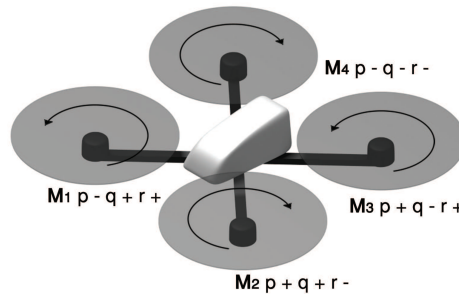


Figure C.1 Coupling of the actuators

Since each control objective affects all four of the actuators, the mapping D consists of combinations of the actuator limits, of the four actuators. With a correct mapping D , the full convex hull $d(\Phi)$ which spans the attainable control set can be defined. For these combinations of actuator limits the sum has to be equal to a value proportional to the acceleration command in the body's z axis, as shown in equation C.7.

The actuator constraints depends on the current command for the acceleration in the z axis. It can be the case that the minimum possible increment $\Delta\omega_{min}$ is a more limiting factor for attitude control than the maximum increment $\Delta\omega_{max}$. This is displayed in figures C.2a and C.2b. The current acceleration in the z axis is shown as the "Current Hover Position".

$$c_{a_{z,b}} = \sum_{i=1}^m C_{z_{d\omega}} \Delta\omega_i \quad (C.7)$$

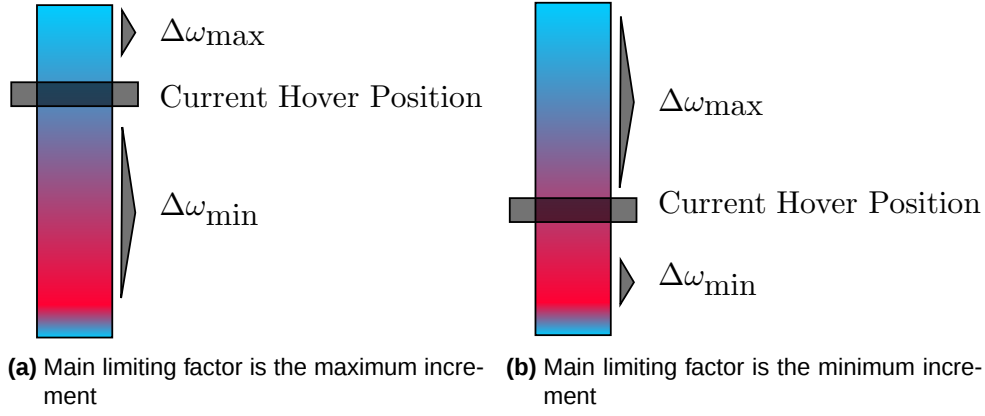


Figure C.2

C.2.2 Attainable Control Objective set in R^3

The combinations of actuator increments D in R^4 can be mapped to the attainable control commands in R^3 . This is achieved by multiplying D with the control effectiveness matrix $(G_1 + G_2)$ as shown in equation C.6. The convex set of these points yield 4 distinct points in the attainable control space R^3 . In this space the x , y and z represent $\Delta\dot{\Omega}_p$, $\Delta\dot{\Omega}_q$ and $\Delta\dot{\Omega}_r$ respectively.

A symmetric attainable control set is displayed in figure C.3. The 4 vectors from Φ form a figure known as a tetrahedron. It contains 4 vertices, 6 edges and 4 faces. Note that the axis scaling is not equal, such that the figure appears to be less "flat" than it actually is.

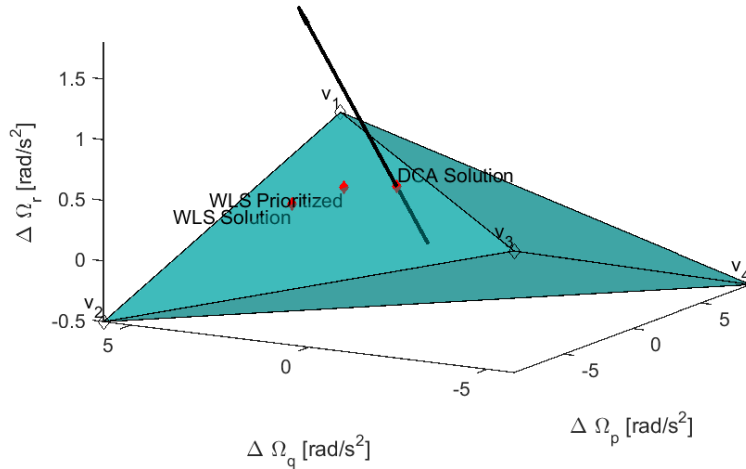


Figure C.3 Attainable Control Set Visualization, with multiple solutions

For this application with 4 actuators, the control set is fully defined by the 4 vectors in Φ . By connecting these points in the 3 dimensional space, the convex hull $d(\Phi)$ is obtained.

C.2.3 Geometric DCA Solution

With $gDCA$, the solution is defined as the intersection with the control objective vector c and the convex hull $d(\Phi)$. For this research specifically, a new and efficient $gDCA$ algorithm was established, which is displayed in algorithm 1. One of the inputs for the algorithm is a triangulation of the vectors present in the convex hull $d(\Phi)$. In the case of a Quad-Copter, this can be done manually, since for each problem, the set of vectors which establish the triangulation faces do not interchange. Their magnitudes, and direction however, do change.

If the Cyrus-Beck test shows a positive value, it is determined the command is unattainable, and

Algorithm 1: Implemented Geometric Direct Control Allocation

for triangulations $1, \dots, n$ **do**

Project the vectors of the current triangulation v_i ($i = 1, 2, 3$) onto the plane, which is normal to the control objective c . This results in the projected triangulation vectors w_i .

$$w_i = v_i - \hat{v}_i \quad \text{with: } \hat{v}_i = \frac{v_i \cdot c}{c \cdot c} c \quad (C.8)$$

Now check if the vectors w_i enclose the command vector c as depicted in figure C.4. This can be done by solving the following system:

$$\begin{bmatrix} w_{i,2} - w_{i,1} & w_{i,3} - w_{i,1} & -w_{i,1} \end{bmatrix} = 0 \quad (C.9)$$

The solution of the system in equation C.9 yield the barycentric coordinates s, t .

if $0 \leq s, t \leq 1$ **then**

The vector c is enclosed by the active triangulation. Now there is a need to determine if the vector is inside or outside the current triangulation face. This can be done by first determining the outside facing normal vector n_f .

$$v_{i,1} = v_2 - v_1 \quad \text{and: } v_{i,2} = v_3 - v_1 \quad (C.10)$$

$$n_f = \frac{w_{i,1} \times v_{i,2}}{\sqrt{(v_{i,1} \cdot v_{i,1}) (v_{i,2} \cdot v_{i,2}) - (v_{i,1} \cdot v_{i,2})^2}} \quad (C.11)$$

Now the Cyrus-Beck test can be applied:

$$T_{CB} = (c - v_1) \cdot n_f \quad \text{with: } T_{CB} > 0 \text{ if } c \text{ is outside of face } f_i \quad (C.12)$$

if $T_{CB} \geq 0$ **then**

The solution can be accepted.

$$c_s = v_{i,1} + s(v_{i,2} - v_{i,1}) + t(v_{i,3} - v_{i,1}) \quad \text{with: } 0 \leq t, s \leq 1 \quad (C.13)$$

else

This is not the solution, move to the next triangulation.

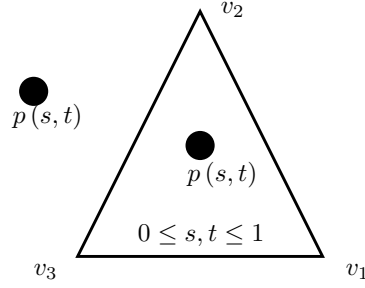


Figure C.4 Two dimensional intersection

there is a need to modify the command. The solution of the intersection of the control vector with its intersecting face is the maximum attainable control objective c_s in the direction of c . This conveniently is the same solution as found in C.13 for the vectors tangent to the normal plane of the control vector.

With c_s established, the actuator command $\Delta\omega_s$ from equation C.14 of the *gDCA* Control Allocator (CA) can be obtained using the indices i of the solution triangulation, the combination $\Delta\omega_{\min}, \Delta\omega_{\max}$ of matrix D corresponding to i ($D(i)$) and the barycentric coordinates s, t . Remember that D determines the mapping of incremental actuator commands which determine Φ .

$$\Delta\omega_s = s D(i) + t D(i) \quad (\text{C.14})$$

C.3 DCA: Linear Programming

Arguably less efficient than geometric control allocation, [3] introduced a linear programming formulation of Direct Control Allocation (*DCA*). It is a much more flexible application of *DCA* for a multitude of cases.

Algorithm 2: Linear Programming Formulation of Direct Control Allocation

The control allocation problem is reformulated in a linear program format:

$$\min_{\omega, \epsilon} J = f(\omega) - \epsilon \quad \text{subject to:} \quad (\text{C.15a})$$

$$(G_1 + G_2) \omega = c \quad (\text{C.15b})$$

$$0 \leq \epsilon \leq 1 \quad (\text{C.15c})$$

$$\omega_{\min_i} \leq \omega_i \leq \omega_{\max_i} \quad \text{with: } i = 1, 2, 3, 4 \quad (\text{C.15d})$$

There are no equality constraints present, and the cost function J is minimized with the variables ω and ϵ left to vary. With the linear programming formulation, the variable ϵ scales the total control objective c .

For this linear programming problem, many solving algorithms can be used. Matlab package contains a default tool to solve linear programming solver, which enables selecting the default simplex solver.

C.3.1 Sequential Linear Programming

In [3] a new linear programming method enables sequential realization of the control objective. This entails that an order of realization is assigned to control objective axes. The axis with the highest priority is satisfied first, until its respective $\lambda_i = 1$. Then the algorithm (shown in algorithm 3) sets out

to realize the consecutive axis.

Algorithm 3: Sequential Linear Direct Control Allocation

$$(G_1 + G_2) \omega = \Lambda \mathbf{c}$$

with: $\Lambda = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$ **and:** $\Delta\omega_{\min} \leq \Delta\omega \leq \Delta\omega_{\max}$, $0 \leq \lambda_i \leq 1$

if $\lambda_i = 1$: **then**

$$\left| \lambda_{i-1} = \lambda_{i-2} = \dots = 1 \right.$$

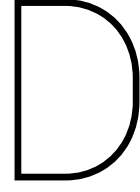
if $\lambda_i = 0$: **then**

$$\left| \lambda_{i+1} = \lambda_{i+2} = \dots = 0 \right.$$

if $0 \leq \lambda_i \leq 1$: **then**

$$\left| \begin{cases} \lambda_{i-1} = \lambda_{i-2} = \dots = 1 \\ \lambda_{i+1} = \lambda_{i+2} = \dots = 0 \end{cases} \right.$$

With objective: $J = f(\omega) - \sum_{i=1}^3 \lambda_i$



Quadratic Programming using the Active-Set solver

In this section the *WLS* and *SLS* active-set *QP* Control Allocation algorithms are implemented for the application of this research. Both follow from the work of [4], where the active-set *QP* is applied to constraint control allocation. The underlying principle of the active-set method, is that active in-equality constraints are selected in the working set W , and subsequently regarded as equality constraints.

In *QP* methods the l_2 norm is used as displayed in equation D. Again the weighting matrices W_v and W_u are introduced, which weigh the control objective realization and actuator deflections respectively. Also the preferred solution ω_d can "steer" the actuator commands which provide the solution.

P-norm notation: With $p \geq 1$ a real number:

$$\|x\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}} \quad (\text{D.1a})$$

With $p = \infty$

$$\|x\|_p = \max(|x_1|, \dots, |x_n|) \quad (\text{D.1b})$$

D.0.1 Visualization of QP solution

The *QP* optimization method can be visualized geometrically by plotting a Z value on the solution facet on the attainable control set, which is shown in figure D.1. The point with the lowest value in the *QP* solution is visualized by the point with the maximum Z value. This corresponds to the highest value of the function displayed in D.2. Note that the cost function works with the barycentric coordinates s and t from the solution facet of the *gDCA CA* from section C.2.3. For convenience of notation, the subscript of the values is simplified. For example, the weight on incremental roll acceleration $w_{\Delta \dot{\Omega}_p}$ is denoted simply as: w_p . For $0 \leq s, t \leq 1$ the combination of s, t can be found which realizes the maximum value of Z . This can be achieved numerically, by computing Z for an array of s, t values, and subsequently selecting the s, t which correspond to the maximum computed value for Z .

$$\begin{aligned} Z = & -1 \left(w_p \left(c_p - (v_{1_p} + s(v_{2_p} - v_{1_p}) + t(v_{3_p} - v_{1_p})) \right) \right)^2 + \dots \\ & w_q \left(c_q - (v_{1_q} + s(v_{2_q} - v_{1_q}) + t(v_{3_q} - v_{1_q})) \right)^2 + \dots \\ & w_r \left(c_r - (v_{1_r} + s(v_{2_r} - v_{1_r}) + t(v_{3_r} - v_{1_r})) \right)^2 \Big)^{\frac{1}{2}} \quad (\text{D.2}) \end{aligned}$$

D.1 Sequential Least Squares (SLS)

The sequential least squares solution consists of two phases. First, a feasible solution for the actuator commands is found. Consequentially, this solution may be optimized based on actuator preference

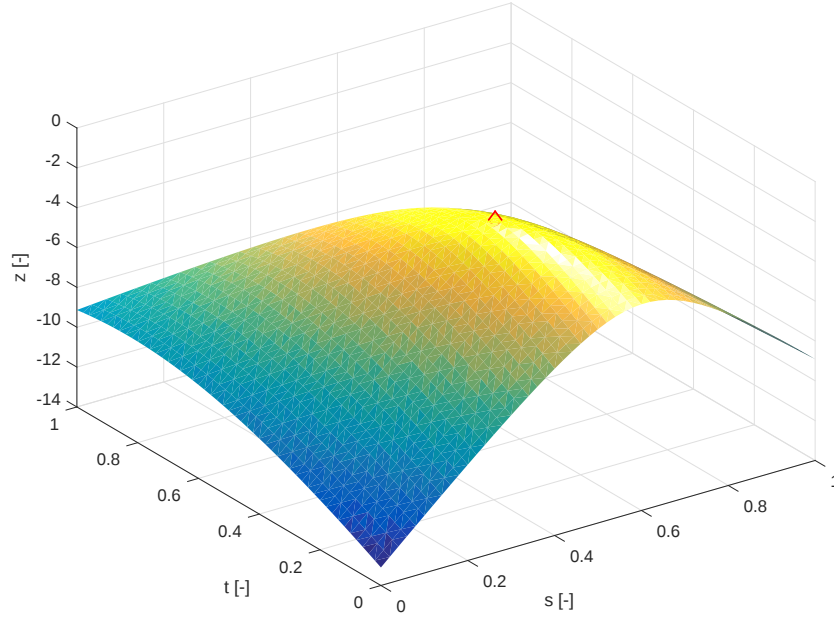


Figure D.1 Visualization of the QP solution, using the solution facet on an Attainable Control Set

and actuator weighting matrix. Two active-set optimizations are thus applied. Note the line in equation D.3c stating: $p_j = 0, \quad j \in W$. This entails that the actuator associated to j has already been saturated.

Phase 1: Realized Control Objective

With working set W and the solution ω^0 from the previous time-step
Iteratively solve

$$\omega_{\Theta} = \arg \min_{\omega} \|W_v ((G_1 + G_2) (\omega^i + p) - c)\| \quad (\text{D.3a})$$

$$\omega = \omega^i + p \quad \omega_{\min} \leq \omega \leq \omega_{\max} \quad (\text{D.3b})$$

$$p_j = 0, \quad j \in W \quad (\text{D.3c})$$

The step p is computed such that it minimizes the difference between the solution c_s and the control objective c . The computation follows from:

$$d = W_v (c - (G_1 + G_2) \omega^i) \quad (\text{D.4})$$

For convenience the weighted control effectiveness matrix is introduced: $A = W_v (G_1 + G_2)$. When iterating, the active columns in the control effectiveness matrix are defined by the working set:

$$A_{\text{active}} = A(:, h), \quad h \notin W \quad (\text{D.5})$$

With the active control effectiveness method it allows computation of p . Note that A^+ denotes the pseudo-inverse of A .

$$p(h) = A_{\text{active}}^+ d \quad p(j) = 0 \quad \text{with: } h \notin W, \quad j \in W \quad (\text{D.6})$$

Algorithm 4: Implemented SLS Phase 1

if $\omega^i + p$ is feasible **then**

$\omega^{i+1} = \omega^i + p$ In the next step the Lagrange multipliers are calculated. First the direction (or gradient) has to be computed with a new residual term:

$$d = W_v (c - (G_1 + G_2) \omega^{i+1}) \quad (D.7)$$

Now the gradient can be obtained with: $\nabla = A^T (A \omega^{i+1} - W_v c)$. This can be combined with the expression introduced in equation D.7 and the expression for A ($A = W_v (G_1 + G_2)$).

The gradient now can simply be found with:

$$\nabla = -A^T d \quad \text{LaGrange Multipliers: } \lambda = -W \nabla \quad (D.8)$$

f if all $\lambda \geq 0$ **then**

The solution ω^{i+1} is optimal. Now $\omega_\Theta = \omega^{i+1}$;

else

The constraint associated with the most negative λ has to be removed from the working set. Re-iterate with this working set.;

else

The solution moves along constraints towards an optimum. The proposed solution might violate a constraint which is not in the working set, or not be on an extremum of the current set of constraints. Now determine the factor α such that $\omega^{i+1} = \omega^i + \alpha p$ is feasible. If applicable, the constraint active at ω^{i+1} has to be added to the active set.

The algorithm stops when $(G_1 + G_2) \omega = c$, a maximum number of iterations $i_{\max}$ has been reached, or all $\lambda > 0$. Now Phase 2 can be commenced.

Phase 2: Actuator Solution Now $\omega_0 = \omega_\Theta$ and the working set W is the working set from phase

1.

Second optimization problem:

$$\omega = \arg \min_{\omega} \|W_u ((\omega^i + p) - \omega_d)\| \quad (D.9a)$$

$$\omega = \omega^i + p \quad (G_1 + G_2) p = 0 \quad (D.9b)$$

Now there are more equality constraints such that the solution ω_Θ from phase 1 remains satisfied. Also the constraints from the working set are placed in a new matrix C_0 .

$$C_0(i, j) = W(j) \quad \text{where } i \text{ is the index of the constraint and: } j \in W \quad (D.10)$$

$$C_0(i, j) = 0 \quad \text{where } i \text{ is not the index of the constraint} \quad (D.11)$$

The equality constraints $(G_1 + G_2)p = 0$ are added:

$$C_0(i, j) = W(j) \quad \text{where } i \text{ is the index of the constraint and: } j \in W \quad (D.12)$$

$$C_0(i, j) = 0 \quad \text{where } i \text{ is not the index of the constraint} \quad (D.13)$$

All the equality constraints are summarized in the matrix E_p :

$$E_p = \begin{bmatrix} (G_1 + G_2) \\ C_0 \end{bmatrix} p = 0 \quad (D.14)$$

A QR decomposition of E_p yields orthogonal matrix Q and the upper triangular matrix R .

$$E^T = QR \quad (D.15)$$

The number of control axes in c plus the number of equality constraints from the working set is the number k_c and allows separation of the matrices Q and R .

$$Q_1 = (:, 1 : k_c) \quad (D.16)$$

$$Q_2 = (:, (k_c + 1) \dots) \quad (D.17)$$

$$R_1 = \text{rows}_R(1 : k_c) \quad (D.18)$$

The direction of the optimization is determined by the residual:

$$d_2 = W_u (\omega_d - \omega) \quad (D.19)$$

The perturbation can be computed with the intermediate vector q_2 which is the solution from the linear system $W_u Q_2 q_2 = d_2$. Now the perturbation is given by:

$$p = Q_2 q_2 \quad (D.20)$$

Again, the system iterates according to the active-set method:

Algorithm 5: Implemented SLS Phase 2

if $\omega^i + p$ *is feasible*: **then**

$\omega^{i+1} = \omega^i + p$ and the Lagrange multipliers λ can be computed. This is permed similarly as in equation D.8, now with $A = W_u$ and $d = W_u (\omega_d - \omega^{i+1})$. The orthogonal decomposition and the gradient are subsequently used to find the Lagrange multipliers:

$$\lambda = R_1 (Q_1^T g). \quad (D.21)$$

if all $\lambda \geq 0$ **then**

The solution ω^{i+1} , now set $\omega = \omega^{i+1}$.

else

Vary the active constraints in the working set associated with the most negative λ by changing the active constraints and re-iterate.

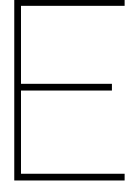
else

Determine the maximum step length α such that $\omega^{i+1} = \omega^i + \alpha p$ is feasible. The constraints need to be re-evaluated.

D.1.1 Weighted Least Squares Method

The WLS method combines both steps from the SLS method in one step similar to phase 1 of the SLS method. This is achieved by combining all the constraints in one equation shown below in equation D.22. The main difference is the introduction of the weighting factor γ which generally is chosen to be in the order of 10^2 to 10^4 . This weighting factor ensures that the solution of the control allocation problem enjoys priority.

$$\|W_u (\omega - \omega_d)\|^2 + \gamma \|W_v ((G_1 + G_2) \omega - \mathbf{c})\|^2 = \left\| \begin{pmatrix} \gamma^{\frac{1}{2}} W_v (G_1 + G_2) \\ W_u \end{pmatrix} \omega - \begin{pmatrix} \gamma^{\frac{1}{2}} W_v \mathbf{c} \\ W_u \omega_d \end{pmatrix} \right\|^2 \quad (D.22)$$



Setup of Experiment using a Dynamic Model

Errata: The simulation of the control allocators with the dynamic model took place prior to the physical experiments. A small modeling error, was made: the sampling time T_s in the simulation has been chosen at 1/500 [s] whereas the used control effectiveness matrices are taken from logged data which for which T_s was 1/512 [s]. The simulated quadcopter now has a comparatively higher control effectiveness and has slightly more actuation capabilities.

In this chapter, we introduce a experiment where the performance of different control allocators is assessed using a dynamical model. Central to this experiment is a model generated to simulate the dynamics of the *INDI* controlled quad-copter. This model is briefly introduced in section E.1. The control allocators are implemented in the model, and a set of control objectives are introduced. This enables us to show the differences in performance and limitations which different control allocators yield. This experiment will provide qualitative performance data. This data, subsequently is analyzed with a set of performance metrics. With the performance metrics, suitable information is obtained to draw conclusions for continuation of this research and implementation in Paparazzi.

E.1 Dynamic Model

The objective of the dynamic model is to provide a limited fidelity approximation of the dynamics of the MAV. In this model, the control allocators can be implemented to show the differences in response and control allocation.

The dynamic model is established based on the dynamic model layed-out in [1]. Here the incremental dynamic inversion controller is derived, and subsequently also an incremental formulation of the dynamics of the quad-copter. This is displayed in equation E.1.

$$\Delta\dot{\Omega} = G_1\Delta\omega + G_2(\Delta\omega - z^{-1}\Delta\omega) \quad (\text{E.1})$$

This incremental angular acceleration $\Delta\dot{\Omega}$ is then subsequently summed to obtain $\Delta\Omega$. The angular rate and angle in the body-frame can be obtained by integrating the angular acceleration Ω over time twice. The G matrix used, has been previously identified in experiments for the Parrot Bebop consumer drone.

A stabilizing angular acceleration reference is needed to provide input to the *INDI* controller. A Proportional Derivative (*PD*) Controller is implemented to provide this, and its gains adjusted which provide satisfactory response.

In an ideal environment, the *INDI* controlled MAV is fully dependent on the actuator dynamics. The actuator dynamics are modeled as a first order filter, with α set at 52.68. The observed dynamics in an ideal environment resemble that of the actuator dynamics controlled with the *PD* controller.

$$H_{\text{actuators}} = \frac{\alpha}{s + \alpha} \quad (\text{E.2})$$

An overview of model architecture is displayed in figure E.1. The outer loop with the linear controller is shown in figure E.2. The contents in the *MAV* block in figure E.1 are displayed in figure E.3

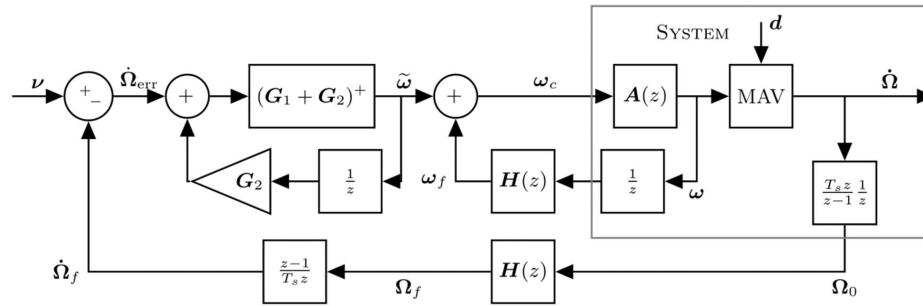


Figure E.1 Overview of the model architecture

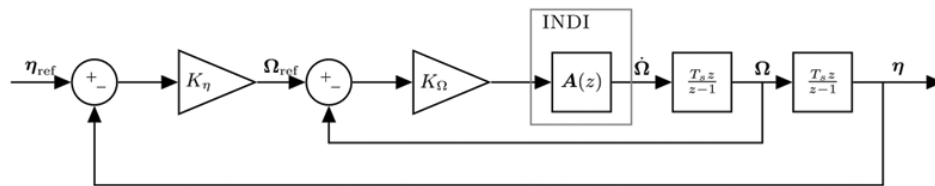


Figure E.2 Linear Controller, with feedback gains on the angular rates and attitude angle

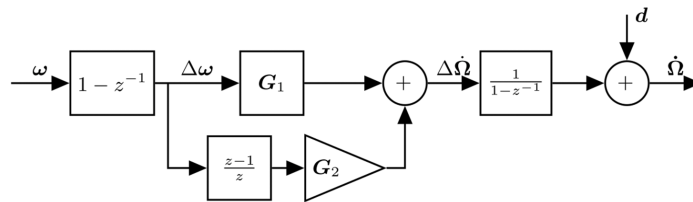


Figure E.3 Contents of the MAV block

The model only provides a low fidelity approximation of the attitude dynamics of the quad-copter. Lateral accelerations are not taken into account. This also means the total height of the modeled MAV. This entails that the sum of all Actuator Angular Velocity (ω), for all t has to be 0.

E.1.1 Assumptions

Several important assumptions in the model are made to make the model more simple and focus on the control allocation aspect.

- The model only models the attitude dynamics. As such only rotations, rates and angular accelerations are modeled.
- There is noise-less angular acceleration feedback, as well as ideal actuator feedback
- There are no aerodynamic disturbances
- The *INDI* model, models the *MAV* perfectly except for a discrepancy in the actuator effectiveness matrix G_1
- The assumptions which simplify the model of the *MAV* introduced in [1] are also valid in this model

E.1.2 Model Discrepancy

To introduce some modeling inaccuracies, the matrix G_1 and G_2 used by the INDI controller are different from those of the modeled dynamics.

The G_1 matrix contains the inverse of the vehicle's inertia I_v^{-1} matrix multiplied by the lateral distance b and longitudinal distance l to the actuators. This also contains the force and moment constant of the rotors, k_1 and k_2 [1].

$$G_1 = 2I_v^{-1} \begin{bmatrix} -bk_1 & bk_1 & bk_1 & -bk_1 \\ lk_1 & lk_1 & -lk_1 & -lk_1 \\ k_2 & -k_2 & k_2 & -k_2 \end{bmatrix} \quad (\text{E.3})$$

The G_2 matrix also contains the inertia of the rotors $I_{r_{zz}}$ around the z axis. Other inertia terms are neglected. The sampling time is denoted as T_s .

$$G_2 = T_s^{-1} I_v^{-1} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ I_{r_{zz}} & -I_{r_{zz}} & I_{r_{zz}} & -I_{r_{zz}} \end{bmatrix} \quad (\text{E.4})$$

The error of the G_1 and G_2 INDI matrices with respect to the G_1 and G_2 matrices of the actual dynamics is 3.45 % and 2.63 % respectively. The matrices are displayed below:

$$G_{1,\text{Model}} = \frac{1}{1000} \begin{bmatrix} -21.5198 & 22.3571 & 22.3314 & -20.9631 \\ 15.3528 & 15.1991 & -14.3894 & -14.0796 \\ 0.7517 & -0.9730 & 1.4182 & -1.2538 \end{bmatrix} \quad (\text{E.5})$$

$$G_{1,\text{INDI}} = \frac{1}{1000} \begin{bmatrix} -21.5198 & 21.5198 & 21.5198 & -21.5198 \\ 14.3894 & 14.3894 & -14.3894 & -14.3894 \\ 1.2538 & -1.2538 & 1.2538 & -1.2538 \end{bmatrix} \quad (\text{E.6})$$

$$G_{2,\text{Model}} = \frac{1}{1000} \begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 0.0758 & -0.0793 & 0.0761 & -0.0803 \end{bmatrix}, \quad (\text{E.7})$$

$$G_{2,\text{INDI}} = \frac{1}{1000} \begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 0.0758 & -0.0758 & 0.0758 & -0.0758 \end{bmatrix} \quad (\text{E.8})$$

E.2 Performance Metrics

To quantify the performance of the control allocators, several performance metrics are introduced. These are used to analyze the time-series data obtained from the simulations. Whereas it is important that the CA fulfills the Control Objective (c) (a Incremental Angular Acceleration ($\Delta\dot{\Omega}$) input), the purpose of the CA is to provide higher performance is satisfying the reference input which the MAV has to track. The following performance metrics are used to asses the performance of the CA methods.

E.2.1 Error Vector

The error vector can be obtained by subtracting the achieved η_{MAV} from the η_{ref} . This error is **not** weighted according to the weighting matrix Control Objective Weighting Matrix (W_v) used by the SLS and WLS algorithms. The magnitude of the total error of the control objectives yields the error. The average absolute error over the n samples is then used as the ϵ_η performance metric.

$$\epsilon_\eta = \frac{\sum_{i=1}^n |(\eta_{\text{ref}} - \eta_{\text{MAV}})|}{n} \quad (\text{E.9})$$

E.2.2 Weighted Error Vector

The obtained error vector for each sample is weighted using the weighting matrix W_v which penalizes error in pitch and roll.

$$\epsilon_{\eta W} = \frac{\sum_{i=1}^n W_v |(\eta_{\text{ref}} - \eta_{\text{MAV}})|}{n} \quad (\text{E.10})$$

E.2.3 Magnitude Error

The magnitude error follows from the division of the magnitude of the η_{MAV} vector by the magnitude of the η_{ref} attitude vector. The average over the total amount of samples is taken. The magnitude error is computed as such:

$$\epsilon_M = \frac{\sum_{i=1}^n \frac{\|\eta_{MAV}\|}{\|\eta_{ref}\|}}{n} \quad (E.11)$$

Computational Effort

In the computational the total computation time to execute the control allocation in the experiment is observed. This is done with the `tic` and `toc` functions in the Matlab environment. It has to be noted that this is just an indication of the time, as the experiment is executed in Matlab, and uses built in Matlab functions. These are much less efficient than comparable C based script.

E.3 Experiment Set-Up

To asses the control allocators, a qualitative experiment is executed. In this qualitative experiment, several reference angle commands are given. Two different scenarios are introduced which will lead to different responses of the control allocators:

- Doublet input on the pitch axis (+45 [deg] and -45 [deg])
- Step input on the yaw axis (90 [deg])

As mentioned earlier, there is a discrepancy in the *INDI* controller's actuator effectiveness matrices and that of the actual model. This discrepancy will lead to coupling of states for a given single axis command. For example, if a pitch step input is issued, the actuator commands send out by the control allocator will lead to a small yaw and roll angular acceleration. This is experienced as a disturbance by the linear controller, and therefore will send correcting roll and yaw commands to the control allocator. The discrepancy is introduced on purpose to introduce the coupling of states.

For the experiments the following constraints were applied to the actuators:

- Maximum angular velocity: $\omega_{max} = 450$ [rad/s]
- Minimum angular velocity: $\omega_{min} = -450$ [rad/s]

It is assumed the vehicle hovers at approximately half power and as such, approximately +450 and -450 [rad/s] are available for each actuator to maneuver. The thrust and rotational speed diagram is displayed in figure E.4.

E.3.1 First experiment: Doublet Input on the Pitch Axis

In the first experiment a doublet input is applied to the pitch axis. The duration of each of the doublet block inputs is 0.5 seconds, and the amplitude 45 degrees. The inputs are applied at 0.1 and 0.6 seconds on the pitch axis. The user reference input is displayed in figure E.5.

In this experiment, we observe the constrained control allocation properties of the control allocators for the doublet pitch input. The pitch and roll angle are controlled effectively by the MAV. It is expected not much difference between the control allocators is visible.

E.3.2 Second experiment: Yaw step input

In the second experiment a step input is applied to the yaw axis at $t = 0.1$ seconds. The amplitude of the step input is 90 degrees. This is displayed in figure E.6.

In this experiment, the control allocation problem will become more apparent. Since the control effectiveness for yaw maneuvers is much less than that of pitch and roll, a large actuator load is almost instantly required. Due to the mismatch in actuator effectiveness matrices G_1 and G_2 , a pitching and rolling motion will be generated upon saturation. How the control allocator deals with this coupling effect is central to this research.

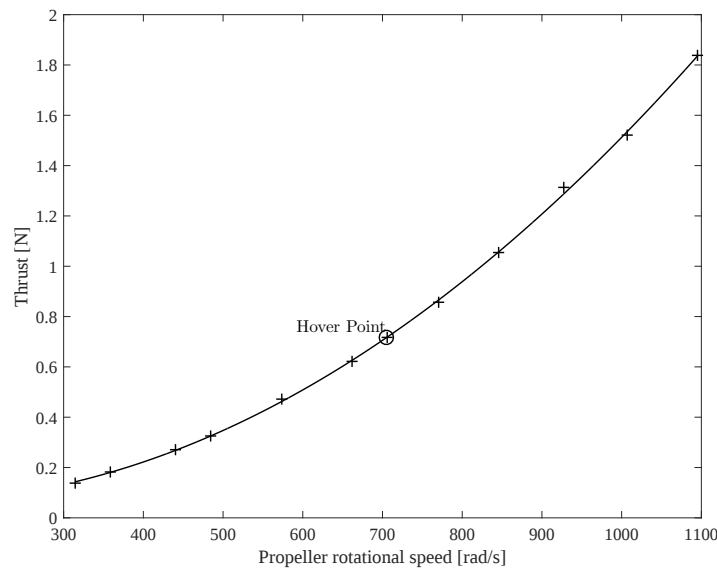


Figure E.4 Thrust Curve of Simulated Actuator

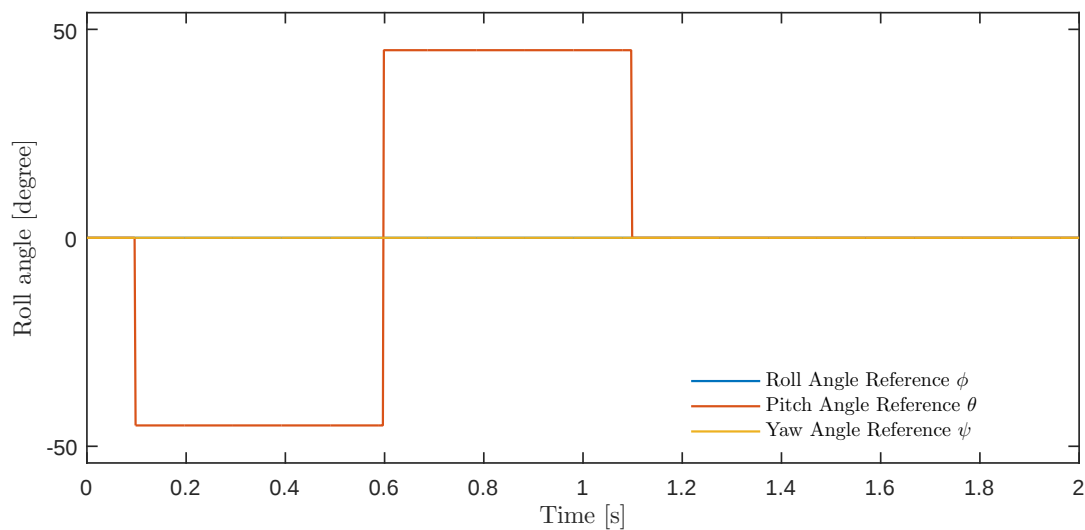


Figure E.5 Reference input to linear controller, pitch doublet input

E.3.3 Control Allocators

The CA algorithms which will be tested are summarized below:

- (RPI uc) Unconstrained Pseudo Inverse (without actuator constraints)
- (RPI) Pseudo Inverse
- (WLS) Weighted Least Squares Optimization
- (WLS pq) Weighted Least Squares Optimization, prioritized for the roll and pitch control objective
- (SLS) Sequential Least Squares Optimization
- (gDCA) Geometric Direct Control Allocation
- (lin DCA) Simplex Direct Control Allocation

Note the inclusion of the **Unconstrained Pseudo Inverse**. This is to have a benchmark for optimal performance, in an ideal world where the actuators have no limits.

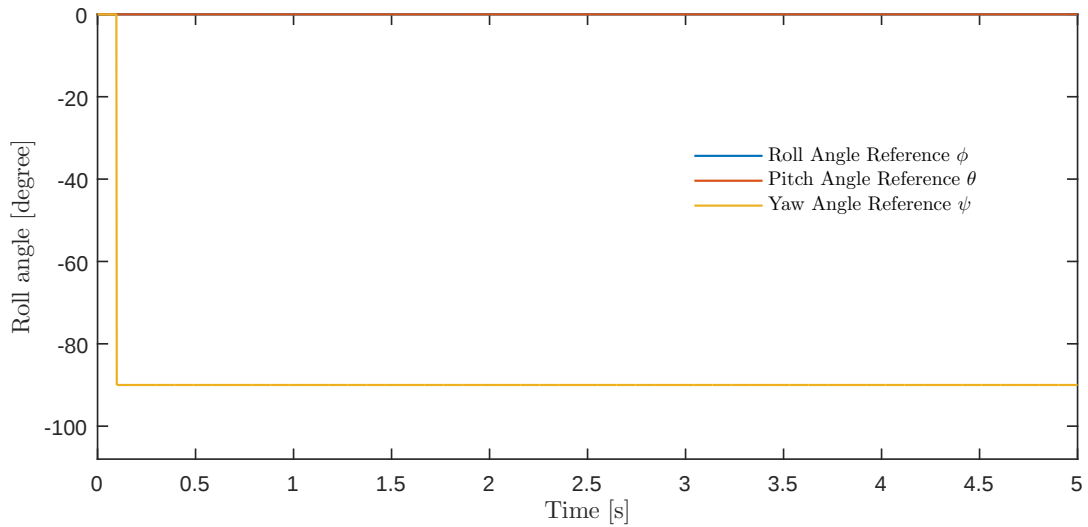
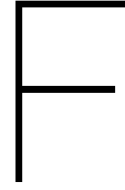


Figure E.6 Reference input to linear controller, yaw step input

For the Sequential Least-Squares method (*SLS*) and Weighted Least-Squares method (*WLS*) algorithms the weighting matrices W_v and W_u need to be defined. For all control allocators the priorities of the control axes are displayed below:

- *WLS*, *SLS*: $W_p = 1$, $W_q = 1$, $W_r = 1$, $W_u = \text{diag}(1)$
- *WLSpg*: $W_p = 10$, $W_q = 10$, $W_r = 1$, $W_u = \text{diag}(1)$

For the *SLS* algorithm a maximum number of iterations is defined and set at 100. The *linDCA* algorithm is the simplex formulation of the *DCA* CA and makes use of the `linDCA` matlab function, using the simplex solver with a tolerance set at $10E-6$.



Analysis of Experiment using a Dynamic Model

To perform a analysis of the *CA* algorithms, a set of control objectives is introduced which originate from the dynamic model introduced in chapter E. Two simulations were performed at 500 Hz.

1. Pitch doublet input (Amplitude 45 degrees)
2. Yaw step input (Amplitude 90 degrees)

Both reference inputs are applied on a single axis, and in ideal conditions this will lead to response only on that specific axis. However, a model discrepancy is introduced, much like would be apparent in actual flight. The constraints on the actuators are constant: $\omega_{\max}$ is set at $+450 \frac{\text{rad}}{\text{sec}}$, and $\omega_{\min}$ is set at $-450 \frac{\text{rad}}{\text{sec}}$. To keep an overview in graphical comparisons and prevent an abundance of similar figures it is chosen to not display the *SLS* and *gDCA* control allocators. The response of these is almost identical to the *WLS* and *linDCA* algorithms.

F.1 First Experiment: Doublet Input on Pitch axis

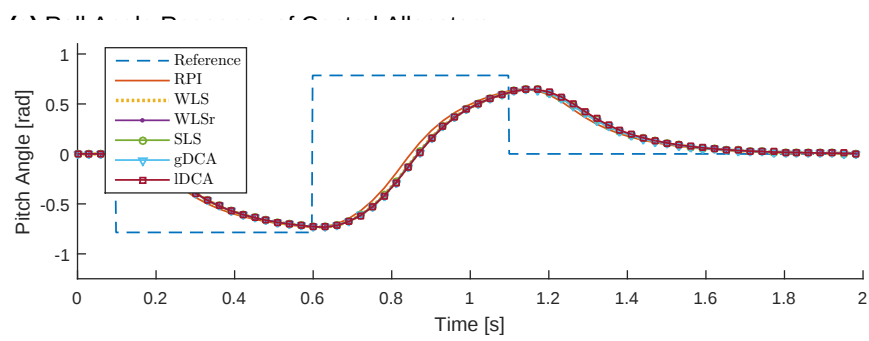
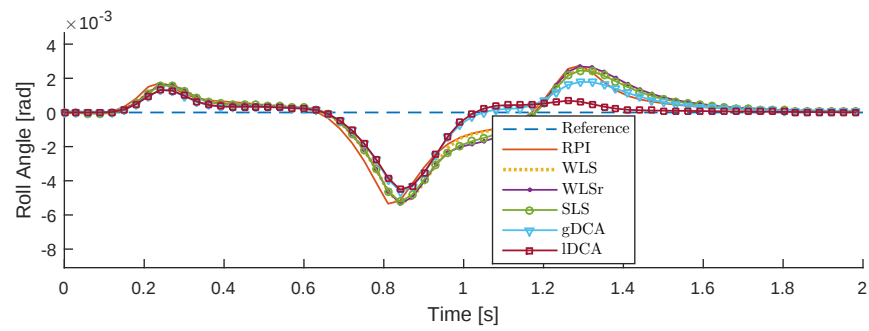
In this experiment, a pitch doublet input is applied on the pitch axis of 45 degrees amplitude. The input is visualized in figure E.5. The individual responses of all the *CA* can be found in appendix G.1. For an initial comparison, the responses of the unconstrained Regular Pseudo Inverse (*RPI*), constrained *RPI*, *WLS*, Prioritized *WLS* and *linDCA* *CA* algorithms to the reference doublet input are presented in figure F.1. Note that the unconstrained *RPI* algorithm serves as a optimality reference.

F.1.1 Analysis of Attitude Response

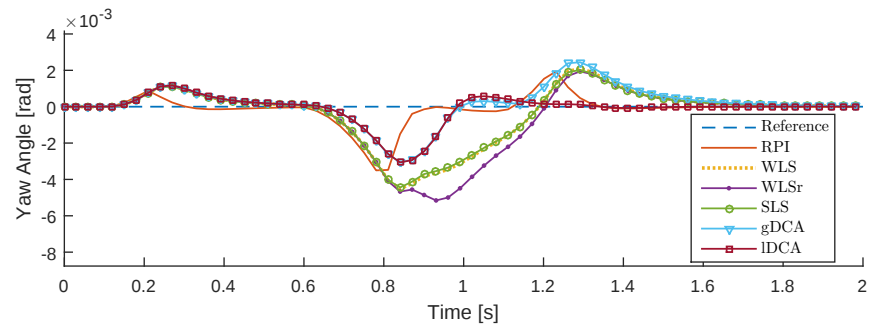
In figure F.1 the time-series response to the reference attitude of the roll, pitch and yaw angle is presented of the first experiment. At a first glance, little difference can be observed from the responses. The response of the implemented control allocators is similar and marginal differences can be observed. For the pitch angle response the *RPI* algorithm shows a slightly better performance compared to the other implemented Constrained Control Allocators.

On the roll and yaw axis, only small disturbances are visible. Again the Unconstrained *RPI* response serves as a reference. On these roll axis the *linDCA* control allocator shows the best performance. One would expect the *WLS* pq *CA* to show optimal performance, however the control objective is in the pitch axis which has an equal weighting factor of 10. On the yaw axis, the *linDCA* algorithm shows the optimal response. Here we would expect that the *WLS* pq *CA* allows for some yaw disturbance, which is also the case. It has to be stressed that the disturbances on the roll and yaw axes are sufficiently small, such that performance is marginally affected.

What can be observed from the response graphs is reflected in the performance metrics shown in table F.1. The $\bar{\epsilon}_{\eta_W}$ and $\bar{\epsilon}_{\eta_W}$ differences between the different *CA* are small. The most optimal performance is achieved by the *RPI* algorithm. The largest difference of the *RPI* Control Objective (*c*)



(b) Pitch Angle Response of Control Allocators



(c) Yaw Angle Response of Control Allocators

Figure F.1 Response of Control Allocators

compared to the other c is visible in the percentage of error compared to the Unconstrained *RPI* algorithm.

F.1.2 Analysis of Actuator Response

Next to the angle response, it is also interesting to observe the actuator response. First observe the actuator response of the unconstrained *RPI* CA. This is pictured in figure F.2. In this figure it becomes apparent that without a control allocator which takes saturation into account, saturation of the actuators will take place. The *RPI* CA response shown in figure F.3 displays a "clipped off" version of the response of the unconstrained *RPI*. The responses of the other CA (in figures F.4, F.5, F.6) is almost identical to each other. Different to the response of the *RPI* CA is the way the constrained CA algorithms reach their saturation limits, slowing a small decrease in rate increase when saturation is almost reached.

Table F.1 Performance Results of Control Allocators in the First Experiment

		Performance Metric					
Control Allocators		$\bar{\epsilon}_\eta$ [rad]	w.r.t RPluc %	$\bar{\epsilon}_{\eta_W}$ []	w.r.t RPluc %	$\bar{\epsilon}_M$ []	w.r.t RPluc %
	RPluc	0.292	0.000	2.918	0.000	0.610	0.000
	RPI	0.377	29.298	3.770	29.208	0.568	-6.791
	WLS	0.400	37.162	3.994	36.885	0.556	-8.823
	WLS r	0.401	37.332	3.998	37.012	0.555	-8.982
	SLS	0.400	37.160	3.994	36.891	0.554	-9.191
	gDCA	0.400	36.924	3.990	36.745	0.556	-8.752
	linDCA	0.402	37.888	4.020	37.788	0.556	-8.778

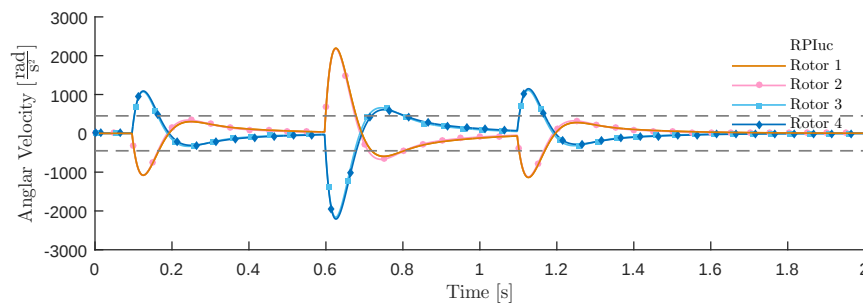


Figure F.2 RPI CA (unconstrained) actuator response for Pitch Doublet Reference

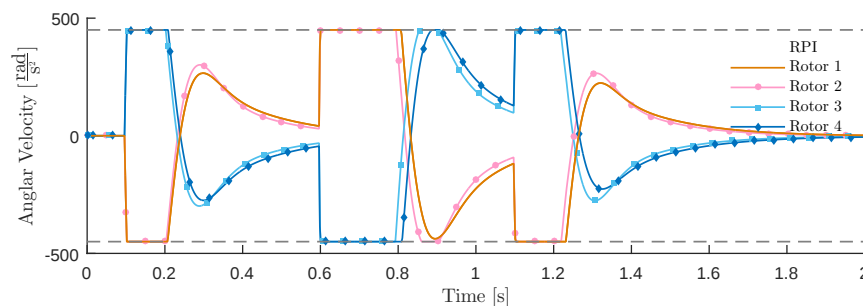


Figure F.3 RPI CA actuator response for Pitch Doublet Reference

F.1.3 Computational Performance

Using Matlab to execute the simulations, the time-span of the simulations for each CA method is used as an indication for computational performance. The *WLS* and *SLS* algorithms both were very efficient, compared to the *gDCA* and *linDCA* control algorithms. The *linDCA* control algorithm uses the simplex

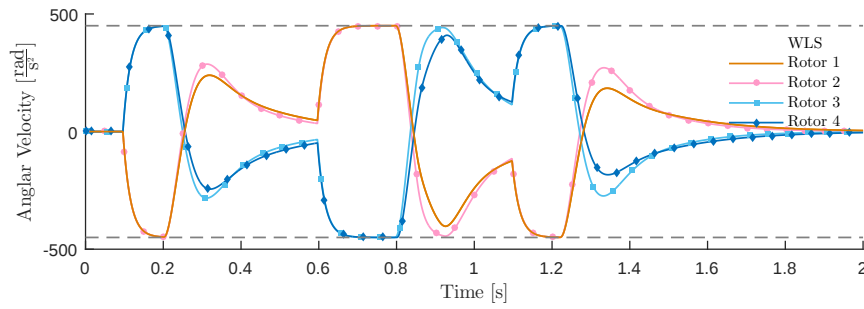


Figure F.4 WLS CA actuator response for Pitch Doublet Reference

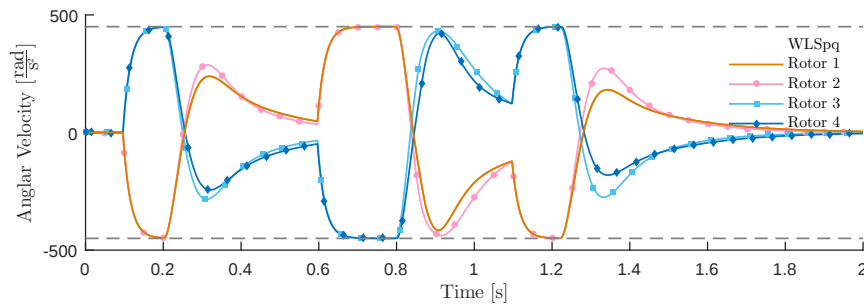


Figure F.5 WLS pq CA actuator response for Pitch Doublet Reference

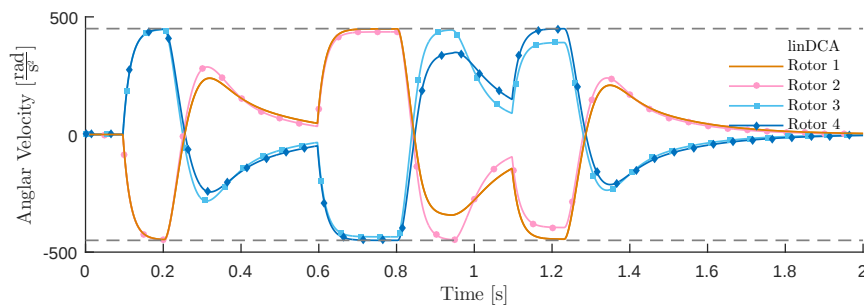


Figure F.6 linDCA CA actuator response for Pitch Doublet Reference

solver built in the linear programming problem solver proved to be the most in-efficient. The computational performance the different CA is displayed in figure F.7.

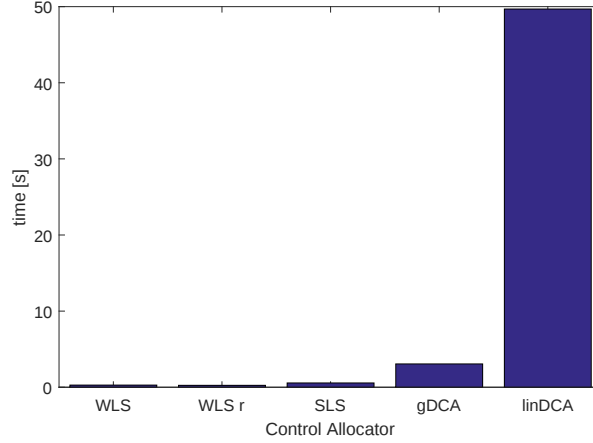


Figure F.7 Computation time of Control Allocators in the second Experiment

F.2 Second Experiment: Yaw Step Input

In the second experiment a step input of 90 degrees amplitude is the reference for the MAV's yaw angle. This experiment resembles the phenomenon observed in practice, where a yaw step input was applied and the response of the MAV with INDI attitude controller showed a loss of height, and a roll and pitch deflection. A qualitative comparable effect was found in the simulation for the yaw step input.

F.2.1 Analysis of Attitude Response

The attitude response for the yaw step input for different implemented CA is shown in F.8. Here the differences between the unconstrained RPI response and the constrained CA becomes obvious. The unconstrained RPI is much faster to converge to the reference yaw angle, without undesired pitch and roll disturbances occurring. Differences between constrained control algorithms are also pronounced.

As can be observed from the F.8 the *linDCA* response stops after approximately 2.6 seconds. The *linDCA* and *gDCA* are unable to maintain stability for the reference. Both algorithms fail to allocate control effort to maintain pitch and roll authority.

Studying the yaw response, the *WLS* and *WLS* prioritized for pitch and roll show nearly identical convergence towards the reference yaw angle. Both exhibit a overshoot, which can be attributed to the non-linearity introduced by actuator saturation to the linearly controlled attitude outer loop. The *RPI* converges towards the reference yaw angle, exhibiting a considerably larger overshoot and slower convergence towards the reference yaw angle.

The *RPI* CA is able to maintain stability in pitch and roll as well as the *WLS* CA, where significant deviations are realized. Here it shows that the *RPI* CA shows higher performance for maintaining zero roll and pitch angle than the *WLS* CA. However the *WLS* CA prioritized for pitch and roll exhibits excellent performance for maintaining the zero pitch and roll angle. For pitch and roll attitude, significantly less actuator effort is required for equivalent attitude deflections compared to the yaw attitude. The weighted *WLS* CA is able to allocate enough effort to satisfy the pitch and roll attitude. The yaw performance only suffers marginally from the control allocation to the pitch and roll attitude as it is similar to that of the *WLS* CA.

In table F.2 the performance metrics of the different control allocators are presented for the yaw step input. The *gDCA* and *linDCA* were unable to maintain stability, such that no performance data is available. The *WLS* CA prioritized for pitch and roll, shows superior performance compared to the *RPI*, *WLS* and *SLS* CAs. This is already apparent in the non-weighted $\bar{\epsilon}_\eta$ and a major difference in the weighted average attitude error $\bar{\epsilon}_{\eta_{wv}}$. Similar in nature, the *SLS* and *WLS* CA have a similar score, where the *SLS* exhibits a slightly better performance, however not significant. Both the *WLS* and *SLS* CAs show improved performance for the unweighted error $\bar{\epsilon}_\eta$, however for the weighted error this increase in performance is reduced to non-significance, as the differences between the CA are

very small.

F.2.2 Analysis of Actuator Response

The actuator response provides valuable insight in the workings of the CA. First, observe the response of the unconstrained *RPI* CA in figure F.9. It shows that for fast convergence to the reference yaw angle, the model requires very high actuator load. The angular velocity of the actuators reach a extreme points of approximately 7000 and -7000 radians per second. This is well over the actuator constraints.

The *linDCA* CA is unable to maintain stability. The actuator response of the *linDCA* CA is depicted in figure F.13. The *DCA* algorithms allocate control effort based on the direction of the control objective. When the yaw command is issued, all the control effort is allocated to accelerate the yaw rotation. Due to the mismatch in the identified control effectiveness matrices G_1 and G_2 , a rolling and pitching rate develops. Because the actuators are saturated at extrema for maximum yaw rate, there is no actuator "budget" available for corrections which require the same sign as the yawing motion. Because the *DCA* algorithms follow the direction of the control objective, control effort will be allocated to the pitch and roll angle once the control objective is sufficiently large with respect to the yaw control objective. This occurs when the linear controller slows the yaw rate, and the MAV's yawing motion has to be decelerated. The errors on the roll and pitch axis are already sufficiently large such that the simulation model cannot be recovered.

The *RPI* CA response, again shows a clipped of variant of the unconstrained *RPI* CA actuator response. The *RPI* is able to recover itself from the induced roll and pitch rate, however it is expected that for a larger yaw step command, divergence occurs. The *WLS* CA response is displayed in figure F.11. Clearly the control effort is allocated to the yaw acceleration. When after approximately 2.5 seconds the error on the roll and pitch angles becomes sufficiently large, the control objective input to the *WLS* CA allows for correcting the pitch and roll attitude. This can observed in the figure by the alternative trajectories rotor 1 and rotor 2.

Finally the response of the prioritized *WLS* CA can be analyzed. As shown in figure F.12, the actuator trajectories are more symmetrical compared to that of the *WLS* CA. However, small perturbations are visible in the duration where the actuators are in saturated positions. This is were the *WLS* prioritized for roll and pitch already corrects for small pitch and roll disturbances experienced by the MAV. Because the control effectiveness is much more effective in roll and pitch, this requires a relatively small control effort. With the *WLS* CA prioritized for roll and pitch, vastly superior performance is achieved by providing minute corrections in the roll and pitch axis to keep these at their reference attitude. The remaining control capability can be fully allocated to achieve the yaw attitude reference.

Table F.2 Performance Results of Control Allocators in the Second Experiment

		Performance Metric					
Control Allocators		$\bar{\epsilon}_\eta$ [rad]	w.r.t RPluc %	$\bar{\epsilon}_{\eta w}$ []	w.r.t RPluc %	$\bar{\epsilon}_M$ []	w.r.t RPluc %
	RPluc	0.062	0.000	0.064	0.000	0.960	0.000
	RPI	0.576	822.005	1.918	2880.419	0.954	-0.666
	WLS	0.469	650.667	1.925	2890.6534	0.909	-5.268
	WLS r	0.317	408.041	0.379	488.735	0.901	-6.137
	SLS	0.461	638.436	1.842	2762.138	0.911	-5.141
	gDCA	N/A	N/A	N/A	N/A	N/A	N/A
	linDCA	N/A	N/A	N/A	N/A	N/A	N/A

F.2.3 Computational Performance

The computational performance during the second experiment, reflects that of the first experiment. The performance of different CA is displayed in figure F.14. Little difference in performance is visible compared to the first experiment, however the *linDCA* algorithm shows an increased computation

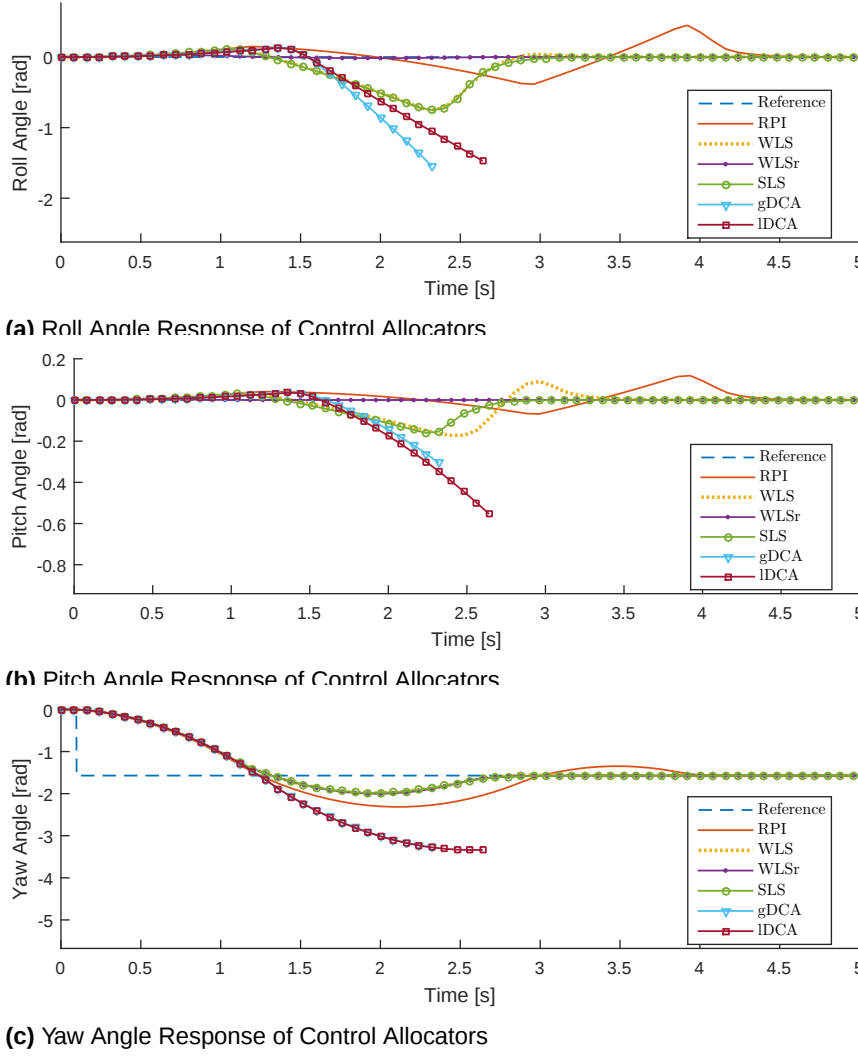


Figure F.8 Response of Control Allocators

F.3 Conclusions from Simulation& Recommendations

In this chapter the general conclusion is presented based on the simulation experiments performed. First a general summary is presented, after which preliminary conclusions are presented as well as recommendations for continuation of this research.

F.3.1 Prioritization in Constrained Control Allocation

What makes *WLS* and *SLS* methods particularly interesting is the ability to include weighting matrices, which can prioritize the control objective solution (W_v) or usage of the actuators (W_u). For example, we can assign a high to the yaw c . In a situation where a mixed (roll, pitch and yaw) c objective normally requires actuator effort over the saturation limits, the CA assigns a disproportionate amount of available actuator control effort to the yaw CA. Theoretically, this solution is not optimal, because the c solution where all states are weighted equal has the error vector with the smallest possible magnitude to the c . This changes however, if the weighted error which is weighted with the same weights as the weighted solution. Also, as the experiments showed, the weighted solution can provide the best possible dynamical properties to the system as a whole.

The main difference between the *SLS* and *WLS* control allocators is how the control objective solution and actuator command output is computed. In *WLS* both solutions are combined in a single *QP* problem, whereas the *SLS* algorithm first finds the best control objective solution and subsequently the optimal actuator command. The *WLS* algorithm is faster and easier in implementation, whereas

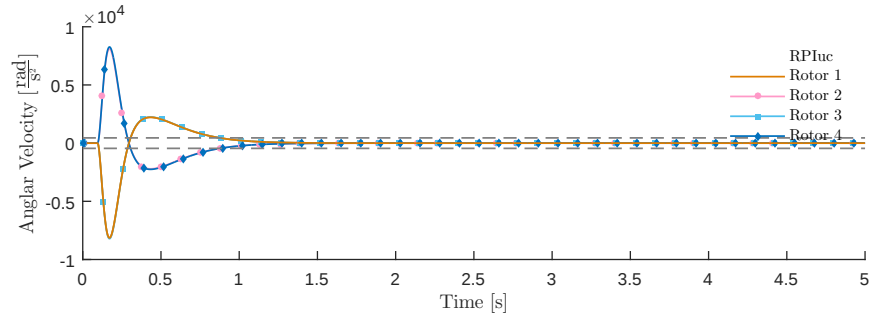


Figure F.9 RPI CA (unconstrained) actuator response for Yaw Step Reference

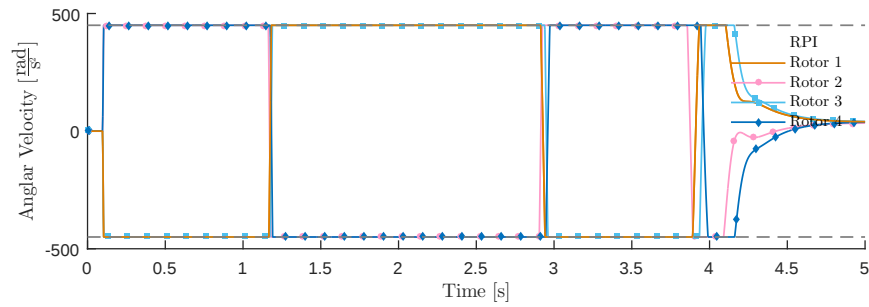


Figure F.10 RPI CA actuator response for Yaw Step Reference

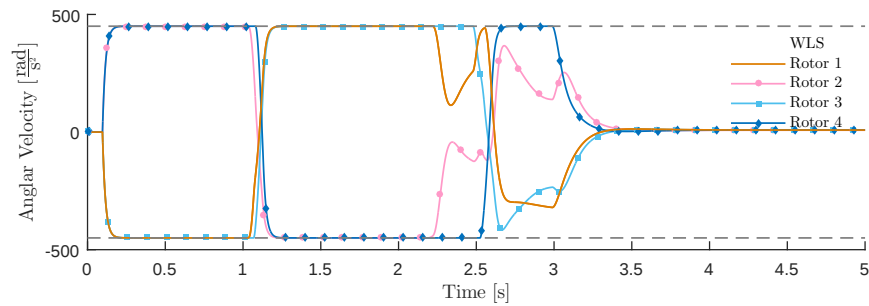


Figure F.11 WLS CA actuator response for Yaw Step Reference

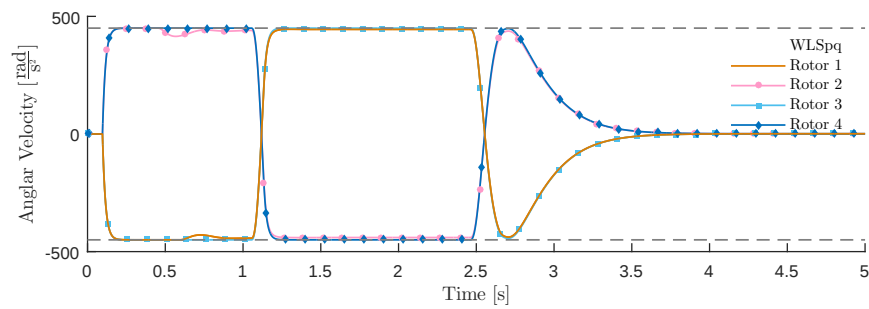


Figure F.12 Weighted WLS CA actuator response for Yaw Step Reference

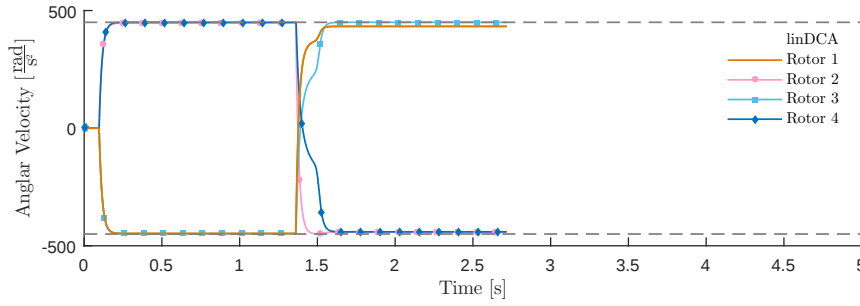


Figure F.13 lin DCA actuator response for Yaw Step Reference

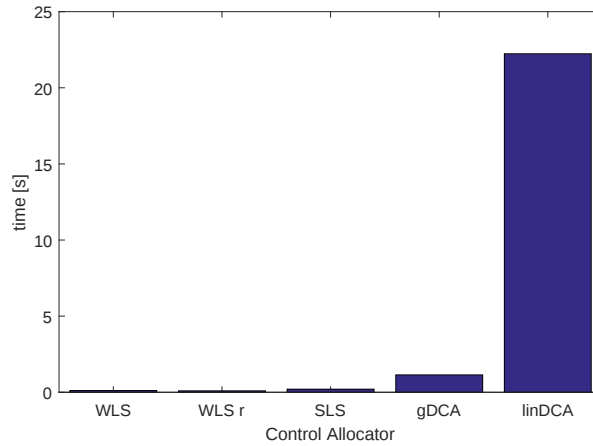


Figure F.14 Computation time of Control Allocators in the second Experiment

the *SLS* provides better solution quality.

F.3.2 Limited Fidelity Model

To assess the control allocators, a limited fidelity model was established. In this model, the *MAV*'s dynamics are modeled with an actuator effectiveness matrix different from the one used by the control allocator used by the *INDI* controller. The difference in between the control effectiveness matrices is small, where the average deviation is approximately 3%. The modeled *MAV* resembles Parrot Bebop consumer drone. The output of the CA will introduce coupled attitude response which the linear controller will experience as a disturbance. It has to be noted that the model provides qualitative results, as the quantitative output is not representative for actual dynamics.

To provide assess the control allocators quantitatively, a set of performance metrics was introduced. This includes the average error as well as a weighted error vector and magnitude error of the of the *MAV*'s attitude with respect to the reference attitude. The response of the *MAV* modeled with **constrained** actuators and the set of control allocators are compared to an *RPI* control allocator which is modeled with **unconstrained** actuators as an optimality reference. The error is also expressed as a percentage of the error which is achieved by the unconstrained *RPI* as it defines the maximum attainable performance.

F.3.3 Conclusions from the Experiments

To assess the control allocators, a qualitative experiment is executed. In this qualitative experiment, several reference angle commands are given. Two different scenarios are introduced which will lead to different responses of the control allocators:

1. Doublet input on the pitch axis (+45 [deg] and −45 [deg])
2. Step input on the yaw axis (90 [deg])

F.3.4 First Experiment: Doublet Pitch Angle Input

The purpose of the first experiment is to demonstrate the performance of the CA for a reference control task where no problems were encountered when flying the *INDI* controlled Bebop. The *MAV* is effective in roll and pitch attitude control relative to the yaw attitude. This can be attributed the physical workings of the Quad-Copter platform, where the roll and pitch-rate are controlled via the thrust output of the actuators, and the yaw-rate by the moment which is caused by the inertia and drag of the propellers. For the first experiment, the vehicle is effective in control of the required reference control task. Therefore, the expected undesired coupling of attitude states is relatively small.

The response of the implemented control allocators for the reference doublet attitude input is similar and marginal differences can be observed between the CA. For the pitch angle response the *RPI* algorithm shows marginally better performance compared to the other implemented Constrained Control Allocators. On the roll and yaw axis small disturbances are visible (in the order of 10^{-3} radians). This experiment shows the beneficial properties of the *DCA* control allocator, where the error on the roll and yaw axes is the smallest.

The differences between the actual actuator output is virtually non-existent between the different CA for the first experiment.

The data from the performance metrics reflects the observed response. The differences between different CA is very small, and therefore this experiment provides inconclusive results to select a control allocator for implementation.

F.3.5 Second experiment: Yaw Step Input

The goal of the second experiment was to simulate the coupled attitude response observed in practice for a yaw step input. Because yaw attitude control requires very high control effort relative to the roll and pitch control, the undesired coupling manifests itself much more relative to reference control task in experiment 1.

The output data of the second experiment showed vastly different performance between the different CA.

The *DCA* control allocators were both unable to produce viable results. The simulation stopped when a roll or pitch angle over 90 degrees was logged. The *DCA* algorithms allocate all their available control authority to the yaw rate acceleration. This leads to full usage of the available actuators. Because no actuator "budget" is available for corrections which require the same sign as the yawing motion, the *DCA* control allocators are unable to allocate the required control authority to the roll and pitch disturbance. Once the control input to the CA of the roll and pitch axes becomes sufficiently large such that the *DCA* control allocators assign control effort to roll and pitch recovery, the system already is unstable.

The default *RPI* control allocator is able to maintain stability and converge to the reference attitude. Compared to the regular (non-weighted) *WLS* and *SLS* CA the pitch and roll deviations of the *RPI* are smaller. However, the *RPI* CA requires significantly longer to converge to the reference attitude and the overshoot on the yaw axis is significantly more than compared to the *WLS* and *SLS* CA. Observing the performance metrics, the unweighted error score of the *WLS* and *SLS* is better than that of the *RPI*, whereas the weighted error score is similar to that of the *WLS* and *SLS* CA.

The weighted Weighted Least-Squares method, prioritized for roll and pitch (*WLS pq*) control allocators shows vastly superior response. The weights are 10 on both the roll and pitch axes compared to 1 on the yaw axis. The *WLS pq* CA maintains the reference roll and pitch angle of zero radians, while providing the identical response on the yaw axis as the *WLS* and *SLS* CA. A small overshoot on the yaw axis is still present. The performance metrics of the *WLS pq* CA is much better compared to the other constrained control allocators on all accounts.

F.3.6 Computational Performance

Using Matlab to execute the simulations, the time-span of the simulations for each CA method is used as an indication for computational performance. The *WLS* and *SLS* algorithms both were very efficient, compared to the *gDCA* and *linDCA* control algorithms. The *linDCA* control algorithm uses the simplex solver built in the linear programming problem solver proved to be the most in-efficient.

F.3.7 Experiment Conclusions

For convenience, the top-level results of the preliminary experiments conducted are displayed in a rudimentary notation in table F.3. The plus and minus signs denote the performance of the different CA with respect to the *RPI* CA with no actuator constraints and each-other. This is a qualitative table, with the purpose to quickly give an overview of the results from the experiments conducted.

Table F.3 Overview of relative Performance of multiple Control Allocators on each axis

		Pitch Doublet			Yaw Step		
Control Allocators		Roll	Pitch	Yaw	Roll	Pitch	Yaw
	RPI_{uc}	++	++	++	++	++	++
	RPI	++	++	+	-	-	--
	WLS	++	++	+	--	--	-
	WLS <i>pq</i>	++	++	+	++	++	-
	SLS	++	++	+	-	-	-
	gDCA	++	++	+	NA	NA	NA
	linDCA	++	++	+	NA	NA	NA

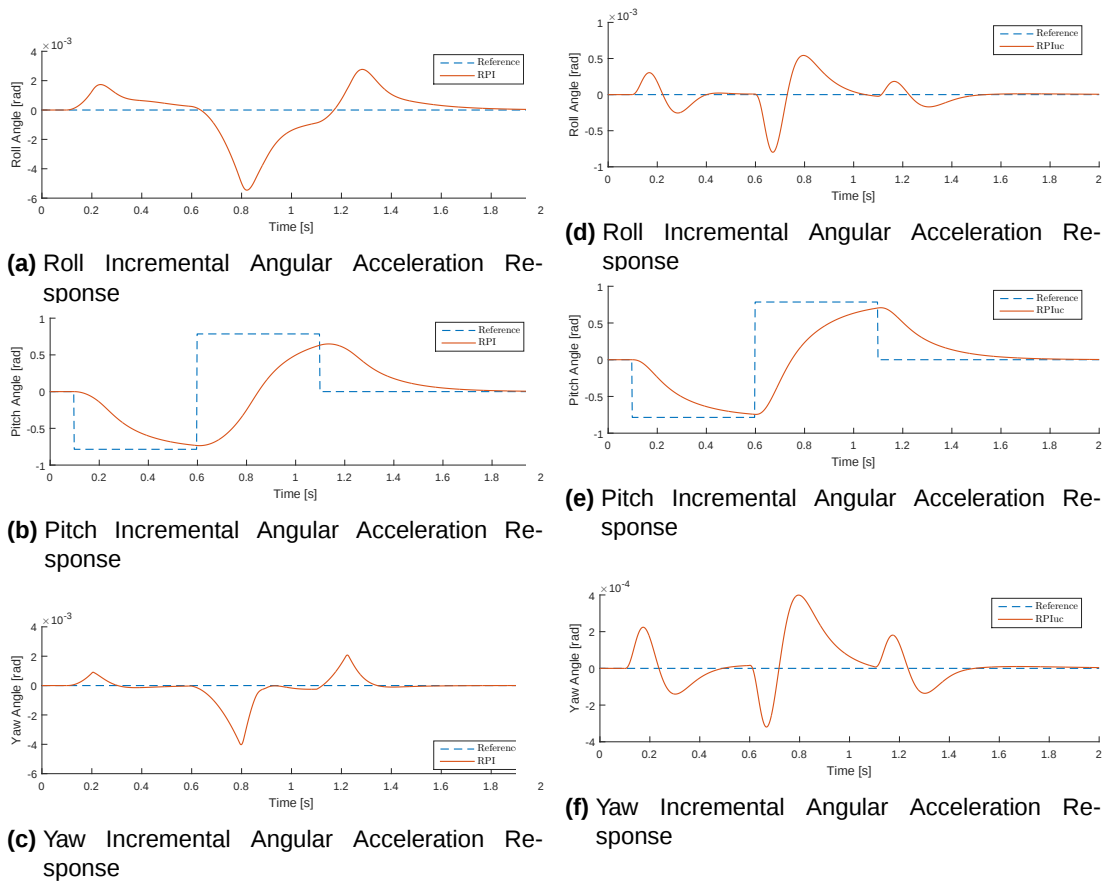
As a general conclusions on the preliminary experiment simulations conducted, the *WLS *pq** CA is identified as the optimal control allocation algorithm. The main reasoning is the vastly superior performance in the experiment where the yaw step reference attitude was required. The *WLS *pq** control allocator enabled virtually full cancellation of the undesired coupled attitude effects, which is a product of this research's objective. While providing superior performance in the second experiment, the *WLS *pq** control allocator does not provide significantly less performance in the pitch reference attitude control task. The difference between the *SLS* and *WLS* control allocation algorithms is small. The *WLS* control allocator is preferred due to it's slightly better computational performance as enabling easier implementation.

G

Simulation Output Graphs

G.1 Control Allocator Output Experiment 1 (Pitch Doublet Input)

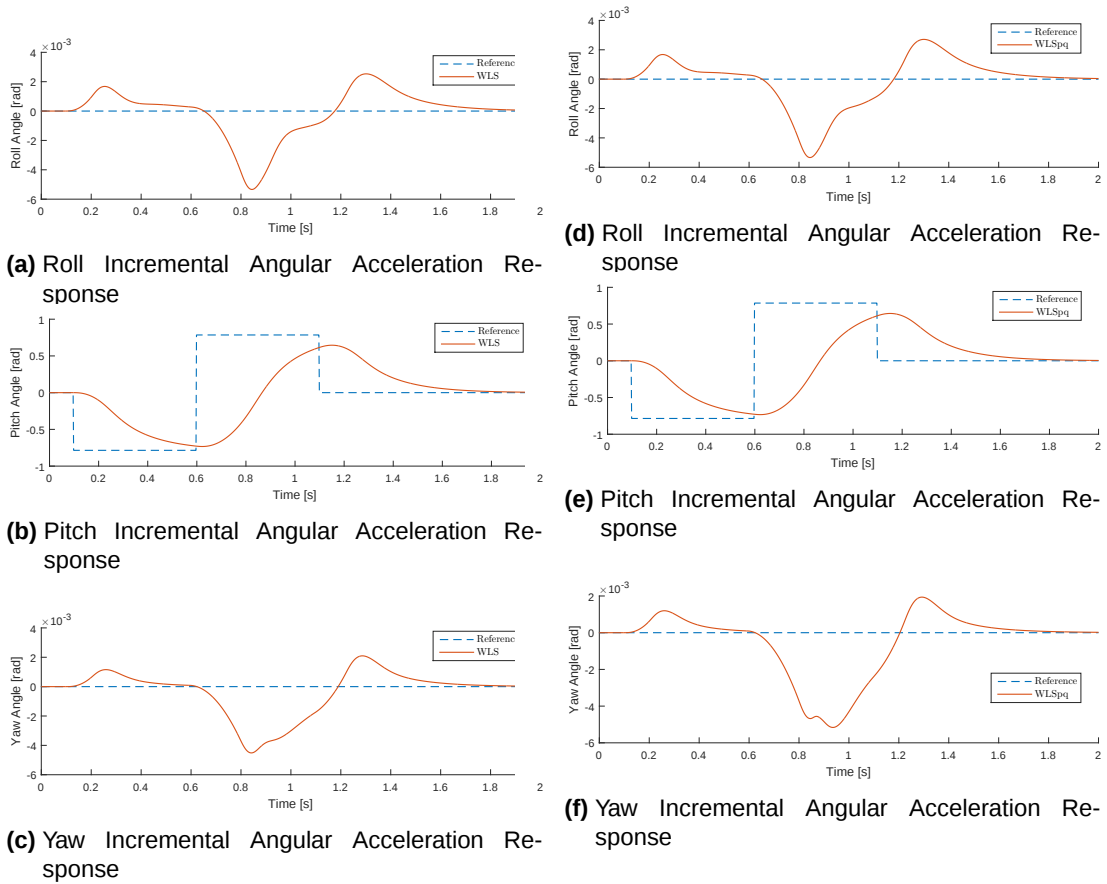
In this appendix the individual plots of the responses of the simulated pitch doublet experiment of all the researched control allocators are presented. This includes prioritized WLS, regular WLS (unit weights), Geometric DCA, Simplex DCA, Unconstrained Pseudo Inverse, and constrained Pseudo-inverse.



Response of RPI Control Allocator

Response of RPI (unconstrained) Control Allocator

Figure G.1 Response of RPI and unconstrained RPI Control Allocators for pitch doublet simulation



Response of WLS Control Allocator

Response of WLS prioritized for pitch and roll Control Allocator

Figure G.2 Response of WLS and WLSpq Control Allocators for pitch doublet simulation

G.2 Control Allocator Output Experiment 2 (Yaw Step Input)

In this appendix the individual plots of the responses of the simulated yaw step experiment of all the researched control allocators are presented. This includes prioritized WLS, regular WLS (unit weights), Geometric DCA, Simplex DCA, Unconstrained Pseudo Inverse, and constrained Pseudo-inverse.

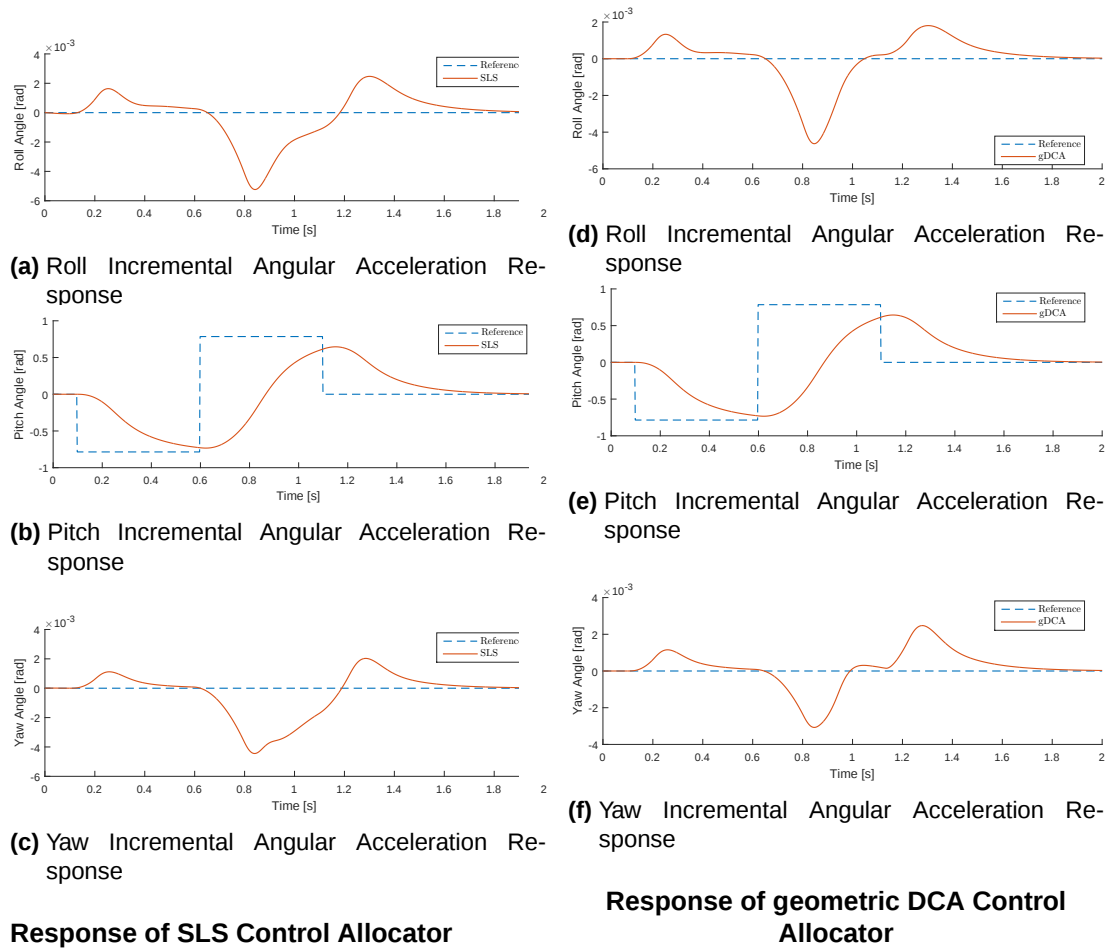
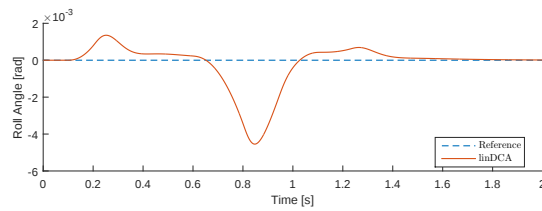
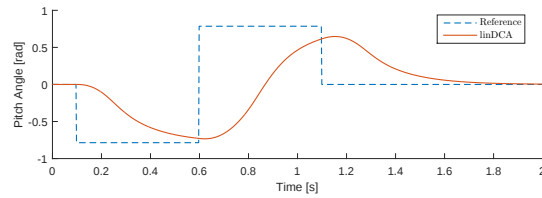


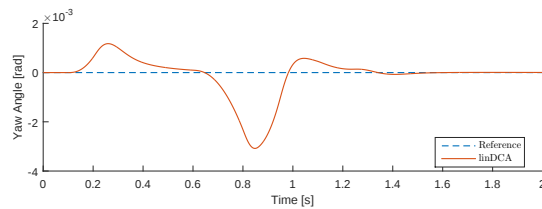
Figure G.3 Response of SLS and gDCA Control Allocators for pitch doublet simulation



(a) Roll Incremental Angular Acceleration Response



(b) Pitch Incremental Angular Acceleration Response



(c) Yaw Incremental Angular Acceleration Response

Response of simplex DCA Control Allocator

Figure G.4 Response of lin DCA Control Allocator for pitch doublet simulation

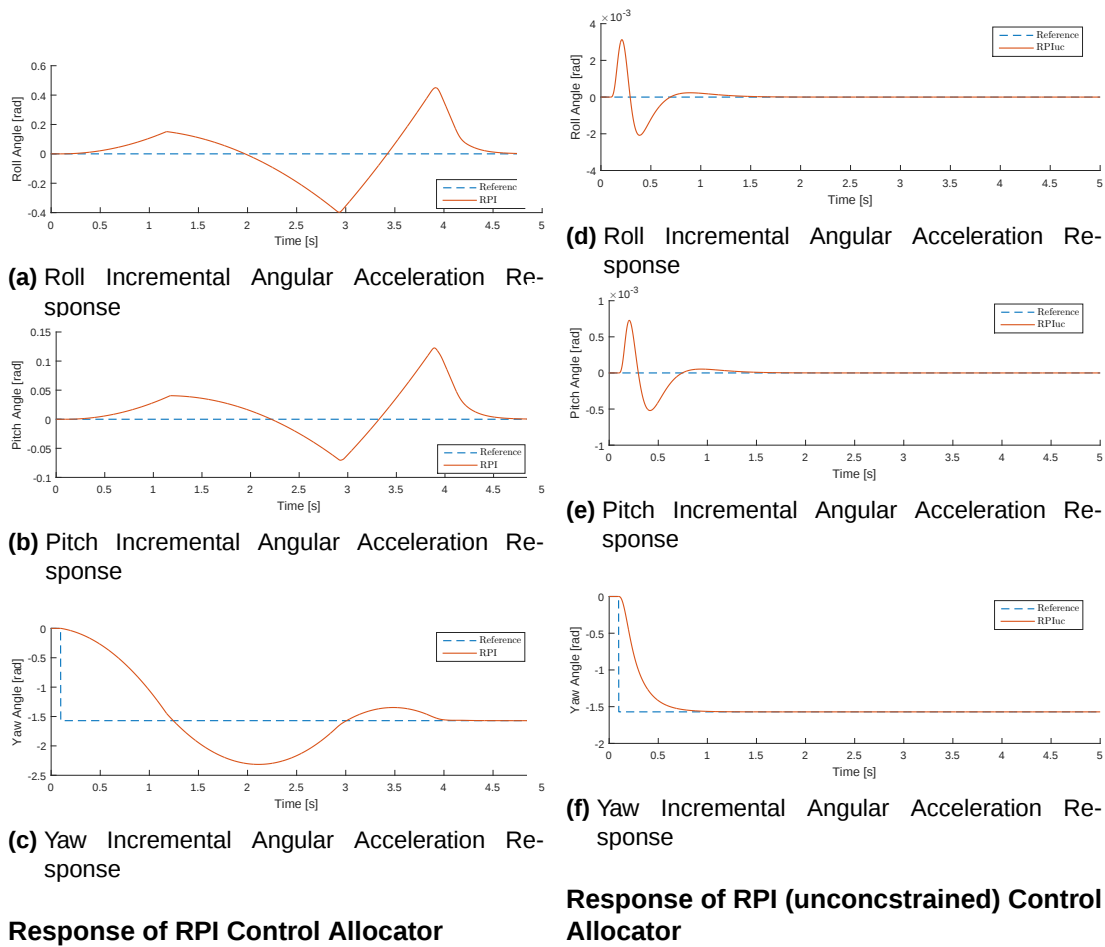
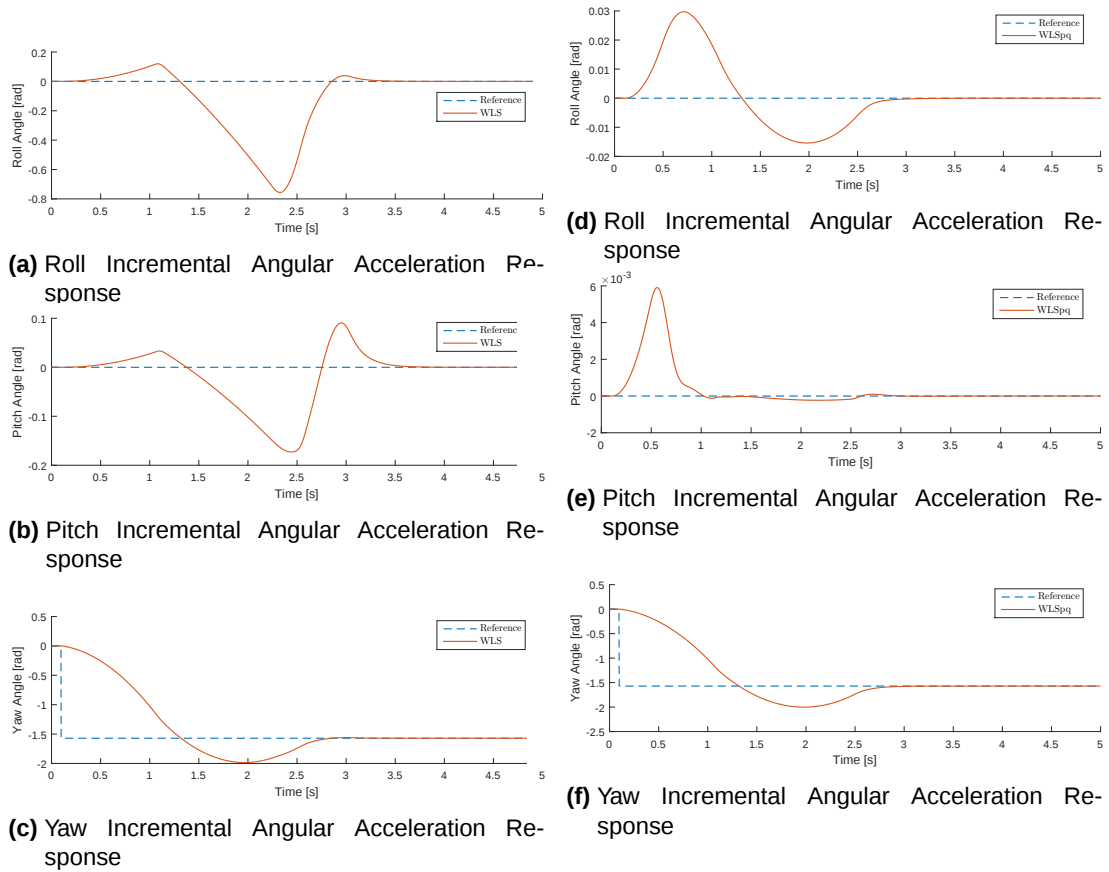


Figure G.5 Response of RPI and unconstrained RPI Control Allocators for yaw step simulation



Response of WLS Control Allocator

Response of WLS prioritized for pitch and roll Control Allocator

Figure G.6 Response of WLS and WLSpq Control Allocators for yaw step simulation

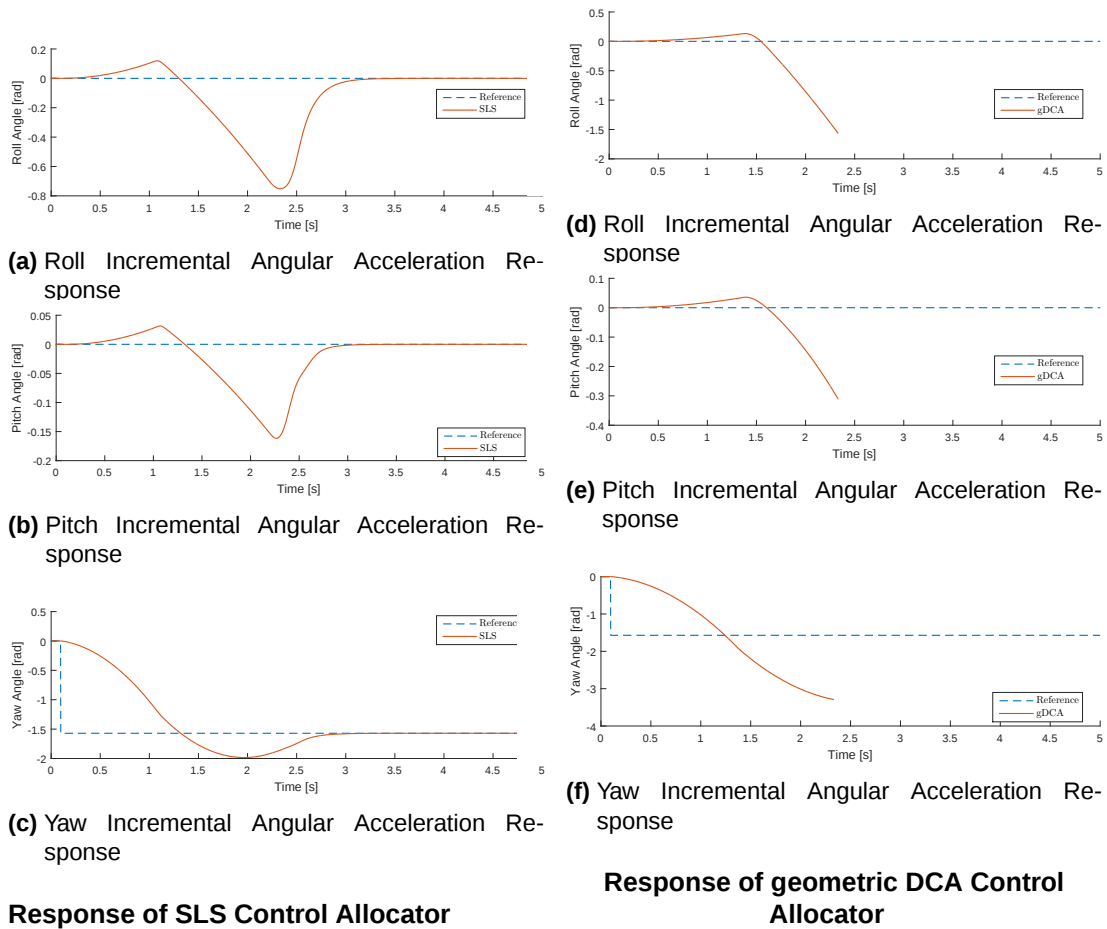
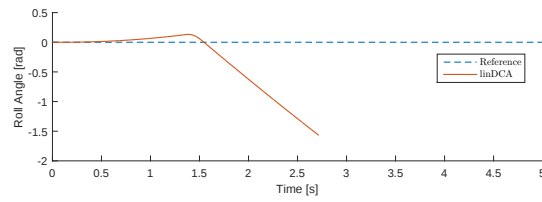
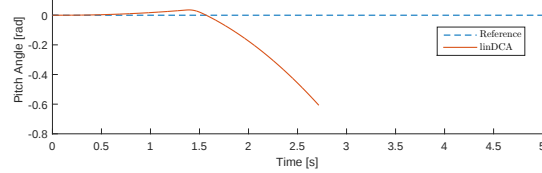


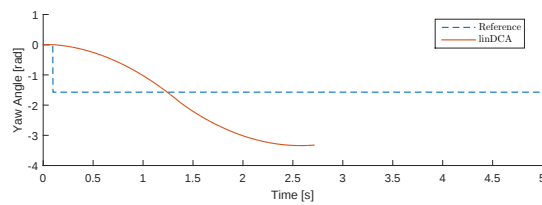
Figure G.7 Response of SLS and gDCA Control Allocators for yaw step simulation



(a) Roll Incremental Angular Acceleration Response



(b) Pitch Incremental Angular Acceleration Response



(c) Yaw Incremental Angular Acceleration Response

Response of simplex DCA Control Allocator

Figure G.8 Response of lin DCA Control Allocator for yaw step simulation

Bibliography

- [1] E. J. J. Smeur, "Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Aerial Vehicles," *AIAA Guidance, Navigation, and Control Conference*, pp. 1–16, 2016.
- [2] B. Theys, G. Dimitriadis, T. Andrianne, P. Hendrick, and J. De Schutter, "Wind tunnel testing of a VTOL MAV propeller in tilted operating mode," *2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014 - Conference Proceedings*, pp. 1064–1072, 2014.
- [3] J. M. Buffington, "Tailless aircraft control allocation," *AIAA Guidance, Navigation, and Control Conference*, pp. 737–747, 1997.
- [4] O. Härkegård, "Efficient active set algorithms for solving constrained least squares problems in aircraft control allocation," *Decision and Control, 2002, Proceedings of the ...*, vol. 2, no. December, pp. 1295–1300, 2002.
- [5] J. C. Virnig and D. S. Bodden, "Multivariable Control Allocation and Control Law Conditioning when Control Effectors Limit," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 572–582, 1994.
- [6] T. Schneider, G. Ducard, K. Rudin, and P. Strupler, "Fault-tolerant Control Allocation for Multirotor Helicopters using Parametric Programming," *International Micro Air Vehicle Conference and Flight Competition*, 2012.
- [7] J. A. M. Petersen and M. Bodson, "Constrained Quadratic Programming for Control Allocation," *IEEE Trans. Control and Dynamics*, vol. 14, no. 1, pp. 91–98, 2006.
- [8] H. Ola, "Quadratic Programming," vol. 27, no. 6, 2004.
- [9] J. Nocedal, S. J. Wright, and S. M. Robinson, *Numerical Optimization*, 1999.
- [10] M. D. Nelson and W. C. Durham, "A Comparison of Two Methods Used to Deal with Saturation of Multiple, Redundant Aircraft Control Effectors," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, no. August, pp. 1–12, 2002.
- [11] M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 45–52, 2014.
- [12] M. V. Kothare, P. J. Campo, M. Morari, and C. N. Nett, "A unified framework for the study of anti-windup designs," *Automatica*, vol. 30, no. 12, pp. 1869–1883, 1994.
- [13] E. Johnson and A. Calise, "Pseudo-control hedging: A new method for adaptive control," *Advances in Navigation Guidance and Control Technology*, vol. September, 2000.
- [14] T. A. Johansen and T. I. Fossen, "Control Allocation - A Survey," *Automatica*, vol. 49, 2012.
- [15] O. Härkegård, *Backstepping and control allocation with applications to flight control*, 2003, no. 820.
- [16] M. Frangenberg, J. Stephan, and W. Fichter, "Fast Actuator Fault Detection and Reconfiguration for Multicopters," *AIAA Guidance, Navigation, and Control Conference*, no. January, pp. 1–24, 2015.
- [17] B. Erginer and E. Altug, "Modeling and PD Control of a Quadrotor VTOL Vehicle," *2007 IEEE Intelligent Vehicles Symposium*, pp. 894–899, 2007.

- [18] W. C. Durham, "Attainable moments for the constrained control allocation problem," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 6, pp. 1371–1373, Nov. 1994.
- [19] W. C. Durham, "Constrained control allocation - Three-moment problem," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 2, pp. 330–336, Mar. 1994.
- [20] W. C. Durham, "Constrained control allocation," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 4, pp. 717–725, Jul. 1993.
- [21] W. C. Durham, "Computationally Efficient Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, 2001.
- [22] P. De Monte and B. Lohmann, "Trajectory tracking control for a quadrotor helicopter based on backstepping using a decoupling quaternion parametrization," *21st Mediterranean Conference on Control and Automation*, pp. 507–512, 2013.
- [23] J. J. Burken, P. Lu, Z. Wu, and C. Bahm, "Two Reconfigurable Flight-Control Design Methods: Robust Servomechanism and Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, pp. 482–493, 2001.
- [24] J. M. Buffington and D. F. Enns, "Lyapunov Stability Analysis of Daisy Chain Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 6, 1996.
- [25] J. M. Buffington and Y. B. Shtessel, "Saturation protection for feedback linearizable systems using sliding mode theory," *Proceedings of the American Control Conference*, vol. 2, no. June, pp. 1028–1032, 1998.
- [26] K. A. Bordignon and W. C. Durham, "Closed-form solutions to constrained control allocation problem," *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 5, pp. 1000–1007, Sep. 1995.
- [27] M. Bodson, "Evaluation of Optimization Methods for Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, pp. 703–711, 2002.