RAM ROBOTICS AND MECHATRONICS

OPTIMIZED DESIGN OF A TILTED PROPELLER AERIAL ROBOT FOR BALLAST TANK CONTACT INSPECTIONS

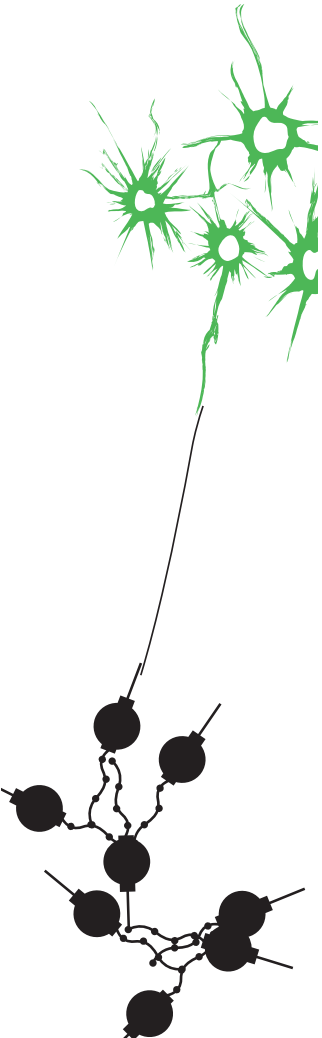
D. (Dimitrios) Nikitas

MSC ASSIGNMENT

Committee:
dr. ir. J.F. Broenink
ir. A.N.M.G. Afifi
dr. ir. R.G.K.M. Aarts

September, 2024

064RaM2024
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands



UNIVERSITY OF TWENTE. | TECHMED CENTRE UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Acknowledgements

I would like to express my deepest gratitude to my professor and supervisor, prof.dr.ir. Antonio Franchi, for his invaluable guidance, support and wisdom throughout my MSc studies. I could not have undertaken this journey without ir. Amr Afifi, my daily supervisor, whose unwavering dedication, insightful feedback and hands-on assistance were pivotal to the completion of this thesis.

I would like to extend my appreciation to the members of the defence committee for their time and effort in evaluating this work. Additionally, I would like to acknowledge Marco Cognetti and Gianluca Corsini for their external input and support during this project.

On a personal note, I am forever grateful to my mother Tina, my grandparents, Nikos and Maria, and my loved ones, whose patience, support, encouragement and unwavering belief have been a constant source of strength.

Last but not least, I want to thank my close friends and partner, for all the memorable moments we shared, their motivation and emotional support when I needed it most.

Thank you all.

Summary

In the last decade, the use of MRAVs (Multirotor Aerial Vehicles) across multiple domains, has expanded significantly. Technological innovations have greatly enhanced their capabilities and rendered them well-suited for operating in perilous areas such as maritime environments, like ballast tanks. These confined and hazardous areas pose risks to human inspectors, making MRAVs a safer and more reliable alternative. However, there remains a pressing need for optimized MRAV designs tailored to complex tasks, such as autonomous contact inspections in these challenging areas. This thesis presents a generalized, systematic, design procedure aimed at optimizing MRAVs based on their tasks and through that, propose a suitable design, optimized for the task of autonomous contact-inspection on the walls of a ballast tank, under the scope of the AUTOASSESS project.

At the core of this procedure lies the formulation of the design problem, as a constrained multivariable nonlinear optimization problem (NLP), allowing the holistic consideration of task requirements through constraints or components of a unified objective function, derived from a combination of modified cost functions from the literature. After all, it is provided a robust mathematical representation of a task, incorporating key requirements inspired by the AUTOASSESS project. The task requirements, serve as input to a MATLAB-based software, developed using CasADi, with IPOPT employed as the NLP's solver. The software outputs optimized MRAV designs tailored to the tasks and with a carefully selected set of performance metrics, one is able to evaluate and compare between them, in order to make informed decisions about the optimal configuration, based on task requirements and design performance. The software framework developed, demonstrates wide flexibility and realism in terms of input parameters and design variables to be optimized, from tilt angles to rotor positions. Besides, it provides extensive configuration options and the ability to design, at will, the wrench space of the final MRAV, further reinforcing the adaptability and practicality of the framework to a variety of tasks.

Validation is carried out through a series of simulations in Gazebo and real-world experiments, where a set of designs from the proposed optimization framework, candidates for the AUTOASSESS, is used. The simulations replicate the narrow corridors of ballast tanks, testing the MRAV's ability to generate the required contact force while navigating tight spaces and challenging its endurance. Real-world experiments are conducted to determine the thrust and drag coefficients and maximum rotational speeds of various sets of propellers, candidates for the proposed design, to ensure that the theoretical models align with practical, realistic outcomes. Although time constraints did not allow for flight and contact tests with the real drone proposed for the AUTOASSESS project, the design adhered exactly to all parameters outlined by the framework and successfully passed hover tests. Future work will focus on conducting contact-based experiments.

Finally, the results demonstrate the efficiency and feasibility of the optimized designs. Unlike traditional configurations which optimize for specific performance characteristics, the proposed procedure takes a holistic approach, optimizing the system for multiple task requirements such as wrench generation, size, and energy efficiency. The framework's modular and adaptable nature allows it to be applied across various tasks and environments, extending its utility beyond the immediate application.

Contents

List of Figures

List of Tables

List of Acronyms

AU Actuator Unit
CCW Counter-Clockwise
CoM Center of Mass
CORP Configuration Options for Rotor Positions
COTA Configuration Options for Tilt Angles
CW Clockwise
DC Direct Current
DOF Degrees of Freedom
EE End Effector
ESC Electronic Speed Controller
EU European Union
FA Fully Actuated
GUI Graphical User Interface
IMO International Marine Organization
IPOPT Interior Point OPTimizer
MRAV Multirotor Aerial Vehicle
NDT Non Destructive Testing
NLP Nonlinear Programming
PID Proportional-Integral-Derivative (controller)

1 Introduction

Recent advancements across a wide technological spectrum have significantly enhanced the capabilities and applications of Multirotor Aerial Vehicles (MRAVs) in various sectors, including commercial, industrial, scientific, and military domains. Innovations such as Nondestructive Testing (NDT) sensor miniaturization, lightweight materials, advanced path planning, and machine learning-based defect identification are democratizing drone technology and reshaping industries by addressing complex challenges with increased safety and efficiency. However, there remains a critical need for optimized MRAV designs specifically tailored for demanding tasks, such as autonomous contact inspections in hazardous and confined environments. MRAVs are particularly well-suited for operating in these spaces, due to their exceptional maneuverability. Although, their adoption for infrastructure inspection tasks is noteworthy, with numerous examples in the literature and industry from wind turbines and storage tanks (Voliro.com (2023)), to bridges (Ikeda et al. (2017)), their adoption for contact inspections in hazardous maritime environments, such as ballast tanks and cargo holds, has been limited. This research aims to address this need by developing a systematic framework for MRAV design optimization (theoretical and software), with a particular focus on enhancing operational efficiency and safety in the maritime industry.

1.1 Motivation

This study is driven by two main factors. The first being a noticeable gap in research concerning a systematic design procedure aimed at optimizing Multirotor Aerial Vehicles (MRAVs) tailored to tasks, for example contact inspections. Additionally, it seeks to address a practical challenge identified within the scope of the "AUTOASSESS (2024b)", a maritime Horizon Europe initiative.

Focusing on the second and the inspection of maritime assets, traditional inspection methods, rely on human surveyors navigating hazardous and confined spaces. The inspection of these areas, and more specifically ballast tanks, are not only costly (up to 1M per vessel) and time consuming, with an average inspection taking up to 15 days (some of them are traveling to far eastern low cost docks), but also pose significant risks to human life, with 1 person being killed on average per week due to the perilous conditions, according to IMO statistics. All these add up to a staggering 11B euros for the entire industry. On the other hand, a UAS-based inspection approach, like the one proposed by the new AUTOASSESS project (Co-funded by the European Union), offers a safer and more efficient alternative. Not only promises to address the economic factor, by reducing the, per vessel, inspection to 200.000 euros and cutting down the inspection time to 3 days, but more importantly save up to 50 lives per year while significantly reducing C02 emissions, ultimately saving over 9 B euros for the entire industry (see AUTOASSESS (2024a)).

Building on this, modern MRAV inspections present several advantages, particularly with their inherent safety aspect. By leveraging such a system equipped state-of-the-art sensors, one can eliminate the need for human surveyors in hazardous environments while significantly enhancing the accuracy, repeatability, and speed of vessel inspections. The interest in this topic is confirmed by a number of European Union projects. For example, the EU MINOAS project CORDIS (2017), focused, on providing a fully autonomous platform, with successful field tests performed in different types of vessels (see Bonnin-Pascual et al. (2019)). Moreover, the EU INCASS project CORDIS (2014) that concluded in 2017 which, among others, focused on the development of an aerial robotic tool for visual inspection of the inner hull of a vessel and thus laid solid foundation on autonomous drone maritime inspections.

However, these kind of tasks also come with their share of challenges. These include limitations in payload capacity that restrict the types of sensors or equipment that can be carried. Man-

euvering in small confined spaces, requires not only the small size of the MRAV but also the full actuation of it, which impacts negatively the energy efficiency of the system. This brings into the scope of disadvantages the limited battery life of such a system which restricts the duration of MRAV flights, requiring frequent recharging or battery swaps for prolonged inspections, potentially delaying or disrupting inspections.

This brings us to the second part of the motivation which stems from the fact that, these challenges extend beyond contact inspections in ballast tanks. To be more specific, different tasks, like visual and contact inspection, package delivery, and more, require different performance criteria, such as wrench generation, volume constraints, and energy efficiency. This necessitates optimized designs that accommodate specific task requirements, such as anisotropic force characteristics for generating wrench in preferred directions to complete the task. Although, literature extensively covers techniques for minimizing/maximizing various functions representing different drone aspects or requirements (to enhance their performance), there is a lack of a generalized, systematic, design procedure to transition from task requirements to a final MRAV configuration.

This research seeks to contribute and address the two aforementioned technical challenges: the development of an optimized MRAV for hazardous contact inspections in ballast tanks, and the creation of a generalized, systematic design procedure for transitioning task requirements into optimised MRAV configurations. By transitioning formal task requirements into a mathematical formulation and feeding it into a design optimization framework, optimized MRAV designs can be identified for various tasks, including contact inspection in ballast tanks. An endeavour like this requires adequate mathematical representation of task requirements, as well as a unified cost function[1], both of which are still evolving.

1.2 Objective & Research Questions

Deriving from the AUTOASSESS project (a Horizon Europe project) and against the aforementioned backdrops of the traditional methods, our project endeavors to propose a suitable Multirotor Aerial Vehicle (MRAVs), optimized for the task of autonomous contact-inspection on the walls of a ballast tank with a focus on detecting cracks and other structural issues, as well as derive a generalized, systematic, design procedure from task requirements to final MRAV configuration. The design procedure will follow the requirements of (but not limited to) the AUTOASSESS use case, which can be translated to the following task. The design of the MRAV must adhere to several constraints imposed by the task:

- The size of the MRAV must match the specifications of the ballast tank which in narrowest of cases is $40x60cm^2$.[2]
- The MRAV must have the minimum payload capacity required to carry inspection equipment and be able to push the hull to perform contact inspection (decided 6N).
- The MRAV must achieve a required minimum flight time to conduct thorough inspections (Because of the early stages of the project this is not specified, but considered as much as possible)

As baseline for the proposed design, will be tilted propeller MRAVs, optimized for contact-based tasks, like the "Tilt-Hex"[3]. However, tilted-propeller designs, often exhibit lower performance in terms of energy efficiency compared to standard aerial platforms due to the presence of internal forces. Therefore, the proposed framework will also aim to address these efficiency concerns while maintaining optimal performance for the contact-inspection task. All the aforementioned, lead to the following research questions that will guide this study:

[1] For instance, while different functions are employed to minimize energy consumption, there's potential to identify commonalities in these approaches and develop a generalized tool to address these.

[2] WidthxHeight

[3] A fixed-tilted propeller hexarotor, of the RAM lab, at the University of Twente.

With respect to the previous task requirements and constraints:

- What are possible metrics when using fixed-tilted MRAV designs ?

This question attempts to explore the performance metrics that need to be considered when designing MRAVs with fixed-tilted propellers. With the aforementioned backdrops of fixed-tilted designs in terms of endurance, it's crucial to define measurable parameters that can gauge these inefficiencies and provide a comparative baseline for improvements. Moreover, this analysis aim to provide a consistent basis for comparison between different MRAV configurations generated by the generalized software framework. By applying these metrics across various design solutions, it will be possible to evaluate and compare their performance in terms of energy efficiency, wrench generation etc. This comparative analysis will be critical for identifying the most optimal MRAV configurations and understanding how different design choices affect different performance aspects.

- What could be a generalised way, given the task requirements, to propose a design?

This question addresses one of the two primary goals of the project by aiming to develop a systematic procedure that starts from formal task-specific requirements (such as payload, endurance, and dimensions) and outputs an optimized design. A generalized approach is essential for creating adaptable MRAV designs that can be tailored to different tasks and environments. The focus on "generalization" ensures that the design framework can be applied beyond the scope of this project and offer broader industry applicability.

With respect to the force space of the MRAV:

- What is an appropriate mathematical representation of the task requirements ?

In this question, the emphasis is on developing a mathematical formulation for representing formal task requirements with the aim of creating a bridge between formal task needs and the physical capabilities of the system that can be optimised. A precise mathematical representation is not only crucial for accurate definition of constraints and objectives during the optimization process, but also allow to evaluate how well an optimized design fulfills the task's needs.

- Would redesigning the force space of the MRAV achieve the required improvements?

This question aims to determine if, or up to which extend, limitations in MRAV performance (based on the aforementioned task), can be overcome by wrench space redesign. The answer to this question will aid in evaluating the effectiveness of different configurations generated by the framework and may lead to innovations in MRAV architecture that enhance task performance without compromising on other aspects.

- What is a design procedure to redesign the force space?

The last question aims to provide a practical methodology for redesigning the wrench generation capabilities of an MRAV in order not only to account for each task's wrench needs, but to be used as a generalised design tool, upon which optimised designs will be based upon. The solution will primarily emerge from the software framework. By establishing a systematic procedure like that, we ensure that the MRAV performs effectively in real-world scenarios, addressing its adaptability, stability, and efficiency.

1.3 Methodology & Organisation

The methodology culminates in the development of a generalized optimization framework, aimed at designing optimised Multi-Rotor Aerial Vehicles (MRAVs) by addressing their formal task requirements, through their mathematical representation, inspired by the task requisites of the AUTOASSESS task. In this context, great importance is assigned to the adaptation of a diverse range of methodologies gathered from the appropriate literature and ensuring that they form a comprehensive groundwork for the optimization process. This union of optimization techniques promises to be the solution for the development of one single optimization framework, which is tuned to minimization and subject to design and task specifications.

To be more specific, by minimizing an objective function derived from a combination of modified, conflicting[4] cost functions from the literature, (and/or directly from the mathematical representation of the task itself), the software will solve the design problem at hand, as a constrained multivariable nonlinear program and ultimately yield an optimised design of an MRAV, tailored to the task at hand. A general framework sketch can be seen in Fig. 1.1. The tool developed in this way can also be effective over the whole range of MRAV systems, subject only to varying numbers of AUs.

Finally, the framework will be used to address the first of the two primary goals of this thesis, that of proposing a design for the AUTOASSESS project. Particular emphasis will be placed on accommodating size, payload capacity, and flight duration requirements, alongside the imperative consideration task of wrench generation capabilities. This rigorous validation process underscores the methodology's aptitude in addressing the specific demands of the AUTOASSESS initiative, while concurrently validating its broader applicability to diverse tasks.

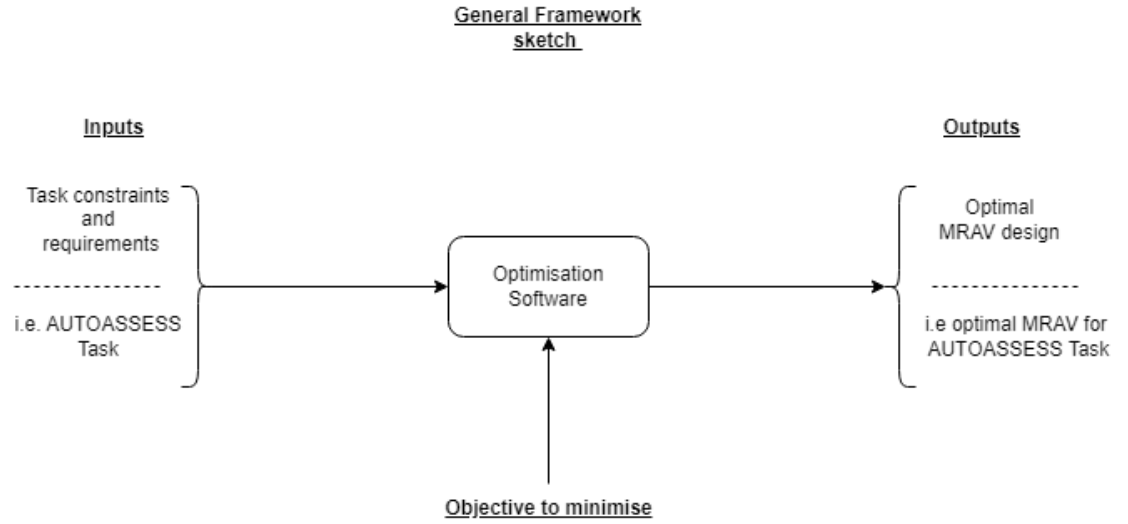


Figure 1.1: General Framework Sketch: Task as inputs through their mathematical formulation (Left). Objective to be minimised (Down) and Optimized Design as output (Right).

1.3.1 Related works & Background

The methodology begins with an extensive search of the literature to gather relevant theoretical concepts and methodologies in Subsection 1.4. This groundwork will support the development of a theoretical framework that will form the basis for the subsequent software framework. In Section 2, the necessary kinematic and dynamics of an MRAV system will be introduced upon which the theoretical and software framework was based, as well as the solver that will be used during the optimization, with all the necessary background knowledge.

1.3.2 Theoretical Framework Development

In Section 3, the mathematical formulation of the task requirements[5] and the metrics[6] will be presented, concluding with the derivation of the design problem at hand[7] and the NLP problem to be solved. All following theoretical concepts that were drawn from various disciplines,

[4] By balancing conflicting objectives we aim to ensure that the resulting MRAV designs are both efficient and effective for their intended tasks

[5] Research Question 1

[6] Research Question 3

[7] This is the mathematical formulation of the "Design Problem" we will try to solve, upon which the Optimization Software will be based upon.

including control theory, optimization techniques, and aerospace engineering, to ensure a comprehensive approach to MRAV optimization.

1.3.3 Software Framework Development

In Section 4, utilizing Matlab and CasADi (see Andersson et al. (2019)), the objective is to develop the optimization software framework based on the theoretical foundation established earlier. The framework will be designed to accommodate specific design and task prerequisites, ensuring an optimal MRAV design for each specific task. The first step will be to realize the optimization algorithm for the Tilt-Hex platform (which we consider as baseline) and depending on it's results will serve as benchmark for the designs to follow.

1.3.4 Simulation Environment Setup

Following the software's development, we proceed to construct, tune, and simulate the designs in Gazebo (Gazebo (2024)), enabling the testing and validation of these, under various operating conditions [8], providing insights into their performance and capabilities. The optimized parameters and configurations derived from the software's output, serve as inputs for the creation of the virtual MRAVs.

1.3.5 Experiments, Results and Test Case: AUTOASSESS Project

In section 5, experiments will be carried out within the simulated environment and in the real world. In the first case, we aim to evaluate the performance of the optimized MRAV designs, and in the second, to determine the thrust and drag coefficients of various propeller sets, ensuring the realism and reliability of the optimized designs, while paving the way to realise the, to be proposed, design. More specifically, we will try to replicate the AUTOASSESS conditions into Gazebo and also use realistic thrust and drag coefficients through coefficient identification experiment, with the goal of building the design. The discussion and analysis of the results includes the visualization of the wrench space, comparison with the benchmark results, and evaluation of the specific requirements (through, but not limited by, the aforementioned metrics 1.2) of the AUTOASSESS project.

1.3.6 Discussion, Conclusion and Future Work

Last but not least, in section 6, a discussion on each research question will be made, deriving from the results of the whole thesis and section 7 provides conclusion to the findings and contributions and offers recommendations for future research.

1.4 Related Work

Before proceeding, a literature review is essential to examine the current state of research in optimized MRAV design and control, outlining key contributions that can be used and identifying gaps that this study aim to address. This review reflects the author's understanding at the time of writing and acknowledges that some relevant work may not have been fully explored. It serves as an overview of related prior work, explaining why the current problem remains insufficiently addressed.

The existing literature contains a number of significant contributions on the design and control of MRAVs optimized for specific tasks. Specifically, Rajappa et al. (2015) examined the design and control of a fully-actuated hexarotor with tilted propellers, allowing for enhanced maneuverability in complex environments. The paper considers a fixed-tilted propeller design to achieve full actuation, whose tilt angles are defined by solving an optimization problem, that addresses several constrains with the goal of reducing the control effort over a trajectory and thereby increase the energy efficiency of the platform. While underactuated platforms struggle with physical interaction tasks, the proposed design addresses this underactuation problem and tries to provide a solution in increasing flight time. This approach increases flight time but is highly

[8] flight through narrow spaces, contact tasks, etc

dependent on the desired trajectory and does not account for task-specific requirements, like wrench generation. Although also very effective for 6D trajectory tracking, it lacks a generalized framework or mathematical representation to optimize MRAV configurations for diverse tasks, such as those with volume constraints.

Building on the need for more versatile MRAV designs, Park et al. (2018) introduced the ODAR (Omnidirectional Aerial Robot) platform. This platform enables omnidirectional wrench generation, making it ideal for performing complex manipulation tasks, although limited in its energy efficiency. The omnidirectional control capabilities and objective functions, described in this study could be adapted to enhance the current research in terms of generalisation of the framework and minimum guaranteed wrench generation of the output designs. It also gives a thorough perspective on weighting of the objective function elements, based on the specific task the authors wanted to achieve. Something like that is very desired as it can pave the way to connect the "task" to the "design". Despite the advantages of the ODAR platform, it remains tailored to specific manipulation tasks (mathematically formulated but specific to the current task) and thus does not provide a solution to the aforementioned literature "gap".

Addressing some of the limitations of previous designs, Hamandi et al. (2020) developed the Omni-Plus-Seven (O7+), an omnidirectional aerial prototype with a minimal number of unidirectional thrusters. They aim to present an optimization method to find tilt angles for any generic number of propellers ($n \geq 7$), that can guarantee the omnidirectional property of the platform, while enforcing equal thrust sharing between the propellers[9]. Moreover, the presented system simplifies the mechanical structure while also maintaining the ability of multidirectional wrench generation, making it suitable for manipulation tasks. The approaches and theory used for this design could be leveraged to form a generic design procedure that will not be constrained by the number of AUs (Actuator Units). Furthering the discussion on omnidirectional aerial vehicles, Brescianini and D'Andrea (2016) explored the design, modeling, and control of an omnidirectional aerial vehicle. The research provided valuable insights related to the static force and torque analysis for generic actuator configurations, out of which an optimal MRAV configuration was derived which maximized the vehicle's agility in any direction. However, both the aforementioned O7+ and this, do not provide a design procedure that ties task requirements to an optimised configuration.

The mechanical design and control of aerial manipulators have been duly studied by Nikou et al. (2015), with their focus being on optimizing the mechanical structure of MRAVs for specific manipulation tasks. These could be instrumental in developing robust MRAVs capable of carrying and operating specialized inspection tools without compromising flight stability. Their approach also has origins in technical optimization problems with large flexibility in the constraints, and therefore it could provide a valuable starting point for this research. However, their work does not take into account generalized task-oriented optimal designs, which is a very essential topic of this project.

Another relevant contribution is Rashad et al. (2017), which examined the design, modeling, and geometric control of an FA hexarotor optimized for aerial interaction tasks. Their "design problem", is formulated as a constrained optimization problem, with an objective function to be minimized, composed of components similar to the ones, the paper from Park et al. (2018)[10] uses. Although the research emphasizes the importance of precise control and stability for specific interaction tasks, it provides valuable insight on the formulation of a design problem, that can be exploited. However, it does not generalize to task-optimized MRAV designs.

[9]The attribute of equal sharing of thrusts is of paramount importance and seen in the literature, thus it can be explored as a metric of a good design.

[10]Based on the paper of cable-driven robots Bosscher et al. (2006)

A key theoretical foundation for MRAV design is provided by Hamandi et al. (2021b), which developed a taxonomy of multirotor aerial vehicles based on input allocation. This taxonomy is essential for determining the design and control complexities for various MRAV designs, when aiming to develop a generalized framework. Due to that, it can provide guidelines for the selection and optimization of inputs and design variables for the MRAVs in this project, based on the specific task requirements.

Continuing the exploration of omnidirectional MRAVs, Tognon and Franchi (2018) investigated into the theory and optimal design of MRAVs with unidirectional thrusters; a study that advances the theoretical understanding of MRAV control that and can be leveraged in formulating the theoretical framework of this research. Although it but does not provide a generalized design procedure, the principles of omnidirectional control can be adapted and integrated into this research.

The omnidirectional capabilities of generic multi-rotor aerial vehicles were also studied by Hamandi et al. (2021a), providing the necessary conditions for designing omnidirectional MRAVs, carefully considering factors such as geometry, weight, and actuation constraints. The study introduces several definitions and metrics for evaluation of omnidirectional capability. While this research does not directly address the primary goals of this thesis, it offers a procedure that can be effectively leveraged to achieve the thesis objectives, specifically in the areas of metric definition (as later be seen), wrench space analysis, and optimization techniques.

Strawson et al. (2021) focused on rotor orientation optimization. In this work, the author proposed a multi-objective optimization (MOO) algorithm to determine rotor tilt angles for a fixed-tilt configuration, based on desired frame parameters. These rotor orientation strategies could be incorporated into MRAV designs to enhance control precision during inspections. However, this research does not address task-oriented optimized designs.

Structural optimization has also been explored by Kiso et al. (2015), which focused on optimizing hexarotors for dynamic manipulability and maximum translational acceleration. Although this provides valuable insights into optimizing hexarotor performance, it does not contribute to a generalized design methodology that could be applied to other MRAV configurations. Nonetheless, structural optimization techniques could be adapted in this project, to ensure that the MRAVs are able to perform efficiently under specified tasks.

Bosscher et al. (2006) contributed to the understanding of wrench-feasible workspace generation for cable-driven robots, which can be applied to optimize the wrench capabilities of MRAVs. Although the study does not directly address MRAV design, the principles of wrench-feasible workspace generation could be used to ensure that the MRAVs developed in this project, can generate the necessary forces to effectively perform their tasks.

Energy efficiency is another critical aspect of MRAV design, as explored by Bauersfeld and Scaramuzza (2022), who investigated range, endurance, and optimal speed estimates for multicopters. This research's insights on energy efficiency could be leveraged not only as potential metrics of efficiency but also for optimizing the flight time and range of the MRAVs of this project.

The paper by Nava et al. (2020) contributes to the control tasks definition for aerial manipulators. Although these tasks are primarily focused on optimizing control, rather than the design of the MRAV itself, they can serve as guidelines in the design process. By addressing the structured approach to task definition provided in this paper, the MRAVs in the current project can be designed with a clear understanding of how they will need to perform in operational environments.

Xu and Saldaña (2023) explored the concept of finding optimal modular robots for aerial tasks, focusing on the adaptability and modularity of MRAVs. This research offers insights into how

MRAVs can be customized for different applications, gives task definitions and provide an insight on the wrench space analysis and optimisation techniques. However, it does not provide a generalized design procedure that can systematically produce optimal MRAV designs for a wide range of tasks, rather than optimal combinations of robots to achieve a task.

Ryll et al. (2016), highlights the innovative approach of a fully-actuated hexarotor with synchronized-tilting propellers. This design significantly enhances the maneuverability and control of MRAVs, making it particularly suitable for complex aerial tasks that require precise positioning and force application. The synchronized-tilting mechanism enables the hexarotor to exert forces and torques in multiple directions, offering superior control and the ability to perform complex maneuvers that are not feasible with traditional configurations. While the primary focus of the FAST-Hex paper is on optimizing control strategies rather than the overall design of MRAVs for specific tasks, the insights provided are highly relevant to the current project.

The analysis and synthesis of multi-rotor aerial vehicles by Jiang et al. (2011) provided further theoretical insights into MRAV dynamics and performance metrics. These insights could aid in the development of a systematic design framework, ensuring that the MRAVs are optimized for both performance and operational efficiency. However, like most of the existing literature, this study does not offer a practical framework that can be generalized across different MRAV applications.

With these and many more significant contributions, the existing literature on MRAV design and control has made progress in optimizing specific aspects of MRAV performance, such as control stability, energy efficiency, and manipulation capabilities. However, as this research aims to address, there lacks a systematic, unified framework, that directly ties the task requirements to design decisions and generates optimized MRAVs.

2 Background

Following the notation in designing and modelling of an MRAV as presented in Rajappa et al. (2015), this section will present the kinematics and dynamics upon which the software framework will be based upon, as well as the off-the-self solver Interior POint OPtimizer (IPOPT) that will be used to solve the NLP that will be formulated in the next sections. Lastly, a brief introduction to CasADi, the tool that will be used to bind all the framework together, will be made.

2.1 Design and Modeling of MRAV

2.1.1 Static System

In any multirotor aerial vehicle (MRAV) system, it is crucial to establish a set of coordinate frames to describe the position, orientation, and motion of the vehicle and its components. Here, we define three key frames of reference:

Inertial Frame (World Frame) (F_W):

$$F_W := \{O_W - X_W Y_W Z_W\}$$

The inertial frame is a fixed reference frame that represents the world coordinate system. All positions and motions are ultimately referenced to this frame, making it essential for describing the overall motion of the MRAV relative to the world.

Body Frame (F_B):

$$F_B := \{O_B - X_B Y_B Z_B\}$$

The body frame is attached to the MRAV's center of mass (denoted as O_B). It moves and rotates with the vehicle, providing a convenient reference frame for describing the vehicle's dynamics and the forces acting on it.

Propeller Frame (F_{P_i}):

$$F_{P_i} := \{O_{P_i} - X_{P_i} Y_{P_i} Z_{P_i}\}$$

Each propeller on the MRAV has its own coordinate frame, with the origin O_{P_i} located at the center of the i-th propeller. This frame is essential for describing the orientation and forces generated by each propeller.

We define the position of the i-th propeller in the body frame F_B as ${}^B\mathbf{p}_i \in \mathbb{R}^3$, where $i = 1, \ldots, n_p$ and n_p is the number of propellers. For simplicity, we assume that all propellers lie in the $X_B Y_B$ plane (co-planar), which is a common configuration in MRAV designs. The position in Body frame is given by:

$${}^B\mathbf{p}_i = \mathbf{R}_z(\lambda_i) \begin{bmatrix} l_{x_i} \\ 0 \\ 0 \end{bmatrix}, \quad \forall i = 1, \ldots, n_p \tag{1}$$

Here, $\mathbf{R}_z(\lambda_i)$ is the rotation matrix around the Z-axis, l_{x_i} is the length of the i-th arm (distance from O_B to O_{P_i}), and λ_i is the angle that the arm forms in the plane it lies on. This setup allows us to precisely describe the layout of the propellers on the MRAV, which is fundamental for modeling the forces and torques they generate.

To relate different frames of reference, we use the rotation matrices ${}^W\mathbf{R}_B \in SO(3)$,that transforms vectors from the body frame F_B to the world frame F_W and ${}^B\mathbf{R}_{P_i} \in SO(3)$,from the propeller frame F_{P_i} to the body frame F_B. Thus, the orientation of each propeller is parameterized by:

$${}^B\mathbf{R}_{P_i} = \mathbf{R}_Z(\lambda_i)\mathbf{R}_X(\alpha_i)\mathbf{R}_Y(\beta_i), \quad \forall i = 1, \ldots, n_p \tag{2}$$

Here, α_i and β_i are the tilt angles (radial and tangential), which are crucial for understanding how each propeller is oriented relative to the MRAV's body frame and then define tuples of these parameters as: $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{n_p})$, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_{n_p})$, $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_{n_p})$, $\mathbf{l_x} = (l_{x_1}, l_{x_2}, \ldots, l_{x_{n_p}})$. These tuples are essential for systematically describing the MRAV's configuration and will be used in the subsequent dynamic analysis.

2.1.2 Equations of Motion

The motion of the MRAV is governed by the principles of dynamics, which here are described using the Newton-Euler approach. This approach combines translational and rotational dynamics to model the full motion of the MRAV.

Rotational Dynamics

The rotational dynamics describe how the MRAV's orientation changes over time due to applied torques. Let $\boldsymbol{\omega}_B \in \mathbb{R}^3$ represent the angular velocity of the body frame F_B with respect to the inertial frame F_W, expressed in F_B. The rotational dynamics are given by:

$$\mathbf{J}_B \dot{\boldsymbol{\omega}}_B = -\boldsymbol{\omega}_B \times \mathbf{J}_B \boldsymbol{\omega}_B + \boldsymbol{\tau} + \boldsymbol{\tau}_{ext} \tag{3}$$

Here, $\mathbf{J}_B$ is the body inertia matrix of the MRAV, capturing its resistance to rotational motion. The term $\boldsymbol{\tau}$ represents the torques generated by the propellers, while $\boldsymbol{\tau}_{ext}$ accounts for external disturbances and unmodeled effects.

The input torque $\boldsymbol{\tau}$ is composed by the thrust and the drag torque. The former is the result of forces generated by the propellers and is written as:

$$\boldsymbol{\tau}_{\text{thrust}} = \sum_{i=1}^{n_p} \left({}^B\mathbf{p}_i \times {}^B\mathbf{R}_{P_i} \mathbf{T}_{\text{thrust}_i} \right) \tag{4}$$

with the thrust generated by the i-th propeller can be expressed as:

$$\mathbf{T}_{\text{thrust}_i} = \begin{bmatrix} 0 \\ 0 \\ c_f \tilde{\omega}_i^2 \end{bmatrix} \tag{5}$$

The coefficient of thrust is c_f and $\tilde{\omega}_i$ is the angular velocity of the i-th propeller. The drag torque, opposes the rotational motion of the propellers and is written as:

$$\boldsymbol{\tau}_{\text{drag}} = \sum_{i=1}^{n_p} \left({}^B\mathbf{R}_{P_i} \mathbf{T}_{\text{drag}_i} \right) \tag{6}$$

with the drag moment generated by the i-th propeller to be:

$$\mathbf{T}_{\text{drag}_i} = \begin{bmatrix} 0 \\ 0 \\ (-1)^i c_t \tilde{\omega}_i^2 \end{bmatrix} \tag{7}$$

and c_t representing the drag coefficient. Finally, the total input torque can be expressed as:

$$\boldsymbol{\tau} = \mathbf{F_1}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \mathbf{l})\mathbf{u} \tag{8}$$

with $\mathbf{F_1} \in \mathbb{R}^{3\times 6}$ a matrix relating the input torque to the control input vector $\mathbf{u}$, defined as:

$$\mathbf{u} = \begin{bmatrix} \tilde{\omega}_1^2 \\ \tilde{\omega}_2^2 \\ \vdots \\ \tilde{\omega}_n^2 \end{bmatrix} \in \mathbb{R}^n \tag{9}$$

This expression allows us to systematically analyze how different configurations and control inputs affect the MRAV's rotational dynamics.

Translational Dynamics

The translational dynamics describe how the MRAV's position changes over time due to applied forces. Using the Newton-Euler formulation, the translational dynamics with respect to the world frame are given by:

$$m\ddot{\mathbf{p}} = -mg\hat{\mathbf{z}} + {}^{W}\mathbf{R}_{B}\mathbf{F_2}\mathbf{u} + \mathbf{f}_{ext} \tag{10}$$

Here, m is the mass of the MRAV, $\ddot{\mathbf{p}}$ is its linear acceleration in inertial frame, g is the acceleration due to gravity, and $\hat{\mathbf{z}}$ is the unit vector in the Z-direction. The matrix $\mathbf{F_2}$ represents the contribution of the propeller thrusts to the translational dynamics, where:

$$\mathbf{F_2}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda})\mathbf{u} = \sum_{i=1}^{n_p} \left({}^{B}\mathbf{R}_{P_i}\mathbf{T}_{\text{thrust}_i} \right) \tag{11}$$

This equation models how the thrust forces generated by the propellers contribute to the MRAV's overall translational motion.

Last but not least for the Control Design the same feedback linearization and decoupling control technique is used, with a P.I.D. controller, as described in Rajappa et al. (2015).

2.2 Interior Point OPTimizer (IPOPT)

Interior Point OPTimizer (Github.io (2020)) is a software package intended for large-scale nonlinear optimization, where both the objective function and constraints can be nonlinear. The following section is written following the work of Wächter and Biegler (2006). The mathematical optimization problems that solves are written as:

$$\begin{aligned} &\min f(x) \\ &\text{subject to } g(x) = 0 \\ &h(x) \leq 0 \end{aligned}$$

where $f(x)$ is the objective function, while $g(x)$ and $h(x)$ are addressing sets of equality and inequality constraints. The solver implements interior point line search filter method[1], that aims to find a local solution of NLP and leverages a primal-dual algorithm to find solutions. Unlike algorithms that work along the boundary of the feasible region, interior-point methods explore the inside of the feasible region to find optimal solutions. More specifically, in IPOPT, the algorithm iteratively approaches the optimal solution by solving a sequence of approximate subproblems. At each iteration, the method uses a barrier function, which introduces a barrier term to the objective function, to ensure that the iterations remain within the feasible region. The larger the barrier term, the closer to the boundary of the feasible region we are, thus preventing iterates from leaving the feasible region.

2.2.1 Primal-Dual Algorithm

The primal-dual algorithm used in IPOPT is designed to solve the Karush-Kuhn-Tucker (KKT) conditions, which are necessary conditions for optimality in nonlinear programming problems. The KKT conditions for the original problem are:

$$\begin{aligned} &\nabla f(x) + \nabla g(x)\lambda + \nabla h(x)\nu = 0 \\ &g(x) = 0 \\ &h(x) + s = 0 \\ &S\nu = 0 \\ &s > 0, \quad \nu > 0 \end{aligned}$$

[1] Class of algorithms for solving linear and nonlinear convex optimization problems.

with λ and v being the Lagrange multipliers of the aforementioned constraints and S a diagonal matrix of slack variables. For solving these conditions, a Newton-type method is employed by the IPOPT, where in each iteration, a linear system derived from the KKT, is solved to find the search direction for the primal x and dual variables λ. Using a line-search strategy, the solution involves updating the aforementioned variables, as well as the slack variables s to ensure convergence.

2.2.2 Line-Search Filter Method

The line-search filter method used by the solver aims to avoid the common issues with traditional merit functions by using, as the name of the method proposes, "a filter". The "filter" is nothing more than a set of criteria, used to accept or reject iterates based on feasibility and optimality, without aggregating these two measures into a single merit function. This method maintains a list of pairs $(h(x), f(x))$, representing "constraint violation" and "objective function value", which simply accept the new iterates, if these reduce either "constraint violation" or the "objective function", without worsening the other.

The steps of the primal-dual interior-point method in IPOPT can be written as:

1. **Initialization:** Start with an initial guess for x, λ, v, and s that satisfies the constraints $s > 0$ and $v > 0$.
2. **Barrier Problem Solution:** Solve the barrier problem using the current value of the barrier parameter μ.
3. **Newton's Method:** Apply a Newton-type method for solving the linearized KKT conditions to find the search direction for x, λ, v, and s.
4. **Line Search:** Using the filter method, ensure that the new iterates improve the objective function or feasibility without overall regression.
5. **Update:** Update the barrier parameter μ and iterate until convergence is found.

In general, the solver is known for its robustness and ability to find feasible solutions in complex and tightly constrained problems. Having said that, the setup and configuration of IPOPT can sometimes be complex, especially for users not familiar with nonlinear optimization. That was one of the reasons, we chose to interface IPOPT with CasADi as we will see also in the next chapters.

2.2.3 CasADi

CasADi (see Andersson et al. (2019)) is an open-source software framework designed for nonlinear optimization and optimal control, particularly suited for problems involving dynamic systems. At its core, CasADi provides a symbolic framework that enables the efficient formulation and solution of complex optimization problems. One of the key features of CasADi is its support for automatic differentiation, which allows for the exact computation of derivatives, a critical component in optimization algorithms. This capability is particularly valuable in the context of constrained multivariable optimization, where the accuracy and efficiency of derivative calculations directly impact the convergence and performance of the solver.

CasADi's integration with advanced and off-the-self solvers, such as the aforementioned IPOPT, further enhances its capability to tackle large-scale nonlinear optimization problems and one of the reasons that we chose to work with it. These solvers are optimized for performance and are adept at handling complex constraints and large variable sets, such as in our case. Additionally, CasADi is designed to exploit sparsity in matrices—a common characteristic of large-scale optimization problems—thereby improving both computational speed and memory efficiency (Casadi.org (2018)).

3 Theoretical Framework

In this chapter, the theoretical cornerstone of the software framework will be derived. Starting by defining a formal generic Task and continue to the mathematical description of what is considered as MRAV design, the mathematical formulation of a task's requirements will be presented, followed by the general Design Problem to be solved. Moreover, the metrics that will be used to evaluate the performance and optimality of the MRAVs and the NLP problem to be solved, by the software framework, will be presented.

3.1 Mathematical Formulation of Task Requirements and Design Problem

Inspired by the AUTOASSESS case study, we will define the formal task requirements and constraints (not mathematical) as:

$$Task^1 \begin{cases} \text{Requirements : } \mathbf{w}_{des} \text{ (N), Endurance (s)} \\ \text{Constraints: Volume (size)} \end{cases}$$

with the objective to find an optimised design $\boldsymbol{\theta}$ (or more), that satisfies them. Let us denote the design optimization variables by the tuple $\mathbf{s}$ and the task by $\mathbf{T} = (\mathbf{T_1}, \mathbf{T_2}, .., \mathbf{T_n})$. We consider as in Tognon and Franchi (2018), an MRAV design, the tuple: $\boldsymbol{\theta} = (n, \mathbf{c}, \mathbf{k}, {}^B\mathbf{p}_i, {}^B\mathbf{v}_i)$, with, $c_i = \{-1, 1\}$, denoting CCW or CW rotation, $k = \frac{c_t}{c_f}$ and ${}^B\mathbf{v}_i = {}^B\mathbf{R}_{P_i}\, {}^{P_i}\mathbf{z}$. In the subsections to be followed, we will mathematically represent the aforementioned formal task requirements, afterwards we will derive the design optimisation variables and lastly formulate the design problem. Following the idea of solving an optimisation problem (for optimised designs), we aim to translate the formal task requirements, into task constraints or members of the cost function and thus we will have in our hand a constrained optimisation problem to solve.

3.1.1 Mathematical representation of the formal Task requirements and constraints

Mathematical Formulation of Task Volume constraint:

Given the dimentional "Volume constraints", we use the idea of an "Occupancy Box[2]" of $L \times W \times H(m^3)$, which represents the 3D boundaries that encapsulate a design $\boldsymbol{\theta}$. We separate and write the total width (diameter) of the robot in the y-direction as $d_y \in \mathbb{R}_+$. Subsequently, we write $d_y = 2r_y$, where $r_y \in \mathbb{R}_+$ is the maximum radius of the robot in the y-direction.

Now, let the radius r_y in the y-direction be expressed as:

$$r_y = \max_i \left(\left| {}^B\mathbf{p}_i \cdot \hat{\mathbf{y}} \right| + r_p \right), \tag{12}$$

where ${}^B\mathbf{p}_i$ is the position of the i-th motor in the body frame and $r_p \in \mathbb{R}_+$ is the radius of the propeller. We follow the same procedure for the "Length" on the x-axis. A design choice we made for simplicity of the designs is to work with co-planar designs, thus the "Height" task constraint can directly be compared with the height of the MRAV on the z-axis.

We name as set of "Acceptable Occupancy Boxes", the set $\mathbb{V}_T$ that can be described as the set of all points $(d_x, d_y, d_z) \in \mathbb{R}^3_+$. These points in the 3D space, are representing the valid 3D "Occupancy Boxes" dimensions (diameters) of the designs, that fit within the task's dimensional constraints and we write the mathematical formulation as:

$$\mathbb{V}_T = \left\{ (d_x, d_y, d_z) \in \mathbb{R}^3_+ \mid 0 < d_x \leq L,\ 0 < d_y \leq W,\ 0 < d_z \leq H \right\}, \tag{13}$$

[1] Here, "requirements" and "constraints" are used as formal descriptors, which will be mathematically formulated as components of the optimization problem, either as constraints or as part of the cost function.

[2] A 3D construct.

where $\{L, W, H\} \in \mathbb{R}_+$ are the maximum length, width and height requirements of the task. For example during a task the system needs to navigate through openings (i.e. doors) of maximum 20x50cm^2 (WxH). Now let $\mathbb{O}(\boldsymbol{\theta})$ represent the "Occupancy Box" of a design $\boldsymbol{\theta}$, where $\mathbb{O}(\boldsymbol{\theta}) = (d_x(\boldsymbol{\theta}), d_y(\boldsymbol{\theta}), d_z(\boldsymbol{\theta})) \in \mathbb{R}^3_+$. Thus a design $\boldsymbol{\theta}$ must satisfy the aforementioned Task Volume constraint or $\mathbb{O}(\boldsymbol{\theta}) \in \mathbb{V}_T$.

Mathematical representation of the formal Task's Wrench requirement:
Given the inputs $\mathbf{u}$, that lies on $\mathbb{A} \subset \mathbb{R}^n$, and the design $\boldsymbol{\theta}$, that lives on $\mathbb{P} \subset \mathbb{R}^m$ we denote as $\mathbf{w}$, the generated wrench for which holds:

$$\mathbf{w} : \mathbb{A} \times \mathbb{P} \rightarrow \mathbb{R}^6,$$

$$(\mathbf{u}, \boldsymbol{\theta}) \mapsto \mathbf{w}(\mathbf{u}, \boldsymbol{\theta})$$

Also, $\mathbf{w} = \mathbf{F}\mathbf{u}$, we call $\mathbf{F} \in \mathbb{R}^{6\times n}$ the Allocation matrix of the design $\boldsymbol{\theta}$. Deriving from the aforementioned, for a certain (given) design $\boldsymbol{\theta}' \in \mathbb{P}$, we define the set of wrenches that can be generated from that certain design $\boldsymbol{\theta}'$ as:

$$\Omega(\boldsymbol{\theta}') = \{\mathbf{w}^* \in \mathbb{R}^6 \mid \exists \mathbf{u} \in \mathbb{A},\ \mathbf{F}(\boldsymbol{\theta}')\mathbf{u} = \mathbf{w}^*\} \tag{14}$$

For the general MRAV case (although we will work mostly with "hexarotors"):

$$rank(\mathbf{F}(\boldsymbol{\theta})) \begin{cases} < n, & \text{No real solution} \\ = n, & \text{Unique solution } \mathbf{u} = \mathbf{F}^{-1}\mathbf{w}\text{, injective map (specific } \mathbf{u} \text{ corresponds to specific } \mathbf{w}) \\ > n, & \text{Infinitely many solutions} \end{cases}$$

For the inverse image (thus the set of inputs) given a $\mathbf{w} \in \Omega(\boldsymbol{\theta}')$, then the inverse image is defined as (also seen in Fig.3.1) :

$$\Omega(\boldsymbol{\theta}', \mathbf{w})^{-1} = \{\mathbf{u} \in \mathbb{A} \mid \mathbf{w}(\mathbf{u}, \boldsymbol{\theta}') = \mathbf{w}\} \tag{15}$$

For the requirement of "generation of wrench", we denote $\mathbb{W}_T = \{\mathbf{w}_{T_1}, ..., \mathbf{w}_{T_n}\}$, the set of desirable wrenches required to be generated from a design $\boldsymbol{\theta}$. This set can be finite or infinite (theoretically). Thus, we opt to find a design $\boldsymbol{\theta}$, with $\Omega(\boldsymbol{\theta})$, such that $\mathbb{W}_T \subset \Omega(\boldsymbol{\theta})$. With all the aforementioned in mind, let us examine the "wrench space" of a general unidirectional thrust MRAV case, as seen in Fig. 3.1.

The input vector $\mathbf{u}$ is bounded due to motor limits. Thus $u_{min} \preceq \mathbf{u} \preceq u_{max}, \forall \mathbf{u} \in \mathbb{A}$ (elementwise). As presented in Xu and Saldaña (2023), this defines a half-space, with the boundaries of $\mathbb{A}$ being the hyper-planes of $\mathbf{u}_i = u_{max}\hat{\mathbf{i}}$. For n-dimentional inputs, we have 2 x n half-spaces. These 2 x n half-spaces define the boundaries of the feasible region of the 6D input space. The intersection of these half-spaces forms a bounded convex polytope, which represents the feasible region for the vector $\mathbf{u}$ (in other words $\mathbb{A}$).

By mapping the input polytope $\mathbb{A}$ with an affine function $\mathbf{F}$ to the $\mathbb{R}^6$ wrench space, we obtain feasible wrenches $\Omega(\boldsymbol{\theta})$. Thus $\Omega(\boldsymbol{\theta})$ is a zonotope. Moreover, this zonotope is constructed by generator vectors that are nothing more than the column vectors of $\mathbf{F}(\boldsymbol{\theta})$. The representation of a zonotope with only its generators, allows for the generation of the convex hull directly from $\mathbf{F}$, since the convex hull of a convex polytope (a zonotope) is the same polytope.

The aforementioned, allows us to check if wrenches $\mathbf{w}$ belongs to it. The $\Omega(\boldsymbol{\theta})$ set is infinite, but a zonotope is not due to its convexity. In this way we can compare the maximum magnitude on the boundary point of the wrench space in the direction of $\mathbf{w}_{T_i}$! This can be achieved (both the representation of the force and torque space that is of paramount importance to our software framework and the comparison) if the wrench set is convex. For this case of unidirectional thruster MRAV designs, we made a proof of convexity of the wrench set in Appendix A.

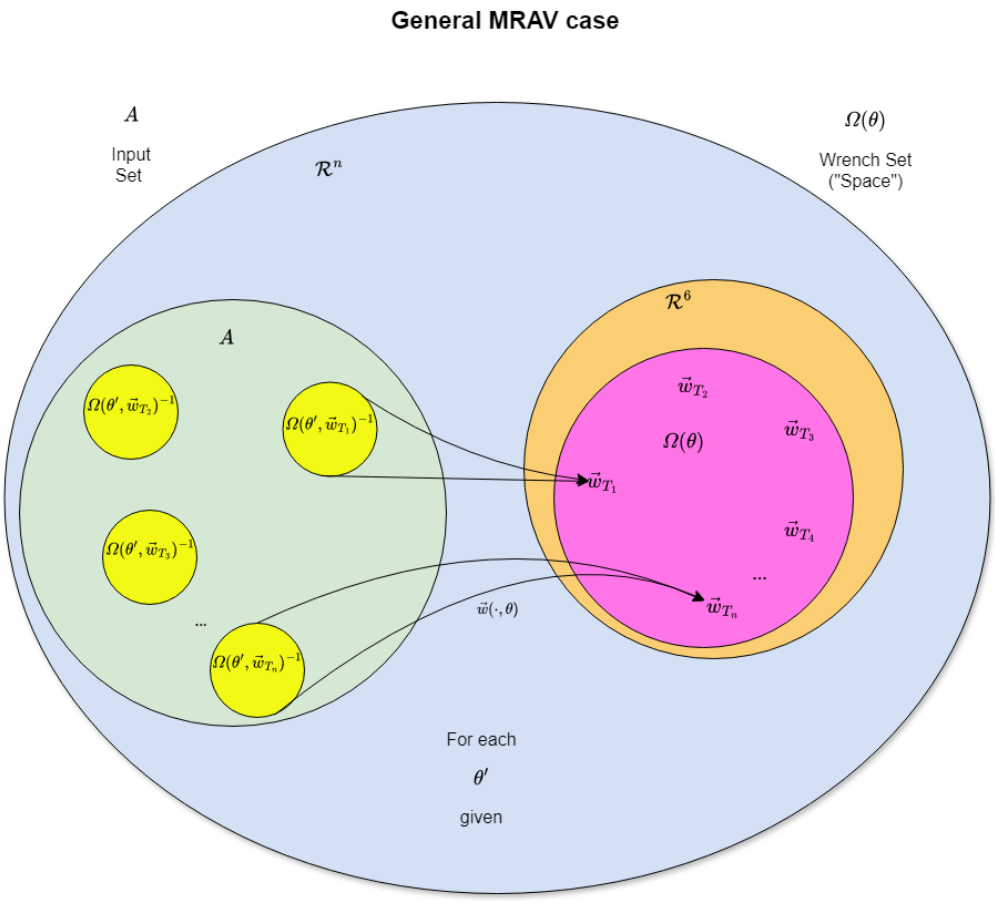


Figure 3.1: General unidir. MRAV case were $\Omega(\boldsymbol{\theta}', \mathbf{w})^{-1}$ can be unique or not. $\mathbf{w}_{T_i}$ are the task's desirable wrenches. For rank{**F**}> n, a desired wrench can correspond to a set of inputs (not one necessary), for a certain design, due to redundancy.

Let us consider a typical hexarotor with $n = 6$ and a unique solution. Thus we say that $\mathbf{w}_{T_i}$ belongs to $\Omega(\boldsymbol{\theta})$ iff $\mathbf{w}_{T_i} = \|\mathbf{w}_{T_i}\| \cdot \hat{\mathbf{w}}_{T_i} \preceq w_{max} \cdot \hat{\mathbf{w}}_{T_i}$, and thus $\mathbb{W}_T \subset \Omega(\boldsymbol{\theta})$. In the same way for the image, $\exists \mathbf{u} \in \mathbb{A}$ that,

$$u_{min} \preceq \mathbf{u} \preceq u_{max} \implies \tag{16}$$

$$u_{min} \preceq \Omega(\boldsymbol{\theta}, \mathbf{w}_{T_i})^{-1} \preceq u_{max} \implies \tag{17}$$

$$u_{min} \preceq \mathbf{F}(\boldsymbol{\theta})^{-1} \cdot \mathbf{w}_{T_i} \preceq u_{max} \tag{18}$$

Thus, for a $\mathbf{w}_{T_i}$ to be able to be generated from a design $\boldsymbol{\theta}$, there has to $\exists \mathbf{u} \in \mathbb{A} \mid u_{min} \preceq \mathbf{F}(\boldsymbol{\theta})^{-1} \cdot \mathbf{w}_{T_i} \preceq u_{max}$. We check for every element in $\mathbb{W}_T$, to determine if $\mathbb{W}_T \subset \Omega(\boldsymbol{\theta})$.

In the case where the number of inputs $n > 6$ or the rank of the allocation matrix $\mathbf{F}(\boldsymbol{\theta})$ is greater than 6, we encounter an underdetermined system. This implies that there are infinitely many solutions to the equation:

$$\mathbf{w}_{\text{des}} = \mathbf{F}(\boldsymbol{\theta}) \cdot \mathbf{u} \tag{19}$$

However, the framework can still be applied in this scenario by assuming that the input allocation strategy employed in the controller relies on the pseudoinverse of the allocation matrix, $\mathbf{F}(\boldsymbol{\theta})^{\dagger}$. The pseudoinverse provides the minimum norm solution, which can be expressed as:

$$u_{\text{min}} \preceq \mathbf{F}(\boldsymbol{\theta})^{\dagger} \cdot \mathbf{w}_{\text{des}} \preceq u_{\text{max}} \tag{20}$$

Thus, for any desired wrench $\mathbf{w}_{\text{des}}$ to be achievable with the design $\boldsymbol{\theta}$, there must exist a corresponding input $\mathbf{u} \in \mathbb{A}$ such that:

$$u_{\text{min}} \preceq \mathbf{F}(\boldsymbol{\theta})^{\dagger} \cdot \mathbf{w}_{T_i} \preceq u_{\text{max}} \tag{21}$$

We verify this condition for every element in $\mathbb{W}_T$, ensuring that $\mathbb{W}_T \subset \Omega(\boldsymbol{\theta})$.

Mathematical Representation of the formal Task's Requirement of Endurance:
We can approach the requirement of endurance as a consequence of minimising the loss of energy within the system and/or by minimising the norm of the input, over a maneuver (or trajectory) of the task. In other words we choose some of the components of the objective function that addresses the energy requirement. In order to do that, we first define the general cost of a given input, $J(\mathbf{u}) : A \rightarrow \mathbb{R}$ and the general cost of a given wrench $\mathbf{w}' \in \Omega(\boldsymbol{\theta})$ as (see Fig. 3.2):

$$C(\boldsymbol{\theta}, \mathbf{w}') = \min_{u \in \Omega(\boldsymbol{\theta}', \mathbf{w})^{-1}} J(\mathbf{u}) \tag{22}$$

Keeping these in mind, we can formulate the components in the cost function that will address the formal endurance requirement.

- Component 1 : Minimisation of Internal Forces

For each propeller i, it can be derived:

$$^B\mathbf{T}_i = c_f \cdot u_i \cdot {}^B\mathbf{v}_i = c_f \cdot u_i \cdot {}^B\mathbf{R}_{P_i} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \implies {}^B\mathbf{T}_i = c_f \cdot u_i \cdot \mathbf{R}_z(\lambda_i)\mathbf{R}_x(\alpha_i)\mathbf{R}_y(\beta_i) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{23}$$

We opt for minimising the loss of energy due to internal forces on the body frame, or maximise the generation of thrust in the z direction of the body frame, by optimising for $\boldsymbol{\alpha}, \boldsymbol{\beta}$. Thus we aim to find the solution to $\min_{\boldsymbol{\alpha}, \boldsymbol{\beta}} [Z(\vec{u})]$, with

$$Z(\mathbf{u}) = -\left\| \sum_{i=1}^{n} {}^B\mathbf{T}_i(\alpha_i, \beta_i) \cdot \hat{z} \right\|_2 \tag{24}$$

- Component 2: Minimisation of Control Effort

Inspired by the work of Rajappa et al. (2015), this component minimizes the control effort based on the desired trajectory. This is of paramount importance for a generalised procedure, since the trajectory or manuevers play a crucial role for the task and energy efficiency of the MRAV. Thus, we aim to find the solution to $\min_{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}} [U(\mathbf{u})]$ with,

$$U(\mathbf{u}) = \int \|\mathbf{u}(t)\|_2 \, dt \tag{25}$$

and subject to constraints. Thus we can address the requirement of the flight time as components of a larger objective function, indicated as J for now.

The previous analysis of Section 3.1.1, directly addressed the third research question of the appropriate mathematical representation of task requirements, through representing the formal task requirements in the form of constraints and objective function components.

3.1.2 Selection of design variables

As optimization variables, we chose to optimize for the tilt angles and the position of the motors ($\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}$), because, intuitively, the first primarily influence endurance and wrench generation, while the latter affect the dimensions (hence the volume). It is imperative though to mathematically show that our design variables are such that they have an effect on the mathematically represented task requirements (on the Task). For the aforementioned set of tasks: $\mathbf{T} = \{\mathbf{T_1}, ..., \mathbf{T_m}\}$, with $\mathbf{T_i}$ a tuple of sets, with i=1,...,m, we proved in Appendix B.1 that $\nabla \mathbf{T} \neq 0$ for the respective optimization variables and thus:

$$\mathbf{T_i}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}) = (\mathbb{V}_{T_i}(\boldsymbol{\lambda}, \boldsymbol{l_x}), \mathbb{W}_{T_i}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x})) \tag{26}$$

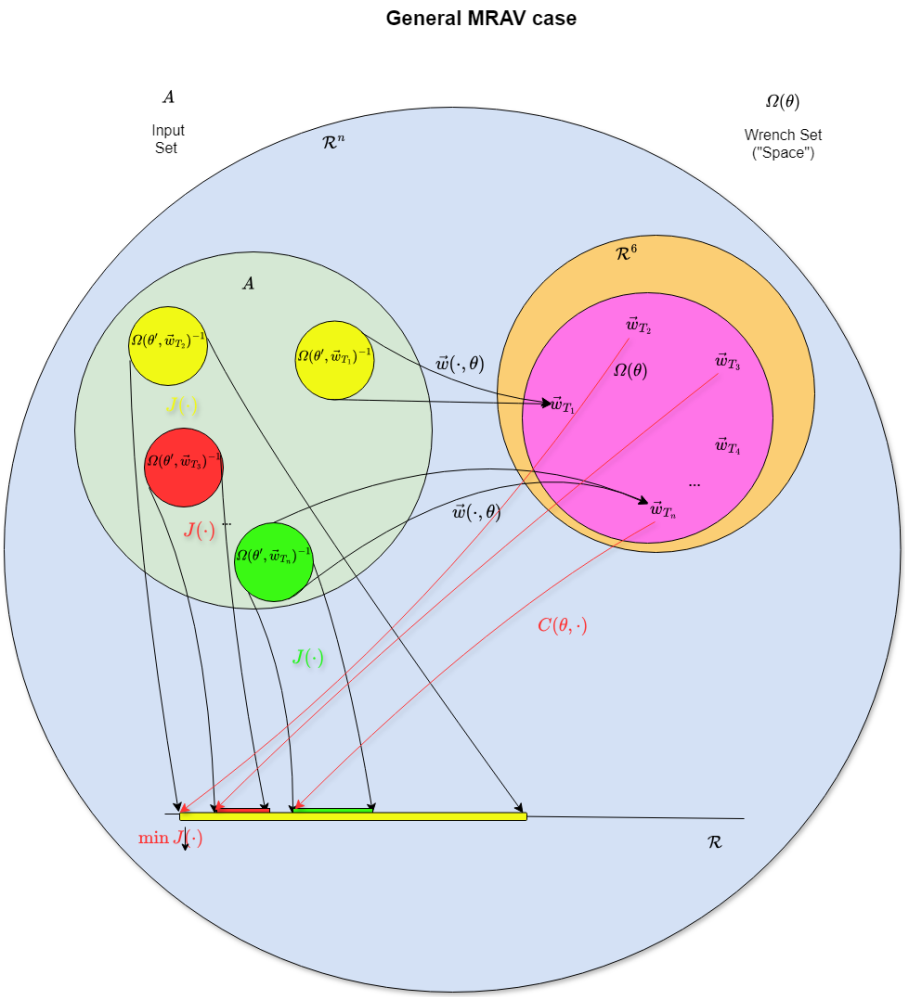


Figure 3.2: General unidir. MRAV case with the cost of a given input J.

3.1.3 Design Problem

Finally, after the mathematical representation of a task's requirements (inspired by the task of AUTOASSESS, but not limited), we are ready to derive the design problem, that the software will try to solve.

Let a task of interest or a set of tasks of interest be given, defined by: $\mathbf{T} = \{\mathbf{T_1}, ..., \mathbf{T_m}\}$, where $\mathbf{T_i}$ tuple of sets, with i=1,...,m:

$$\mathbf{T_i} = (\mathbb{V}_{T_i}, \mathbb{W}_{T_i})$$

1. **Existence Problem:** Find a design $\tilde{\boldsymbol{\theta}} \in \mathbb{P}$, or a set of designs, if it exists, such that
$$\begin{cases} \mathbb{O}(\tilde{\boldsymbol{\theta}}) \in \mathbb{V}_{T_i} \\ \qquad\qquad \text{and denote with } \mathbb{E}(T) \subset \mathbb{P}, \text{ the set of all solutions of 1.} \\ \mathbb{W}_{T_i} \subset \Omega(\tilde{\boldsymbol{\theta}}) \end{cases}$$
2. **Optimal Problem:** Find the solution to
$$\min_{\boldsymbol{\theta} \in \mathbb{E}(T)} J(\mathbf{u})$$

.

3.2 Optimization Metrics

The evaluation metrics used to assess the output design were carefully chosen to create a robust framework for optimization and validation, drawing inspiration from the needs of the

AUTOASSESS project, also incorporating metrics proposed in the literature. Each metric was chosen to serve a distinct purpose, by shining light on the system's endurance, maneuverability, stability, wrench generation and feasibility. Moreover, as will later be shown, because the problem will be solved as a nonlinear optimization problem [3], these metrics will collectively aid in informing the user about the design's optimality level concerning the specified tasks of interest. This systematic approach allows direct comparison of the output design with other optimal designs (local solutions), ensuring a thorough and objective evaluation process. This section aims to address the first research question (see 1.2).

3.2.1 Full Actuation

Ensuring the full actuation of the MRAV is critical for the ability to control all six degrees of freedom (DOF). By verifying that the rank of the combined force and torque matrices is at least 6 ($rank[F,H] \geq 6$), the framework guarantees that the MRAV can precisely maneuver, perform interaction tasks and maintain stability in complex and constraint environments, such as the one of the AUTOASSESS. This metric is also passed as a constraint in the constraint optimisation problem, later to be solved.

3.2.2 Force Efficiency Index

The force efficiency index, based on the method shown in Ryll et al. (2016), evaluates how effectively the hexarotor utilizes its generated forces or, in other words, it is an indication of how present are the internal forces in the system. An index value close to 1 indicates high efficiency, suggesting that the hexarotor is using its thrust capabilities optimally (it would correspond to all the thrust vectors pointing in the z direction). This metric is crucial for assessing the overall efficiency of the propulsion system.

$$\eta_f(\alpha,\mathbf{u}) = \frac{\left\|\sum_{i=1}^{6} f_i^B(f_i,\alpha)\right\|}{\sum_{i=1}^{6}\left\|f_i^B(f_i,\alpha)\right\|} = \frac{\left\|\sum_{i=1}^{6}\frac{f_i^B(f_i,\alpha)}{f_i}\right\|}{\sum_{i=1}^{6} f_i} \in [0,1] \tag{27}$$

3.2.3 Condition Number

The condition number of the Allocation Matrix is an important metric for assessing the numerical stability of the control system. A low condition number indicates a well-conditioned system, meaning the control inputs will be reliable and accurate or that small changes in the input do not result in large changes in the output. This metric helps in ensuring that the hexarotor's control system is robust against numerical errors and perturbations and can be found as a metric of balanced MRAVs or even component of objective functions to be minimised by optimizers, in various papers in the literature such as Tognon and Franchi (2018). As mentioned in the mathematical formulation (see Section 3.1.1), let the set of wrenches of a design $\boldsymbol{\theta}$ be described as in 14 we write the set as in Hamandi et al. (2020)

$$\Omega(\boldsymbol{\theta}) = \{\mathbf{w} \in \mathbb{R}^6 \mid \mathbf{w}^\top \boldsymbol{\Sigma}\mathbf{w} \leq 1\} \subset \mathbb{R}^6 \tag{28}$$

where $\Sigma \in \mathbb{R}^{6\times 6}$ is a positive definite matrix. The ellipsoid ($\Omega(\boldsymbol{\theta})$) is mapped by $\mathbf{F}^{\dagger}$[4] to the set

$$\Omega(\boldsymbol{\theta},\mathbf{w})^{-1} = \{\mathbf{u} \in \mathbb{A} \mid \mathbf{u} = \mathbf{F}^{\dagger}\mathbf{w}, \forall \mathbf{w} \in \Omega(\boldsymbol{\theta})\} \tag{29}$$

which is a 6-dimensional ellipsoid of $\mathbb{R}^n$, contained in the subspace $\mathrm{im}(\mathbf{F}^\top)$, whose shape is defined by the singular value decomposition of $\mathbf{F}$ and $\boldsymbol{\Sigma}$. According to Tognon and Franchi (2018), definition 3, "a design is optimal if the max input $\mathbf{u} \in \Omega(\boldsymbol{\theta},\mathbf{w})^{-1}$ is minimized, where $\Omega(\boldsymbol{\theta},\mathbf{w})^{-1}$ is the set of inputs that the given allocation strategy maps to $\Omega(\boldsymbol{\theta})$." Thus, it is of great essence the eccentricity of the input set to be small (or minimized), or in other words the condition number $\boldsymbol{\Sigma}^{-1}\mathbf{F}$.[5]

[3] mainly due to the presence of nonlinear objective function

[4] Moore-Penrose inverse (pseudoinverse).

[5] In case of more than 6 inputs, then one can follow the steps of Tognon and Franchi (2018) for a balanced design.

The metric of the condition number should be examined carefully. A small condition number is always desirable, but there can be designs with bigger condition numbers that are more desirable due to other characteristics. The interpretation of the condition number should be made in comparison to the other metrics provided. A big condition number in this case suggests that there is a direction in which the design can generate more thrust than some others. A very low condition number usually happens when the design approaches large omnidirectionality. Something like this can be favorable in terms of stability and generation of wrenches, but is losing in terms of energy efficiency.

3.2.4 Balanced distribution of thrust forces

The following metric, inspired by Jiang et al. (2011)[6], aims to achieve a balanced distribution of thrust forces, as this is paramount for ensuring stability, control, and efficiency. The metric employed to achieve this involves minimizing the sum of squared differences between the mean thrust force vector and the individual thrust force vectors. This metric will also be used directly as an objective function component.

Mean Thrust Force Vector

$$\tilde{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{F_i} \tag{30}$$

Objective Function term

$$J = \sum_{i=1}^{n} \| \tilde{\boldsymbol{\mu}} - \boldsymbol{F_i} \|^2 \tag{31}$$

Equal distribution of thrust is crucial for maintaining the multirotor's balance and stability during flight. An uneven thrust distribution can lead to imbalances, resulting in unstable flight dynamics. As a side effect, when the thrust is evenly distributed, it prevents any single rotor from being overworked while others are underutilized thus enhancing the overall performance and extending the lifespan of the rotors and motors.

3.2.5 Range and Endurance Calculations

These calculations are essential for evaluating the MRAV's operational efficiency by determining maximum flight duration and travel distance under optimal conditions, ensuring the MRAV meets endurance and range formal task requirements. Building on Bauersfeld and Scaramuzza (2022), but with changes made to account for tilted propeller MRAVs, the tool uses an algorithm to estimate range, endurance, and optimal flight speed for straight-line flight. The algorithm, based on momentum theory, calculates induced propeller velocity and mechanical hover power, then uses a hover-to-cruise power ratio and motor and battery models to determine maximum endurance and range. The key in this method is calculating accurately the induced velocity of rotors and power at hover. Unlike Bauersfeld and Scaramuzza (2022), which assumes symmetrical designs with no propeller tilt, this framework accounts for designs that vary from symmetric, with symmetric values for tilt angles, to purely asymmetric, with different tilt angles for each rotor, based on the task. Two methods were conceived for determining the thrust allocation among rotors and ultimately affect the output design as metrics of endurance and range: one assuming equal thrust distribution (hereby called "Method 1" and can be found in the Appendix B.2) and the other using the allocation matrix (hereby called "Method 2"). Method 2, which is more general, will be discussed here.

Unlike Method 1, Method 2 employs a more accurate approach if the allocation matrix is known and thus better suited for designs with varying tilt angles and rotor positions. More specifically, it solves for the input vector $\boldsymbol{u_h}$ and the desired wrench, in a state of counterbalancing only

[6] In his paper, he minimises the differences among angular velocities.

its weight (hover) $\mathbf{w}_h = [0\ 0\ mg\ 0\ 0\ 0]^T$ using the inverse, for FA systems with 6 inputs, (pseudo-inverse for more, or for quads reduce to 4x4 allocation and then inverse it) of matrix $\mathbf{F}$, which informs about the force and moment distribution of the system. The thrust for each propeller is then calculated based on this allocation and, similar to Method 1, the induced velocity and power consumption are calculated based on these thrust values:

$$\mathbf{u_h} = \mathbf{F}^{-1}\mathbf{w_h} \tag{32}$$

$$\mathbf{T_i} = c_f \mathbf{u_h} \tag{33}$$

When the tilt angles are zero or equal for all propellers, both methods yield the same results, as the uniform inputs lead to equal thrust distribution. However, when the tilt angles differ, Method 1 (equal thrust distribution) becomes inaccurate, as it does not account for variations in each propeller's input. In contrast, Method 2, which solves the system considering tilt angles and motor positions, provides a more precise thrust allocation, ensuring the system maintains the required force and moment balance.

All in all, both methods were tested and verified against the results of the method of the paper of Bauersfeld and Scaramuzza (2022). Method 1 is simpler and works well when tilt angles are equal or zero but may not be accurate for varying tilt angles. Method 2 provides a more precise thrust allocation making it suitable for scenarios with different tilt angles and more complex configurations. The rest of the quantities on Bauersfeld and Scaramuzza (2022) were unchanged, except for the fact that, now, one needs to consider the mean value of the induced velocities at hover to calculate the endurance and range. It is imperative to note that the numerical values of the fitted coefficients of some of the models used to calculate the endurance and range were kept the same as there was no time for further simulations and experiments and also it was out of the scope of this project. We expect to not have large deviations for relatively small tilt angles.

3.2.6 Minimum Guaranteed Control Force and Torque with unidirectional propellers

Evaluating the minimum guaranteed control force and torque ensures that the multirotor maintains control authority across all orientations, which is vital for the system's robustness in hazardous environments. Although in the AUTOASSESS case study the omnidirectionality of the robot is not a primary concern, generating wrench in all directions remains a valuable metric for evaluating the design and it can be used in the framework (there is a choice to select optimising for omnidirectionality). The used metric is inspired by the paper of Bosscher et al. (2006), which evaluates the distance to the closest hyperplane of the hull of the wrench space, in each iteration of the optimization process. That distance is the maximum force (force space) and torque (torque space) the robot can exert in all directions with unidirectional thrusters in our occasion and we write:

$$\mathbf{w}_{min}{}^{7} = \begin{bmatrix} \mathbf{F}_{min} \\ \mathbf{T}_{min} \end{bmatrix} \tag{34}$$

The derivation of these can be found in Appendix B.3.

3.2.7 Volume of the Robot

The metric for the volume helps in maintaining a compact design, while ensuring that the arm lengths and rotor positions do not lead to physical interference. This metric is essential for reducing the risk of collisions when passing through enclosed spaces and minimizing the weight of the design.

When one thinks of a fixed-tilted MRAV, full actuation is expected, meant for increased maneuverability and interaction tasks, but also the presence of internal forces and thus energy loss, thus metrics for both are needed. Moreover wrench generation capabilities are of the utmost importance (in all omnidirectional or preferred directions). Stability and balancing of thrusts

[7] This metric is implemented in the software, using the mathematical formulation of Bosscher et al. (2006).

are essential to any flying robotic system with multiple actuator and of course the volume and endurance are crucial task requirements that need to be addressed as metrics of a good design. Thus opting for all the aforementioned metrics, the framework provides a full-proof evaluation "quiver" for each optimal design output and a basis of comparisons between designs. These choices collectively contribute to the development of a robust, efficient, and reliable MRAV, capable of meeting diverse operational requirements.

3.3 Transcription of the Optimization Problem

In this section we focus on solving the aforementioned "Design Problem". The problem was decided to be approached and formulated as a constrained multivariable nonlinear optimization problem (NLP), where the goal is to find the optimal set of design variables, that minimize an objective function comprised of a weighted sum of multiple components. Each component represents a different performance aspect of an MRAV system and the problem is bound to several constraints, that address both the mathematical formulation of the task and the feasibility of the design. The IPOPT solver (see 2.2) was chosen to approach this problem, given its effectiveness in handling large-scale nonlinear programming problems with constraints. The general form of the function to be minimized by the solver is :

$$\min_{vars} \{w_1 \mathbb{U} + w_2 \mathbb{T}_h + w_3 \mathbb{P} + w_4 \mathbb{J} + w_5 \mathbb{Z}\}$$

$$\text{subject to:} \quad \begin{cases} 1)\ Constraint\#1 \\ 2)\ Constraint\#2 \\ n)\ Constraint\#n \end{cases}$$

where $\mathbb{U}$ is a term that represents the control effort over a maneuver or trajectory, the $\mathbb{T}_h$ term represents the thrust required to hover, the $\mathbb{P}$ term represents the volume of the robot through its perimeter, the$\mathbb{J}$ term represents the balanced distribution of thrusts, while the $\mathbb{Z}$ term represents the vertical thrust produced by the propellers. Moreover, the w_1 to w_5 are the weights for every term of the objective function. For their calculation, the knowledge of the bound values (or an approximation of them) is required.

As mentioned before, the nature of our problem is non-linear and thus the solution, is malleable to initial conditions. Given that IPOPT, and similar solvers, do not guarantee finding the global minimum, an iterative multi-start approach can be employed to enhance the chances of discovering the optimal solution. By executing the optimization process multiple times, each with a distinct set of initial values, one can aid in exploring different regions of the solution space and thereby increasing the likelihood of finding a better solution. The different converged solutions of these multiple executions are stored and compared based not only on the values of the objective function at the converged values of the optimization parameters, but also on the individual values of the components of the objective function and the aforementioned metrics (see 3.2). It is important to mention that, although in IPOPT, objective and constraints (but not variables) are scaled automatically, using first-order sensitivities at the initial guess, it is always a good idea to scale the NLP, in order to avoid a lousy convergence.

3.3.1 Objective function terms and constraints

The objective function's terms are a combination of modified cost functions from the literature and other components that were derived in order to produce a unified objective function that address simultaneously multiple performance aspects of an MRAV, with the aim of providing an optimal trade-off between these and the task requirements.

First, the components $\mathbb{U}$ and $\mathbb{Z}$ as formulated in 3.1.1, aim to address the requirement of endurance by minimizing the loss of energy within the system and the control effort, over a maneuver (or trajectory) of the task, ultimately reducing power consumption in flight. At the same time, the term $\mathbb{T}_h$ indirectly positively affects the flight endurance of the MRAV, as stated in 3.2.5, by

minimizing the thrust required for hovering. The term for Perimeter $\mathbb{P}$ is self-explanatory and was derived with the aim to achieve the smaller possible design that is optimal for the set of tasks. Bigger designs usually mean larger mass and inertiae and larger control effort to maneuver. Last but not least, the term $\mathbb{J}$ as presented in 3.2.4 accounts for the balanced distribution of thrusts, essential to any MRAV design (see more in 3.2.4).

Deriving from the aforementioned, the objective function to be minimized is the following :

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\lambda},\boldsymbol{l_x}} \left\{ w_1 \int \|\mathbf{u}(t)\|_2 \, dt + w_2 \mathbb{T}_h + w_3 \sum_{\substack{i,j=1 \\ i\neq j}}^{n} \left\| {}^B\mathbf{p}_i - {}^B\mathbf{p}_j \right\|_2 + w_4 \sum_{i=1}^{n} \left\| {}^B\tilde{\mu} - {}^B\mathbf{T_i} \right\|_2^2 - w_5 \left\| \sum_{i=1}^{n} {}^B\mathbf{T}_i \cdot \hat{z} \right\|_2 \right\}$$

$$\text{subject to:} \quad \begin{cases} 1) & -\frac{\pi}{2} \le a_i \le \frac{\pi}{2}, \\ 2) & -\frac{\pi}{2} \le b_i \le \frac{\pi}{2}, \\ 3) & \text{rank}(\mathbf{F}) = 6, \\ 4) & \omega_{\min}^2 \le \omega_i^2(t) \le \omega_{\max}^2, \\ 5) & \omega_{\min}^2 \preceq \mathbf{F}^{-1}\mathbf{w}_{\text{desired}} \preceq \omega_{\max}^2, \\ 6) & d_{\min_k} \le \| {}^B\mathbf{p}_i \cdot \hat{k} \| + r_{\text{prop}} \le d_{\max_k}, \quad \hat{k} \in \{\hat{x}, \hat{y}, \hat{z}\}, \\ 7) & \| {}^B\mathbf{p}_j - {}^B\mathbf{p}_i \|_2 \ge 2r_{\text{prop}} + \text{threshold}, \quad i \in \{1, \dots, n-1\}, \quad j \in \{i+1, \dots, n\}, \\ 8) & r_{\text{prop}} + \text{threshold} \preceq \mathbf{l_x}. \end{cases} \tag{35}$$

,with $\mathbb{T}_h$ from whichever method chosen from Section 3.2.5 and μ the mean thrust vector as in Section 3.2.4. One can easily observe that the optimization problem is highly nonlinear and non-convex, particularly in the objective function. The non-linearity arises from terms such as the control effort, thrust distribution, and vertical thrust components, each of which depends on nonlinear relationships between the design variables. The constraints themselves also contribute to the non-linearity, particularly the rank condition of the allocation matrix, ensuring full actuation. These interactions result in a complex optimization landscape where the feasible region is irregular, and the solver must navigate through sharp changes in the solution space. Thus, the use of an iterative approach, as mentioned before, with IPOPT is essential for exploring various parts of the solution space and increasing the likelihood of finding an optimal or near-optimal solution.

As far as the constraints are concerned (see eq. 35), they were carefully chosen not only to reflect the task's requirements but also to ensure they accurately capture the physical and operational limitations of the system. The tilt angle constraints (1-2) ensure that the propellers operate within the upper and lower bounds stated. The full actuation constraint (3) guarantees the MRAV's ability to control its position and orientation independently, providing the necessary robustness for performing precise maneuvers and force generation tasks. Additionally, input feasibility constraints (4-5) on the propeller speeds ensure that the drone operates within the specified motor limits and that the task's desired wrenches are feasible by the motors, following its mathematical formulation. The physical volume and clearance constraints (6-8), further ensure that the MRAV design follows the task's volume constraints while also being practical and safe for real-world applications. These constraints govern the arm lengths and propeller clearances, ensuring there is no physical interference between components during operation. This meticulous selection of objective function components and constraints helps in achieving a realistic and practical solution that meets all necessary safety and performance criteria for the drone's operation.

4 Software Implementation

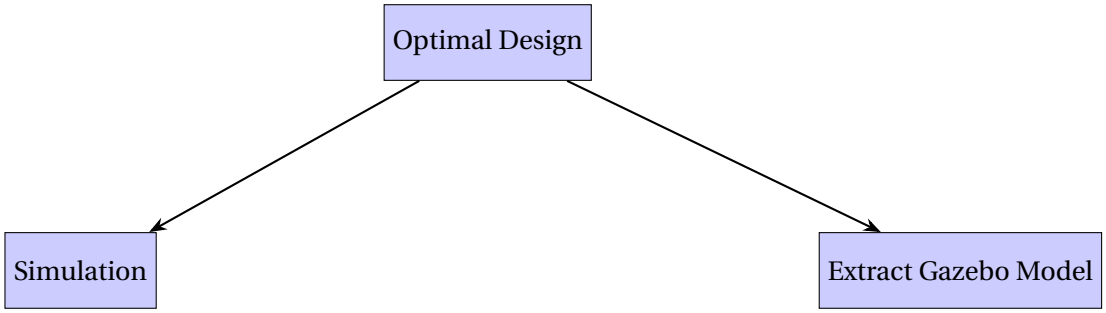
Building upon the theoretical framework established in Chapter 3, the development of the software framework aims to solve the "Design Problem (3.1.3)" formulated as a Nonlinear Programming (NLP) optimization task in 3.3. As mentioned in the previous section, to solve the problem, we will utilize the Interior Point OPTimizer (IPOPT) 2.2, a standard solver designed for large-scale non-linear optimization. The interface with Ipopt and it's integration with the "Design Problem" will be facilitated by CasADi, a symbolic framework that excels in automatic differentiation and numerical optimization[1]. CasADi's time efficiency, fast automatic differentiation, and libraries (like the "Opti") for flexible definition of design variables, constraints, and objective functions make it highly modular and simplify the handling of large systems. Moreover, its symbolic framework, which reduces the complexity involved in defining the dynamics and constraints of the system, streamlined the optimization process.

4.1 Developed Tool

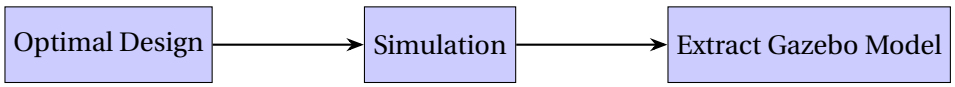
The MATLAB software consists of three main scripts, that comprise the "Developed Tool". The optimization script, a simulation script, and one that extracts the design to a gazebo model. Namely the scripts are the following:

- Optimal Design
- Simulation
- Extract Gazebo Model

The fundamental link between the three scripts, is that the output of "Optimal Design" (let us call it the "parent" script) is given automatically as "input" to both the "Simulation" and the "Extract Gazebo Model" (let us call them "child" scripts), as seen in the nodes directly under.



Though, all scripts can be used separately, the recommended sequence is :



In the following subsection, the structure and design choices of each script will be analyzed.

4.1.1 Optimal Design Script

The Optimal Design script leverages the numerical optimization capabilities of 'CasADi' to formulate and solve the "Design Problem" as a constrained multivariable nonlinear program, formulated in the way that is presented in Section3.3. The software framework's generality is rooted in its flexibility with respect to input and design choices, offering a wide range of design parameters that can be optimized. This flexibility enables users to define the level of constraint in the design problem, thereby shaping the final output. The framework outlines a generalized, systematic design procedure that guides the process from initial task requirements to the final

[1] More on CasADi can be found on the Appendix C.1

MRAV configuration. This approach ensures that the design is tailored to specific tasks while maintaining the adaptability needed to address a variety of MRAV configurations.

Preamble of the script and design choices
The script is to setup the optimization problem for an MRAV drone, including a GUI for configuration, and then to use this configuration to define input parameters and variables to be optimized.

Firstly, we import 'CasADi' and we initiate the 'Opti-Stack' class by ($'opti = casadi.Opti()'$). This initialization sets up a flexible and user-friendly framework for defining nonlinear programming problems. The 'Opti' object acts as a wrapper around CasADi's core functionalities, allowing for an accessible and efficient formulation of optimization models. This approach minimizes the complexity typically involved in setting up and solving nonlinear programs, streamlining the process for users at all levels of expertise.

User configuration is facilitated via the custom GUI, $'createConfigurationGUI()'$. The configuration GUI window can be seen in Appendix C.1. This approach allows users to input or adjust design parameters interactively, providing maximum output flexibility and generalization of the tool for diverse tasks. The script uses the $'waitfor())'$ command to pause execution until the GUI window is closed, ensuring that the user has finished configuring the parameters before the optimization proceeds.

Following the closure of the GUI, various parameters are extracted from the 'params' structure, encompassing physical dimensions, mass properties, thrust and drag coefficients, and trajectory endpoints. Parameters such as $'mass'_{body}$, $'r'_{body}$, $'h'_{body}$, $'mass'_{rotor}$, $'r'_{rotor}$, $'h'_{rotor}$, and the total mass ('mass') are defined to accurately model the MRAV's components. Trajectory parameters, including initial and final orientations ($'\phi','\theta','\psi'$) and positions ('x', 'y', 'z'), are specified to reflect a desired maneuver from point A to point B (in the case of optimization to a trajectory in Section 3.1.1).

To ensure that the optimization considers the task's desired wrenches, the script initializes and populates a matrix $'wdes\_set'$ with desired wrench values (forces and torques) for each task, thus defining the set of desired wrenches of a task. The user needs to know prior to the desired wrenches his task requires and input them in the corresponding GUI. After that the optimiser will check the satisfaction of each one of them, using the method described in subsection 3.1.1, in order to produce the final design. For the desired volume constraints, the Volume constraints are also defined, as maximum allowable dimensions ($'diam\_x','diam\_y'$) and motor placement constraints ($'motors\_on\_x\_axis'$) to be satisfied. Corresponding flags ($'flag\_diam\_x','flag\_diam\_y','flag\_motors\_on\_x\_axis'$) are set to indicate the presence of these constraints, ensuring their consideration during optimization (see Appendix C.1).

As seen in Section 3, the decision variables for optimization, includes the tilt angles ($\boldsymbol{\alpha}$, $\boldsymbol{\beta}$) and the position of each rotor, defined by the arm length $\boldsymbol{l_x}$ and it's angle ($\boldsymbol{\lambda}$). The script's design prioritizes flexibility and user interaction, allowing various parameters to be adjusted via the GUI. This modular and extensible design approach, with structured parameter handling within the 'params' structure, maintains clarity and ease of modification.

In summary, the design choices made here were done based on the theoretical framework derived for a generalised design procedure from task requirements to output design, as well as to emphasize flexibility, user interaction, and precise modeling of the MRAV system. By integrating CasADi for optimization with a MATLAB GUI for user input, the script provides an effective and user-friendly method to address the complex design problem.

Configuration Options

Accounting for generality and aiming for applicability in diverse applications, this optimization framework provides the user with the choice to include multiple configuration options of its optimization variables ($\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}$). This is a well-considered decision that strikes a balance between reducing optimization complexity and providing full flexibility, catering to a wide range of user needs and scenarios. This approach enhances the usability, scalability, and practical relevance of the optimization framework, making it a robust tool for diverse applications. The configuration options are divided into two distinct categories:

"Configuration Options for Tilt Angles"
&
"Configuration Options for Rotor Positions"

These options are selected independently; however, they are always used in conjunction to provide appropriate input for the subsequent optimization problems as seen in figure 4.1. The provided design choices for configuring tilt angles and rotor positions are aimed at optimizing the performance of a system while managing the complexity of the optimization problem. Each option is designed to strike a balance between simplicity and effectiveness, inspired by approaches from the aforementioned literature.

Figure 4.1: Configuration Options for Tilt Angles (up) and for Rotor Positions (down), from the GUI.

Configuration Options for Tilt Angles (COTA) In "Configuration Options for Tilt Angles", one chooses to optimize the tilt angles ($\boldsymbol{\alpha}$, $\boldsymbol{\beta}$), with a certain symmetry or not. Symmetrical configurations (of each rotor frame with respect to the body frame) is typically found in the literature (i.e. standard hexarotors). In this case, the rotors are distributed equiangular in the unit circle, with $\lambda_i = (i-1) \cdot \frac{2 \cdot \pi}{n}$ with i=1,$\cdots$,n while also the arm lengths $\boldsymbol{l_x} = l$ are fixed[2]. The design choices for configuring $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are influenced by the need to reduce optimization complexity, as proposed in Rajappa et al. (2015). Analytically, the 'Configuration options for tilt angles" are:

Option A:

- Design Choice: Assigns common values for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ with alternating signs. More specifically, the tilt angles configuration is: i=1,3,5 $+\alpha/+\beta$, i=2,4,6 $-\alpha/-\beta$.
- Purpose: Inspired by Rajappa et al. (2015), the configurations from A through C, balances simplicity and effectiveness by reducing optimization complexity, through limiting the number of design variables (instead of 2*n[3], only 2 with different signs). They are used for symmetrical designs in conjunction with the "Configuration option for rotor position A" (see 4.2).

Option B:

- Design Choice: Assigns positive α for odd propeller indices and negative α for even indices, with all β values being positive. More specifically, the tilt angles configuration is i=1,3,5 $+\alpha/+\beta$, i=2,4,6 $-\alpha/+\beta$.

Option C:

- Design Choice: Assigns positive α for odd indices and negative α for even, with β following an alternating sign pattern. More specifically, the tilt angles configuration is i=1,3,5 $\alpha/-\beta$, i=2,4,6 $-\alpha/+\beta$.

[2]This is done with the Configuration Options for Rotor Position "A".
[3]n being the number of AUs.

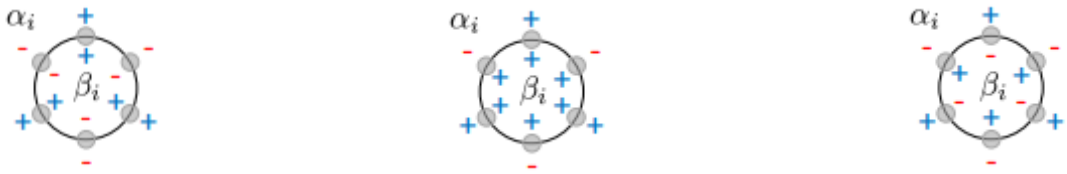


Figure 4.2: Available symmetric Configuration Options for Tilt Angles from the software: A (left), B (middle) and C (right) (from Rajappa et al. (2015))

Option D:

- Description: Treats $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as variables to be optimized without constraints. More specifically, the tilt angles do not follow a specific configuration but they are variable.
- Design Choice: Allows $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ to be fully optimized by the solver.
- Purpose: This provides maximum flexibility, resulting in a more complex 2*n-variable optimization problem[4]. It's suitable for scenarios where designs with no specific symmetry are required [5] and computational resources allow for a more thorough exploration of the solution space.

Configuration Options for Rotor Position (CORP) The design choices for optimizing arm length and arm angle are structured to offer varying degrees of flexibility and computational complexity. Each option provides specific advantages tailored to different optimization needs. Analytically, the 'Configuration options for rotor positions' are:

Option A:

- Description: No optimization of the arm length ($\boldsymbol{l_x}$) or angle $\boldsymbol{\lambda}$.
- Design Choice: This option is designed for simplicity and in reducing the need for complex calculation (standard option when optimizing only for tilt angles).
- Purpose: By using fixed values for both ($\boldsymbol{l_x}$) and angle λ, it ensures a symmetric equiangular configuration. This is beneficial in scenarios where computational resources are limited or where a standard configuration (symmetrical) is required. Moreover, this choice introduces no asymmetries in the MRAV's body and none or a minimum change of the CoM. An example can be seen in Fig. 4.3.

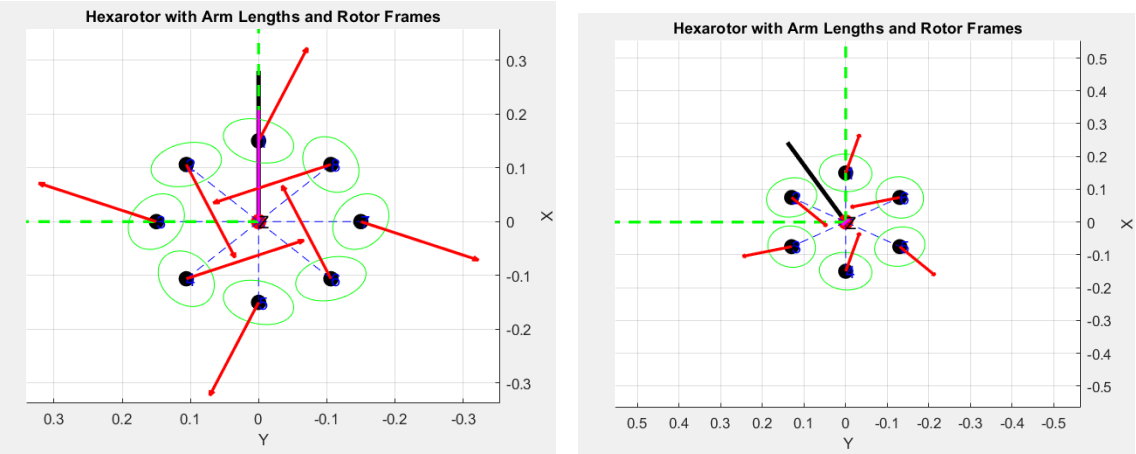


Figure 4.3: Example output of the "Optimal Design Script" for "Configuration Option for Rotor Position (CORP) A", of a fixed-tilted angle octarotor (left) and fixed-tilted angle hexarotor (right). The "Configuration Option for Tilt Angle" is A or (COTA-CORP):=(A-A). Axes are in meters.

Option B:

[4] Simplest case of fully actuated design being the hexarotor with n=6.
[5] Usually used in conjunction with "configuration options for rotor position (CORP) B and C".

- Description: Optimize only arm length (l_x) in equiangular configuration.
- Design Choice: This approach introduces flexibility in arm lengths while maintaining a regular angular distribution (λ).
- Purpose: This option allows for a different configuration that can adapt to specific performance criteria such as adhere to volume constraints, without compromising others[6]. The fixed angular pattern adds a level of complexity to the optimization problem, while providing "task-tailored" advantages over the default configuration. This balance between flexibility and complexity makes it suitable for applications needing customized performance without excessive computational demands. An example can be seen in Fig. 4.4.

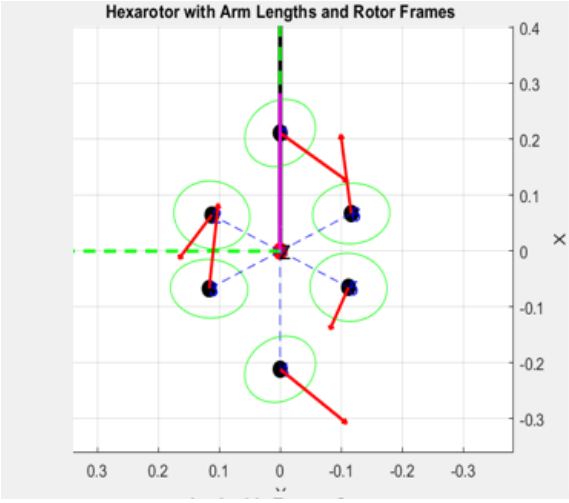


Figure 4.4: Example: "Configuration Option for rotor position" B and "Configuration Option for tilt angles" D, (D-B), of a fixed-tilted angle hexarotor with volume constraint in the y-axis of 0.36m (diameter). Notice how the positions of the rotors, although their angles are 60 degrees apart, their lengths are different. Axes are in meters.

Option C:

- Description: Optimize both arm lengths and arm angles
- Design Choice: This option offers the highest degree of flexibility by allowing both λ and arm length (l_x) to be optimization variables.
- Purpose: This enables the system to find the optimal configuration which is tailored precisely to the specific needs and constraints of the task. Such an approach can lead to optimal performance in terms of wrench generation, volume, and energy efficiency. However, the increased flexibility comes with greater computational complexity and practical implementation of the design. This makes it ideal for high-precision applications where the benefits of optimal configuration justify the additional computational resources. An example can be seen in Fig. 4.5

The design choices for both tilt configuration options and rotor positions provide a spectrum of solutions from "simple" and stable to highly optimized and complex. Options A in both cases prioritize simplicity and stability, making them ideal for initial setups or resource-constrained environments. Options B offer a middle ground by allowing some level of optimization while maintaining a simple structural pattern. Options C and D (for tilt configuration) provide maximum flexibility and optimization potential, suitable for high-precision and performance-critical applications. These choices enable users to select the most appropriate strategy based on their

[6]i.e. navigating narrow corridors with larger designs, without making the whole drone smaller, with smaller propellers, that compromise wrench generation capabilities

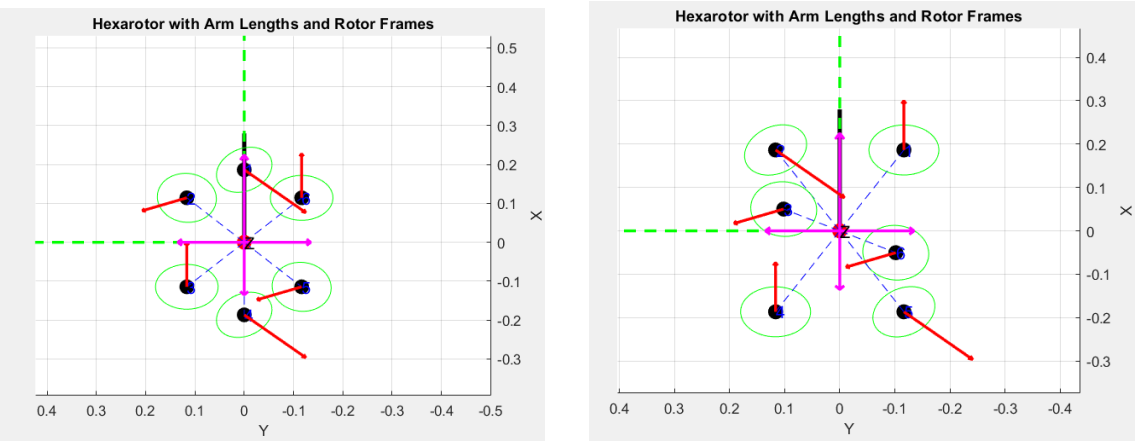


Figure 4.5: Example: Configuration Option for rotor position C, of a tilted angle hexarotor with volume constraint in the y axis of 0.36m (diameter) and 0.5m on the x axis. Both cases are identical in everything except that the left case there is the option of involving some symmetry in this configuration option, by constraining the motors 1 and 6 on the x axis (more balanced distribution of thrusts). Notice how the positions of the motors are different (the purple lines correspond to the desired wrenches, more on that in later chapters). Configuration Option for tilt angles is D. Axes are in meters.

specific requirements and constraints, balancing between ease of implementation and the need for a customized, optimized setup.

Inputs-Configuration Parameters

As stated in the subsection 4.1.1, the Optimal Design script, provides a custom GUI at launch, allowing the user to input the necessary fixed parameters for the design. Therefore, for a more optimal output design, or better stated, closer to an applicable design, a high-level conceptual design of the system is recommended. To be more specific, parameters such as mass, propeller characteristics, motor specifications etc can heavily influence the output design, in terms of flight time, wrench generation capabilities etc, thus the better the knowledge of these parameters are, the more accurate and optimal the final design would be.

It is of great importance to discuss the input parameters that the script requires and their effect on the output. For this, in the tables 4.1 and 4.2 one can be see of all the required parameters to be provided by the user at launch.

Table 4.1: Table of input parameters.

Actuator Parameters	Body Parameters	End Effector Parameters
Number of Actuators	Mass of the body (kg)	Mass of EE (kg)
Radius of Propellers (m)	Radius of the body (m)	Radius of EE
Mass of the actuator (kg)	Height of the body (m)	Height of EE (m)
Radius of the actuator (m)	Arm length (m)	Position of EE (wrt body) (m)
Height of the actuator (m)	Max Width (m)	Attitude of EE (wrt body) (rads)
Min rotational speed (Hz)	Max Length (m)	
Max rotational speed (Hz)		
Coefficient of thrust (N/Hz^2)		
Coefficient of drag (Nm/Hz^2)		

Starting from the actuator parameters, we consider that all the actuators are the same with the same limits and dimensions, as well as unidirectional thrust. More specifically, the script can theoretically accept any number of actuators. The recommended is above 5. The reason for that is that the tool is designed in a way to explore and find designs that are fully actuated.

Table 4.2: Table of input parameters.

Battery Parameters	Trajectory Parameters	Other
Mass of Battery (kg)	Initial Position (m)	Config. Option: tilt angles
Capacity of Battery (Ah)	Initial Attitude (degs)	Config. Option: rotor pos.
Number of cells in series	Final Position (m)	Desired wrench set (N and Nm)
Number of cells in Parallel	Final Attitude (degs)	
Voltage (V)		

Of course, this can change by disabling the full actuation constraint as we will discuss further. The coefficients of thrust and drag as well as the min/max rotational speed of the rotors, are expressed in units of N/Hz^2, Nm/Hz^2, and Hz accordingly. The decision of these values is of critical importance and their prior knowledge (or an approximation) is heavily recommended, due to their big influence in the wrench generation of the output design. The parameters of the actuators and that of their mass are also of great importance, as they can influence the stability, the flight time, and the wrench generation of the system. This is more evident in asymmetric designs and more specifically in the total inertia of the system.

The so-called "body parameters" are the ones corresponding to the frame of the MRAV. More specifically, the mass and dimensions of the body refer to the main part of the body, where the majority of the electronics are, along with the battery etc. The mass of the body expects the total mass of the main body with the electronics and the battery (but not the End Effector, which is a separate part). It is the parameter that affects most of the flight time of the drone. The radius and the height of the body are self-explanatory and an approximation is enough. The arm length parameter is only necessary when the user does not optimise for it and an example could be the typical hexarotor configuration with a chosen arm length.

The End Effector parameters provide the user with the choice of "mounting" a simple End Effector to the design, as an individual component, in order to provide a more realistic final design for potential contact tasks. More specifically, the mass, dimensions, position, and attitude with respect to the body frame, of the End Effector influence the total inertia of the robot, and is typically a good practice to include them in the modeling of the dynamics of the system.

The battery parameters let the user choose their own battery, in order to provide a more realistic approach in calculating the power consumption, of the final design. To be more specific, the user is called upon to choose the characteristics of mass, capacity, cells in series and in parallel, and the voltage of the battery, to be used as inputs in the power consumption section of the "Optimal Design" script. The mass and capacity of the battery (and also the other elements) play a crucial role in the flight time of the final design.

As mentioned in earlier sections, the user can choose to optimize for a certain trajectory or maneuver. So far, this approach is only applicable from a waypoint A to waypoint B. The benefit of this can be found, as an example, in the optimization of the tilt angles for a fully actuated maneuver, while minimizing the control input. Along with this, there is the option to apply an external wrench, to the target to be optimized system, for a fixed duration of time, during this maneuver, by the user.

As far as configuration options are concerned, the expected values are A through D for the tilt-angle configurations and A through C for the configuration of the position of the actuators. Their use has already been discussed, and will be further elaborated in the following sections. Last but not least, the user is asked to input the desired wrenches the final design would want to be able to generate. The user can add how many he deems necessary. A typical input of a

desired wrench is of the type:

$$\mathbf{w_{des}} = [w_{des_x}(N)\ w_{des_y}(N)\ w_{des_z} w_{des_{roll}}(Nm)\ w_{des_{pitch}}(Nm)\ w_{des_{yaw}}(Nm)]^T$$

, where the desired wrench on the vertical axis is a dimensionless number, corresponding to a multiplier of the total weight of the robot, for example 2.5 (generate force of 2.5 times the weight of the robot).

As will be discussed in the next sections, all the parameters contribute together to provide a "spherical", unified, and well-defined input to the Optimal design script. This input forms the foundation of the systematic and generalized procedure, which begins with a thorough understanding of the task and a high-level conceptual design. The more accurate the aforementioned parameters, the more realistic and balanced the optimal output design will be.

Assumptions about the robot geometry
In order to calculate the inertia of each design, the framework considers the modules from which the robots are constructed as solid cylinders. Due to the fact that the framework works closely with the Gazebo simulator, in order to validate the designs, the assumption of solid cylinders is made for the modules of actuators, body, and end effector (EE). Through this assumption, the framework achieves a balance between simplicity and realism. This design choice facilitates both the construction of the model in Gazebo, using the "Extract Gazebo Model" script, and the ease of calculating the robot's total inertia, reducing the computational load in each iteration of the optimizer. This aligns with the input parameters "radius" and "height" for each module.

Another geometrical assumption that the framework makes is that the CoM of the propellers coincides with that of the motors. This means that the framework considers that the thrust vector produced by the actuators starts from the end point of each arm and does not take into account the height of the motor. This design choice was made based on the fact that the arm length is much larger than the motor's height, and thus the additional dynamics there would be negligible.

In the framework's geometrical assumptions, a key consideration is the volume of the Multirotor. Specifically, the framework assumes that the height of the MRAV will not be the primary focus of the optimization process. Instead, emphasis is given to optimizing the perimeter of the MRAV, on the plane, where the rotors will lie. This means that the user must account for the height of the system independently, ensuring the height constraint of the task. This can be done by specifying the approximate "height of the body", through the GUI console during the launch of the "Optimal Design Script".

This design choice was made mainly due to the fact that, by placing the rotors in the same plane, the size and weight distribution of the MRAV are more directly influenced by its perimeter than its height (height of base relatively small to length and width of the system) and thus optimizing the perimeter allows for better management of these factors, contributing to a more robust and balanced design. Having said that, proper consideration of the height is necessary to ensure an optimal volume, proper consideration of the task's volume constraint and weight distribution.

Perimeter Calculation of the MRAV in each iteration
The provided script is designed to calculate the perimeter of a system with multiple arms, taking into consideration both the arm lengths and the contributions from the propellers located at each arm's endpoint. The script employs a methodical approach to ensure the accurate measurement of the robot's perimeter. More specifically, it iterates over each arm and systematically computes the distance between the endpoints of consecutive arms, ensuring that all segments are included in the perimeter calculation. The circular indexing mechanism ($j = mod(i, np) + 1$) is a design choice that ensures that after processing the last arm, the calculation wraps back to the first arm, "closing the loop" and forming the system's perimeter.

The approach includes a simple calculation of the Euclidean distance between the endpoints of each pair of consecutive arms using the 'norm' function and to account for the additional components at the endpoints of the arms, it further adds the contribution of the propellers to the perimeter. This simple but elegant approach accounts for low computational loads in each iteration of the optimization process.

Optimizing for control effort through the norm of control output
An important feature of the framework, inspired by the paper of Rajappa et al. (2015), is the addition of a simulator script, that tracks a trajectory, the aim of which is not only the minimization of the control effort of the MRAV (over the trajectory), but also the verification that the output design can achieve 6-DoF trajectory tracking. As discussed in their paper, it is of great importance to test the ability of the final design to reorient while hovering and respond to external force/torque disturbances (which is a feature of a fully actuated platform). Given different $\boldsymbol{\alpha}$ and$\boldsymbol{\beta}$ tilt angles, not all trajectories may be feasible due to potential negative control outputs. The added feature consists of a trajectory generator and a control loop, which performs feedback linearization and uses a PID controller.

- Trajectory Generation and Smoothing

The script begins by generating smooth trajectories for the orientation angles ϕ, θ, ψ and positions (x, y, z) using a smooth profile function. The smooth profile function generates a smooth trajectory profile for the specified initial and final states specified in the configuration sections, over a specified time period. It uses a quintic polynomial to ensure smooth transitions with continuous derivatives up to the second order. The function also computes the velocity and acceleration profiles and enforces constraints on these profiles to maintain realistic values. More specifically, the function calculates coefficients for a quintic polynomial to generate smooth transitions between the initial and final angles. The coefficients are calculated based on boundary conditions that enforce the trajectory to start and end with zero velocity and acceleration, for smooth maneuvers.

From the position and orientation trajectories and leveraging the differentiation capabilities of the "gradient" function, the desired angular and linear velocities are derived. The downside to this approach is that numerical differentiation can be sensitive to noise and may introduce errors, particularly if the time steps are not fine enough and can potentially affect the precision of the control inputs.

- Control Loop

The main control loop iterates over time steps to compute control inputs and update the state of the MRAV. This structure allows for updating of control inputs and states, making the system responsive to environmental changes and trajectory adjustments. The update frequency can be adjusted and must be considered when working with fast dynamics. The inclusion of a constant external disturbance wrench accounts for scenarios involving contact.

Positional errors are calculated and integrated over time to account for cumulative deviations. This helps to eliminate steady-state errors, improving long-term tracking accuracy. Although, integrating errors can lead to windup issues if not properly managed, especially during sudden changes or prolonged deviations, this computationally cheap approach aids the optimizer.

The script make use of feedback linearization and a PID controller as the control law. More specifically, it computes the desired orientation matrix and angular velocities to control the MRAV's attitude. Using rotation matrices ensures accurate and continuous representation of 3D rotations. This well-known control law combining proportional, integral, and derivative terms provides robust responses to position and velocity errors as well as helps eliminate steady-state errors. A drawback of this method is the hard-coded control gains that might require tuning for

different MRAV configurations or operating conditions. Adaptive gain tuning could enhance the system's performance across varied scenarios but come at a cost of computational load.

- Forward Dynamics and State Integration

The state of the MRAV is updated using forward dynamics and numerical integration via the Euler method. Forward dynamics offer a realistic model of the MRAV's motion, accounting for forces and torques. Euler integration is simple and computationally efficient, making it suitable for our optimization application. However, it can introduce numerical errors, especially with larger time steps. Implementing higher-order integration methods, such as Runge-Kutta, could improve accuracy.

Visualization of the Force and Torque space

The script computes the available force space of the robot when applying zero torques and the available torque space when counteracting only its weight. The method used to calculate and graphically represent the attainable force and torque space, leverages the convexity of the wrench space given a convex input set. By representing the wrench set as a zonotope constructed by the columns of **F**, as generator vectors, we can very quickly construct its convex hull directly from **F**. In order to do that we need to prove that given a convex input set $\mathbb{A}$ and a linear **F**, for a generic Fully Actuated MRAV, with unidirectional thrusters, its wrench space is convex. The proof can be found in the appendix A.

Having established the convexity and inspired by the concepts introduced by Bosscher et al. (2006) for generating the wrench-feasible workspace (WFW) for cable-driven robots, we can graphically visualize the force and torque space. To be more specific, the script takes as input parameters maximum thrust of each actuator (which is assumed to be the same for all actuators), the weight of the MRAV, the allocation matrix **F**, the number of actuators and some other values for the addition of the visualization of the minimum guaranteed control force and torque in all directions of the design, as well as the desired set of forces and torques for comparison. An example can be seen in Fig. 4.6.

Initially, the function generates all possible combinations of on or off states of the actuators. This is accomplished using binary representations, resulting in 2^{np} combinations, where n_p is the number of actuators[7]. For each combination, the resulting force vector is computed and stored (applying always the maximum thrust for each actuator in order to achieve the maximum thrust or torque in each computing direction). This represents the maximum forces or torques generated by the actuators in different configurations. Afterward, to visualise the boundary of the attainable force space (or torque space), the function employs the "convhull" function of matlab, which computes the convex hull of the generated force (or torque) vectors. The convex hull is the smallest convex set that contains all the points, representing the boundary of the force space (or torque space). Again we could not do this step if we could not prove that the wrench space is convex, as it implies that any linear combination of forces (or torques) within this set is also within the set, ensuring that intermediate forces (or torques) between two achievable forces (or torques) are also achievable. This property is essential for using this method of representation of the force and torque space that compared to other methods is extremely fast (order of ms).

Finally, the convex hull of the attainable force/torque space is then plotted using the trisurf function, which creates a triangulated surface plot of the convex hull.

For purposes of omnidirectionality, the script also visualizes a sphere centered at the hovering point of the robot, with a radius equal to the distance of the closest plane of the force/torque hull, representing the minimum guaranteed force/torque control. This sphere is plotted with transparency to provide a visual aid in understanding the force/torque capabilities of the MRAV. An example can be seen in Fig. 4.7. The script also visualizes the desired force and torque vectors

[7] A similar approach can be found in Xu and Saldaña (2023)

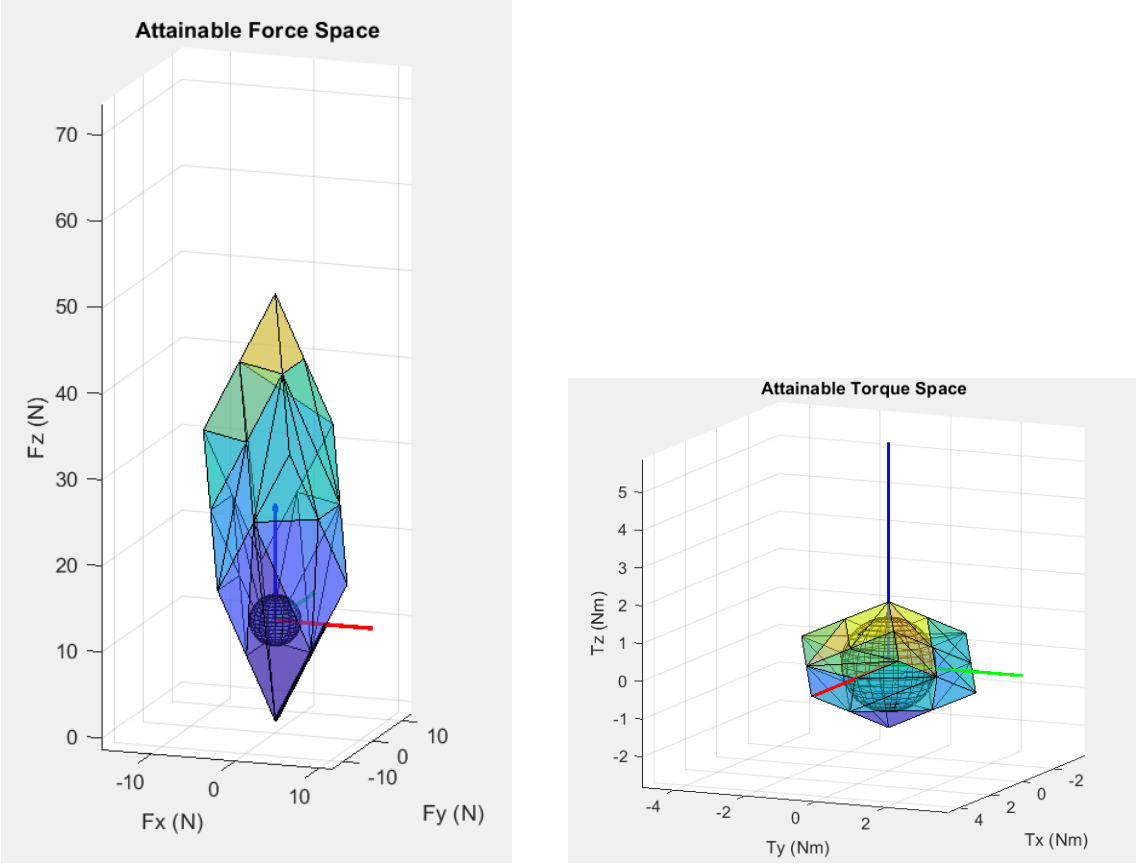


Figure 4.6: Example: Force and Torque space of a tilted hexarotor of Configuration Option for tilt angles C and Configuration Option for the position of motors A. The desired wrench is $\mathbf{w_{des}} = [6\ 0\ 1\ 0\ 0\ 0]^T$, black arrow in force space (left). Notice the sphere that represents the minimum guaranteed force/torque control.

from the desired wrench set that the user "asked for" during the configuration phase, using quiver3 to plot arrows originating from the origin. This helps in understanding the desired forces and torques in relation to the attainable force/torque space. Additionally, coordinate axes are plotted for better visualization, aiding in comprehending the spatial orientation of the force vectors. Finally, the function sets axis labels, limits, and title for the plot to clearly indicate the force/torque components and the title of the visualization. This structured approach ensures that the force space is accurately represented and easily interpretable.

Optimization part and results

As discussed in Section 3.3, the off-the-self solver used in this context is "IPOPT." Prior to running the software, the user has the ability to modify several option parameters of the solver to enhance the accuracy and convergence of the solution. Additionally, the user can visualize the convergence of the solution during the solver's iterations through a "callback" function. This feature aids in diagnosing convergence issues and fine-tuning the solver to the specific application. For instance, if the solver fails to converge, it is often indicative that the tolerances and constraint violation tolerances are too stringent and need to be relaxed. An example of Convergence of a solution, part of the output results of the software, is presented in Appendix C.2.

The general structure of the optimization problem to be solved for each configuration option remains consistent. It starts with defining an objective function to be minimized, which is uniform across all configuration options (see Section 3.3), but can be adjusted to fit the specifications of each application. This minimization process must comply with a set of constraints,

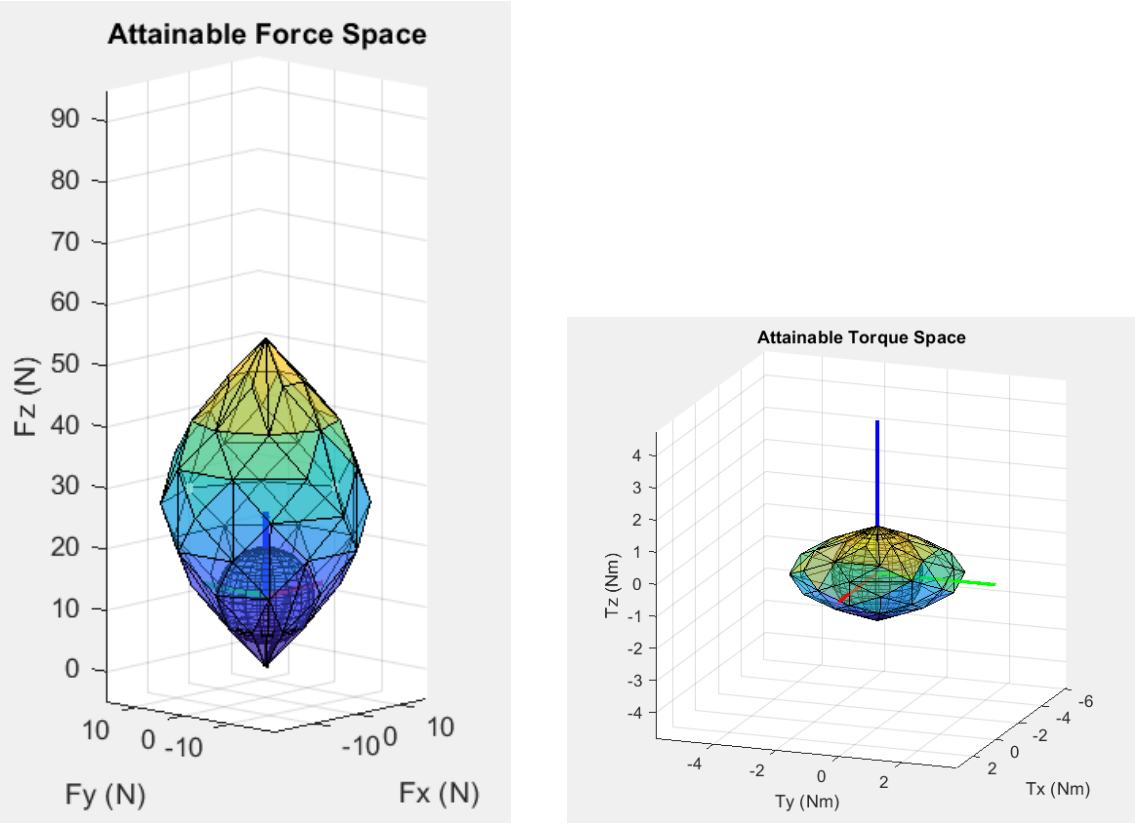


Figure 4.7: Example: Force and Torque space of the octo-rotor of Fig. 4.3

inspired from the mathematical formulation of the problem and of course use some initial values. Following the setting of solver tolerances (see Appendix C.1), as prior discussed, the solver addresses the problem and stores the optimized parameters. This structured approach ensures that the optimization process is both systematic and adaptable, allowing for precise adjustments to achieve optimal solutions tailored to specific requirements.

Upon the solver's convergence to the optimal solution, the next step involves visualizing the Multirotor Aerial Vehicle (MRAV). This visualization includes plotting lines extending from the center of mass (COM) to the base of each rotor, as well as arrows representing the tilt angles. The design parameters are then saved in a text file, which will be utilized by the simulation and the extraction model script for Gazebo.

In addition to visualization, the process involves plotting the force and torque space of the optimal design, along with the minimum guaranteed control force and torque. This provides a graphical representation of the performance capabilities of the MRAV. Finally, the endurance values of the optimal design and the other metrics are calculated and printed, offering a quantitative measure of the vehicle's operational efficiency. This comprehensive approach ensures that the design is thoroughly evaluated and documented, facilitating subsequent simulation and implementation stages. Example outputs of the Optimal Design script can be seen in Appendix C.3 and C.4 . Moreover, some example designs can be seen in the appendix C.5, C.6, C.8 and C.7, with their corresponding force and torque space. It can be clearly seen that one is able to redesign the force space to his liking by simply adding desired wrenches for the optimizer to follow. That way one is able to achieve the specified tasks and find optimized designs.

4.1.2 Simulation Script

The "Simulation" script is a simple but effective simulator aiming to test the optimal solution found in the "Optimal Design" script, in terms of energy efficiency, behaviour under the influ-

ence of external disturbances, stability and of course ability to track independently position and orientation trajectories. It uses a waypoint and trajectory generator with a convenient GUI to select every waypoint in 3D space, duration between each waypoint, orientation and position. Moreover, there is the ability to select only hovering and test (if the user desires) for external disturbance during hovering. A high-level block scheme can be found in Appendix F.1, but it is advisable to review it after reading through this section for better understanding.

Preamble of the script
The simulation script can be used separately, but the recommendation is to be used after running the "Optimal Design" script. For this purpose, this script initiates by reading the optimal design (the solution), saved in the "optimal-design.txt", by the "Optimal Design" script. More specifically it assigns the design values and the loads the current configuration of the position of the motors and orientation of tilt angles[8]. Then the script proceeds to calculate the allocation matrix that is going to be needed in the next steps.

Trajectory generator
As soon as the user runs the simulation script, a GUI pops up. This GUI was made with the goal of making the trajectory generation an easy task and more accessible to everyone. The function behind the GUI which is called, is "Interactive trajectory plot with GUI" and enables the user to intuitively add, edit, and visualize trajectory points, providing a user-friendly interface for trajectory generation and analysis.

GUI Description and Functionalities The primary purpose of the GUI is to facilitate the plotting of a trajectory in a 3D space. The GUI offers both interactive and manual methods for adding trajectory points, allowing users to either click directly within the plotting area or input coordinates through text fields. The following functionalities are provided:

- 3D Plot Setup

A 3D plotting environment is established using MATLAB's figure and axes functions. The plot is configured with a 3D view, grid, and axis labels for X, Y, and Z coordinates. Initial axis limits are set to [0 20 0 20 0 20]. It is presented in Appendix C.9.

- User Interaction

The GUI listens via "callbacks" for mouse clicks and key presses to manage user input. Users can add points to the trajectory by holding the 'Ctrl' key and clicking within the plot area. The current mouse position is continuously displayed in the plot title for reference.

- Manual Coordinate Input

Text fields and a button are provided for manual entry of X, Y, and Z coordinates. Upon clicking the 'Apply' button, the entered coordinates are validated and added to the plot if valid.

- Undo Functionality

Users can undo the last added point by pressing 'Ctrl+Z'. This removes the most recently added point from both the plot and the internal data structure.

- Data Storage

The coordinates and associated trajectory information are stored in an array trajectoryInfo. Each entry is collected through an input dialog when a point is added.

- Closing the GUI:

When the GUI is closed, the trajectoryInfo array is returned, containing all the trajectory points and their respective information.

After closing the GUI, the "trajectoryInfo" array returned by "interactive_trajectory_plot_with_gui", is processed to generate a trajectory using Matlab's "waypointTrajectory" object. The object

[8]For the case that the user wants to use it separately to test a design, they need to hardcode the values of each parameter of the design.

uses the waypoint positions (specified in the data structure created by the GUI), time of arrival, orientation, and sample rate. The orientations are converted to quaternions from Euler angles, ensuring correct representation of rotational data. The sample rate is set to 10 Hz, and the initialized zero velocities are used in order for the MRAV to stop in each waypoint. After that, the initial positions, orientations, velocities, accelerations, and angular velocities are obtained from traj(). The simulation then proceeds in a while loop, iterating through each sample frame, until the trajectory is complete. During each iteration, the trajectory information is updated and stored, in order to be used as desired trajectory inputs in the control part of the script later on. More detailed information about it can be found in Mathworks (2024). The script proceeds to visualise the 3D trajectory with the waypoints and the desired orientation trajectory in separate plots.

Control Loop
The control loop of the simulator is based on the methodology presented in "Optimal Design". This design enables users to tune the parameters of the PID controller utilized in the simulation. Additionally, it offers the capability to introduce external disturbances for specified duration, allowing for a comprehensive analysis of the system's response under varying conditions.

A significant feature of the control loop is the computation of the total drag torque experienced by the design throughout the trajectory. This metric is especially important for stability evaluation, particularly when testing asymmetric designs and subjecting them to external disturbances. Drag torque can significantly influence the stability of a system, and its evaluation provides insights into the design's performance and potential areas for improvement.

To facilitate a thorough analysis, the simulator plots several key aspects of the trajectory in separate graphs. The real position and orientation trajectories are plotted over time, offering a clear visualization of the MRAV's maneuvers throughout the simulation. Additionally, the control signal is plotted over time, with dotted lines representing motor limits. This feature allows users to quickly assess the feasibility of the trajectory and determine whether the system can handle external disturbances of specified magnitudes. Furthermore, the real angular velocity is plotted over time, enabling a comprehensive evaluation of the system's rotational dynamics and stability in 6D.

These visualizations are crucial for a detailed 6D tracking evaluation. By providing clear and detailed insights into the system's behavior, users can identify potential issues and make informed decisions regarding the design and control strategy. The combination of these features makes the simulator a powerful tool to aid in the optimal design process and evaluation.

Metrics calculation and Animation
The calculation of some of the optimal design metrics, specifically the "condition number" and the "force efficiency index," is performed using the simulation script. This approach is adopted due to the practical advantages of numerical calculations over symbolic computations in CasADi. Examples of these are that the "svd" function of matlab does not account for the variable types of CasADi and that numerical methods are often more straightforward and computationally efficient for evaluating these metrics. The design choice to include these calculations here, not only simplifies the computation process but also enhances the robustness and reliability of the results.

An essential feature of the simulation is the real-time animation of the Multi-Rotor Aerial Vehicle (MRAV) as it follows the calculated trajectory. This animation is created using MATLAB's VideoWriter, which enables the generation of a video file that visually represents the MRAV's movement along the trajectory. The animation provides a dynamic and intuitive visualization of the system's behavior, enhancing the understanding of its performance and response to various inputs (controls and disturbances) and is presented in Appendix C.10.

The animation process begins by initializing the VideoWriter with the desired frame rate, ensuring smooth playback that accurately represents real-time motion. A loop is then employed to update the position and orientation of the MRAV for each frame based on the dynamics calculated in the simulation. This continuous update loop reflects the MRAV's real-time response, making the animation a representation of its actual behavior.

To maintain clarity and avoid drawing multiple instances of the MRAV, the 'cla' (clear axis) function is used to delete the previous frame before rendering the new one. This approach ensures that each frame of the animation shows only the current state of the MRAV, resulting in a clean and accurate video representation. The use of 'cla' prevents the overlap of multiple images, which could otherwise clutter the visualization.

By employing this method, the animation closely mimics an actual video, providing a clear and continuous depiction of the MRAV's trajectory. This real-time animation is not only visually appealing but also serves as a valuable tool for analyzing the performance and stability of the MRAV under different conditions. It allows for immediate visual feedback on the effectiveness of the control strategies and the impact of any external disturbances introduced during the simulation.

An output example of the 'Simulation Script' of a simple hover simulation, is presented in Appendix C.11. In the Appendix C.4 we can see a simulation example of an optimised hexarotor, performing a trajectory with waypoints p1=[0 0 0], p2=[0 0 5],p3=[5 5 5] in total time 36 s with external disturbance of 4N in x axis. The multirotor stops in each waypoint.

4.1.3 Extract Gazebo Model Script

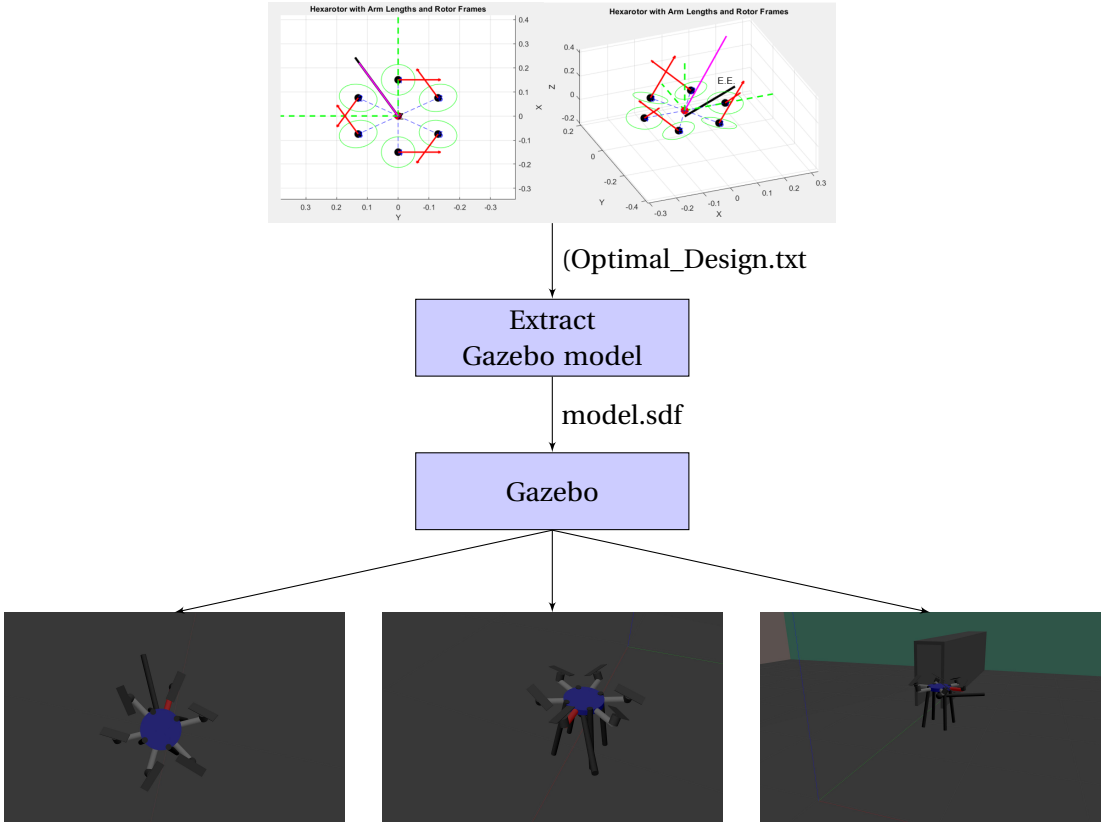
The "Extract Gazebo Model" script plays a crucial role in the workflow of this thesis by bridging the gap between design optimization and practical simulation. This script is designed to read design parameters from the aforementioned configuration file, "Optimal_Design.txt", and use these parameters to generate a Simulation Description Format (SDF) file for the MRAV. The SDF file is then used for simulating the MRAV in the Gazebo simulation environment, enabling rapid testing and validation of the optimal design.

The generated Gazebo model of the MRAV is a simplified representation, primarily composed of cylinders and spheres, as discussed in the "Optimal Design" section. This simplification facilitates efficient simulation without compromising the essential dynamics and interactions necessary for evaluation. The script itself is structured as a series of printing commands interspersed with loops. These loops utilize generalized formulas to accommodate different orientations and positions of the motors within the Gazebo simulator, ensuring the model's accuracy to the design found by the optimizer.

The development of the script involved an iterative process. Initially, a specific MRAV model was manually created in Gazebo. This model served as a reference for translating the design into a generalized format that could be automated using MATLAB. By automating the generation of the SDF file, the script allows for the quick adaptation and testing of new designs produced by the optimization script. This automation is particularly beneficial for evaluating multiple design iterations efficiently, thus accelerating the development process.

The primary purpose of this script is to enable fast testing of the optimal design in the Gazebo simulator. By streamlining the process of translating optimized design parameters into a simulation-ready format, the script facilitates immediate practical evaluation of theoretical designs. This immediate feedback loop is critical for refining and validating the optimization process, ensuring that the designs are not only optimal in theory but also practical and effective in real-world scenarios.

Further sections of this thesis will provide a detailed explanation of how the Gazebo simulator was utilized throughout the research. This includes a discussion on the setup of the simulation environment, the integration of the SDF files generated by the "Extract Gazebo Model" script, and the methods used for analyzing simulation results. By integrating these simulations into the design process, the research ensures a comprehensive approach to optimizing and validating the MRAV designs, as it will be shown in the next section.



One can see in the above nodes the same optimized design in gazebo, through the "Extract Gazebo Model" script. The red arm is the arm corresponding to the x axis of the body frame.

4.1.4 Baseline Designs

In this section, we evaluate the baseline fixed-tilted-propeller MRAV "Tilt-hex", optimized for F.A. and contact-based tasks[9]. It's initial optimization focused on tilt angles to enhance energy efficiency over a trajectory. For this evaluation, we maintain the existing tilt angles, but the drone's volume, particularly the arm lengths of 0.4 m, must be reduced to meet the dimensions of the ballast tank. The aim is to demonstrate that a simple approach — adjusting only one aspect, such as the volume, to meet task requirements — can lead to suboptimal results. Thus the aforementioned, systematic procedure that addresses all design aspects and task requirements, holistically, is needed.

The original "Tilt-hex" in the RAM lab has tilt angles of $\alpha = 30\,\text{deg}$ and $\beta = 10\,\text{deg}$[10], chosen to balance maximum lateral force generation and minimize internal force losses Ryll et al. (2017). These angles were derived from the energy efficiency optimization in Rajappa et al. (2015). Simulations showed the system could apply a maximum force of 6 N on the end effector (E.E.) angled at 30 deg relative to the x-axis, as detailed in the Appendix C.5.

[9] Specifically, we analyze the "Tilt-hex" from the RAM lab as a candidate for the AUTOASSESS project

[10] The tilt angles measured in the lab were $\alpha = 20°$ and $\beta = 0°$, optimized for F.A. tasks. However, since these angles do not meet the task's wrench requirement of 6N for contact, we opted to use the original values.

To fit the ballast tank volume constraints, we scaled the design down to arm lengths of 0.15 meters (like the designs proposed for the AUTOASSESS project). This new size allows an MRAV to navigate the corridors, in a F.A. way with minimum maneuvers, while maintaining a balance between propeller size (for payload capacity) and clearance from the walls. However, when the system rotates by 30 deg, as required by the E.E. for contact tasks, the side rotors would collide with the ballast tank walls, violating the volume constraints [11]. Moreover, the added maneuvers of rotations every time there is the need for contact, severely reduces the endurance of the system. Further scaling down the arm lengths to avoid collision would necessitate smaller propellers, which in turn reduces payload capacity and the MRAV's ability to generate the necessary wrench, as well as having a negative effect on the flight time.

Alternatively, adjusting the E.E. by 30 deg, to be parallel to the x-axis of the base, allows the system to meet volume constraints[12] , but simulations (see Appendix C.5) show the system can no longer apply the required 6 N task wrench along the x-axis at the E.E. tip. All the aforementioned, demonstrate that a naive approach of scaling or adjusting individual parameters to fit task requirements is insufficient. Instead, a systematic design procedure, which considers both the system and task holistically, is essential to achieve optimal solutions.

We conclude the development of the software framework and the systematic design procedure for translating task requirements into optimized MRAV designs. The process begins with a solid understanding of the task and a high-level conceptual design of the system. From there, the formal task requirements are mathematically formulated following the theoretical framework established and translated into inputs for the software. By carefully selecting the appropriate objective function components, constraints, and validation metrics, the software holistically considers the task requirements and the system and optimizes the design accordingly.

[11] Contact while passing through narrow corridors of 60x40cm to inspect the hull, perpendicular to the corridor
[12] No need to rotate for contact, as the E.E. is parallel to the heading of the drone (x-axis)

5 Simulations and Results

Using data obtained from the optimization process, we will virtually construct some MRAVs within the Gazebo simulation environment and perform contact on a wall. More specifically, we will try to replicate the AUTOASSESS application scenario, into Gazebo. The analysis of the results includes the visualization of the wrench space, the plotting of the desired force applied to the wall, the motor plots and comparison with the desired ones and evaluation of the specific requirements of the AUTOASSESS project.

The AUTOASSESS use case (see 1.2), requires a drone to navigate through a narrow corridor with dimensions of 40 cm in width and 60 cm in height, and ultimately make contact with a wall. These requirements are mathematically represented by constraining the robot's maximum width diameter to 40 cm and specifying a desired force vector $\mathbf{w}_{\text{des}} = [6\ 0\ 1\ 0\ 0\ 0]^T$, where the platform must be capable of exerting a maximum of 6N in the x-direction[1]. Although 6N is excessive for contact purposes, this is included for demonstration.

For both the example simulations and the proposed designs that will follow, it was decided to consider MRAVs with 6 AUs, because its the simplest system that can achieve full actuation. Two configuration options were considered for the drone's design: (D-A) and (D-C)[2]. In the first configuration (D-A), all 12 tilt angles were optimized while maintaining a symmetric hexarotor configuration. In the second configuration (D-C), both the tilt angles and the positions of the motors were optimized without any symmetry constraints. The design output and its force/torque space, for the first configuration (D-A) can be seen in Fig. 5.1:

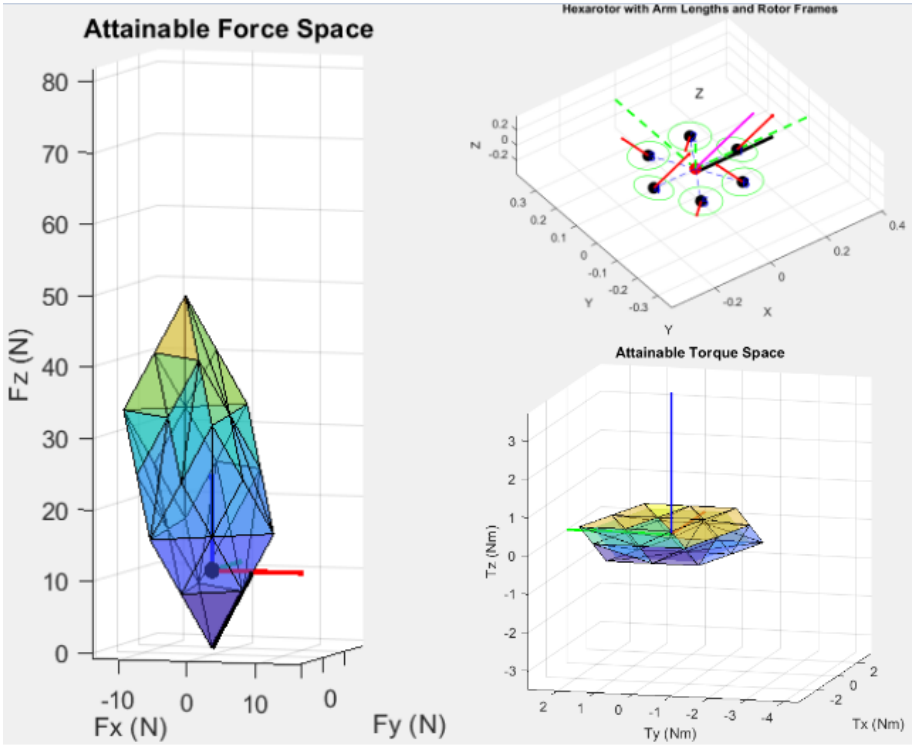


Figure 5.1: Force space (left) notice how the omnidirectionality is small, due to elongation of the wrench space on the x axis by the only desired wrench. Torque space (bottom right), notice how it's torque generation ability on the z axis is very limited

[1] It is not necessary to exert in the x direction, but it is the most effective.
[2] (COTA-CORP)

A top view of the design, as well as the tilt angles and several of the metrics can be found in appendix D.1. The Simulation Script in MATLAB for hover and a constant disturbance of 6N in the x-axis (simulating contact and push) indicates, that this is the maximum applicable disturbance the platform can handle. This result can be found in Appendix D.2.

Using the "Extract Gazebo Model" script, we virtually constructed the optimized hexarotor for the task. Along with the Gazebo simulation, we used Genom3 (see Openrobots (a)) with Telekyb (see Openrobots (b)) and the controllers `uavatt`[3] (see Openrobots (c)) and `uavpos`[4] (see Openrobots (d)), modified to account for the specific allocation matrix, mass, inertia, motors, and rotors. Controller tuning was subsequently performed. For asymmetric designs where motor positions are not symmetric, tuning the controller is particularly challenging. The gazebo simulation is shown in Fig. 5.2 and in Fig. 5.3, the resulting force plot from the logs of the drone is presented.

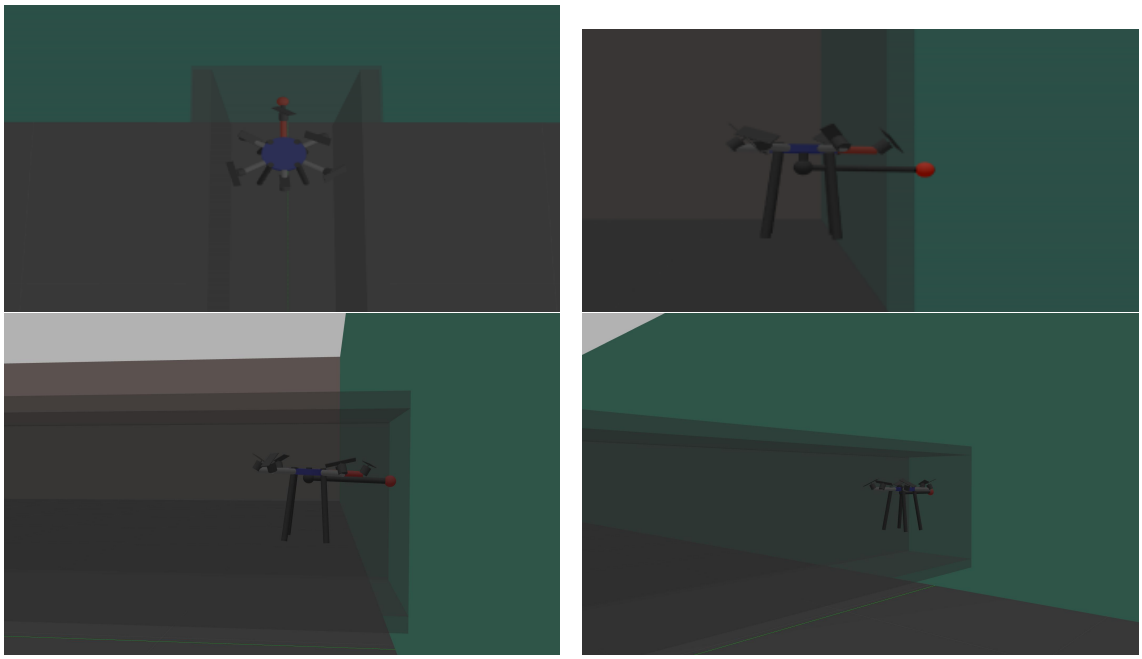


Figure 5.2: In the simulation, a 40x60cm corridor with the wall at the end representing the hull of the ship is shown. The corridor is made transparent. The drone passes through the corridor, contacts the hull, and pushes in an F.A. manner. Design D-A (right-top), Design D-C (bottom).

The force plot from Gazebo aligns with the findings of the MATLAB simulation and the optimizer. It is worth noticing that with larger forces applied, some of the motors saturated and the drone was unstable. Additionally, the metrics of more than 10 minutes of flight time with a 3.7Ah battery, a low condition number, adherence to the volume and wrench generation task requirements, and 90% efficiency, make this design an optimal choice for the current task as seen in D.1.

Reflecting on the limitations in "omnidirectionality" as seen from the force and torque spaces in D.1, one can easily deduce that these are not ideal when the platform encounters unexpected collisions or needs to execute unconventional maneuvers. However, given that we only specified a 6N force in the x-axis, the framework allows for the addition of more desired wrenches and reshaping of the wrench space as will be seen in the (D-C) design. By doing so, one can redesign the force and torque distribution to better meet the current task's requirements and enhance omnidirectionality. This flexibility is one of the key strengths of this framework, that will be taken into account when proposing the new designs for the AUTOASSESS case.

[3] UAV attitude flight controller. It implements the SO attitude controller described in Faessler et al. (2017).
[4] UAV position flight controller. It implements the $\mathbb{R}^3$ position controller described in Faessler et al. (2017).

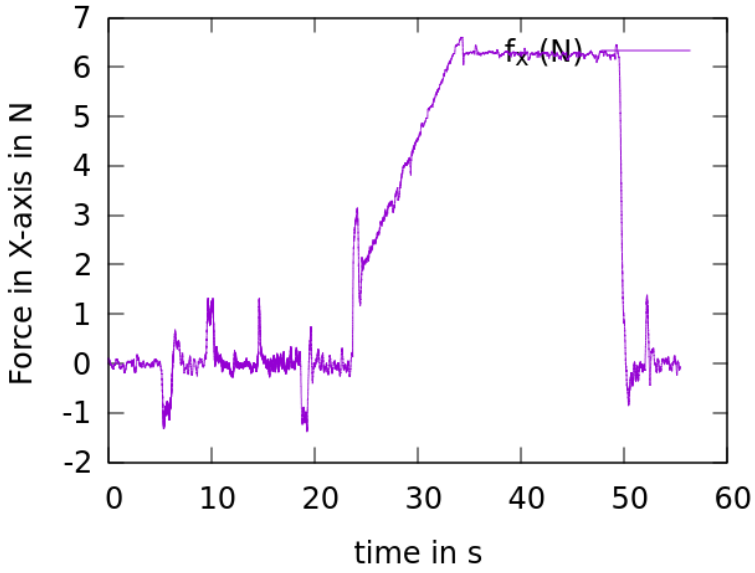


Figure 5.3: Contact on the "hull" and push with a force of 6N on the x-axis of the body frame (where the E.E. is).

For the second design (D-C), we set the desired wrenches to 6N along the x-axis and also aimed to enhance the minimum guaranteed force space to 3N, by redesigning it. To achieve this, we specified, additional, desired wrenched -3N in the negative x-axis and ±3N in the y-axis. The simulation results and design configuration are presented in Appendix D.3. This configuration differs significantly from the previous one, with varying arm lengths, angles, and individually optimized tilt angles for all 12 motors to meet the task's requirements. The design successfully flew through a narrow corridor measuring 40x60 cm and exerted the maximum force of 6N in the x direction during contact with the hull, as specified. From the motor plots of the commanded and measured inputs (see Appendix D.6), it is evident that they align well with the results from our developed simulation script (see Section 4.1.2), thus validating the script's accuracy (of course with added noise, as the simulation script assumes an ideal situation). In particular, when the design exerted the maximum force of 6N (for which the force space was tailored), two of the motors approached their operational limits. This is evident from the force space plot D.5, where the w_{des} lie on the boundary of the force space, causing motor saturation beyond this point. This observation is further confirmed by the control signals from the simulator script in Section 4.1.2. The only discrepancy between the commanded and measured input occurs when the controller commands two motors to stop rotating (0 Hz), but due to the lower limit set at 16 Hz for the motors, the measured input never drops below 16 Hz, as expected.

Both of the above simulation cases contribute to the research question regarding force space redesign. By choosing the right constraints and objective function, based on the aforementioned mathematical formulation and introducing additional desired wrenches geometrically shape the force and torque space, either expanding or contracting its boundaries based on the task's wrench requirements. This malleability and redesign of the wrench space are more evident when one is choosing to increase the design variables to be optimised i.e. in the cases D-A, D-B, and D-C.

5.0.1 AUTOASSESS Use Case

Identification of Thrust and Drag Coefficients for AUTOASSESS Use Case

Before using the optimization framework, because the ultimate goal is building one of the optimized designs, generated for the AUTOASSESS use case, it is crucial to input realistic values for the coefficients of thrust and drag of the propellers. Accurate values lead to more realistic

final designs, particularly because the aforementioned quantities have a significant impact on the wrench space. To identify the thrust and drag coefficients, a well-planned experiment was conducted, for multiple sets of unidirectional propellers[5]. Additionally, the experiment measured the maximum current and rotational speeds of the propellers, essential parameters that influence the overall design.

The experiment was set up using a combination of hardware and software components. The hardware included a custom-built test bench equipped with a Force/Torque (F/T) sensor and its associated peripherals. The test setup also incorporated an "XNOVA Lightning 08-25"[6], an Electronic Speed Controller (ESC) (max 32 Amps), a "Teensy" micro-controller, and 6 sets of propellers, including one bidirectional set. On the software side, a custom script was developed to control the motor's velocity, which was crucial for the experiment. This script was integrated with modified team scripts accessed via GitLab, all running under the Pocolibs and Genomix frameworks. MATLAB was employed for interpreting the results, particularly for interpolation and fitting, which are essential in accurately determining the coefficients. The setup can be found in the Appendix D.7.

The strategy behind the experiment was straightforward but effective. The motor was subjected to incremental velocity steps, and the thrust and torque produced by each propeller were measured. The primary objective was to find the thrust coefficient c_f that minimizes the following, using bisquare robustness to ensure the accuracy of the fit[7] (MATLAB was used):

$$\min_{c_f} \quad \|\mathbf{W} \cdot (\mathbf{T} - c_f \cdot \boldsymbol{\omega}^{\mathbf{2}})\|^2$$

, with $\mathbf{W}$ a diagonal matrix of weights that are applied to each individual residual. A similar approach was applied to determine the drag coefficient. To ensure that the results were representative, 2 to 3 propellers from each set were sampled, and the average thrust and drag coefficients were calculated for each set.

Several critical factors were taken into account during the experiment to ensure the accuracy and reliability of the results. One significant consideration was the ground effect. The propellers, due to their smaller size, were positioned close to the F/T sensor—approximately 1 to 1.5 times the radius of the propellers. This proximity could introduce measurement errors due to ground effect (we expected to see more thrust with less current), so it was carefully monitored. Another important factor was the bias of the F/T sensor, which was reset to zero before each new experiment. To further enhance accuracy, the bias signal of the F/T sensor was measured over a few seconds to calculate an average, which was then subtracted from the experimental data to correct for any systematic error.

The experiment also accounted for the maximum DC current that the motor and ESC configuration could handle, which determined to be 30 Amps, by taking the lower value between the prior modules specifications for max DC current. This value was critical because it dictated the maximum velocity each propeller could achieve under continuous operation. To measure the maximum current, a current clamp was used for DC current measurements. Small input velocity steps were applied incrementally until the maximum continuous current was reached, thereby determining the maximum rotational speed for each propeller. Different sampling times for the components were also considered in the analysis to ensure that the results were not affected by timing discrepancies.

The results of the experiment were highly satisfactory and aligned well with expectations. The force and drag coefficients for the different propeller sets were determined with an error margin

[5] Acquired as potential candidates for the AUTOASSESS use case.

[6] See Lightning (2024)

[7] A method that minimizes a weighted sum of squares, where the weight of each data point depends on its distance from the fitted line (Valyon and Horvath (2007))

of less than 5% between propellers of the same set, indicating the experiment was conducted with precision. However, it is worth noting that in one set of propellers, the bias of the F/T sensor was not properly canceled out, resulting in a 0.7 N bias in the force measurement. Regarding the maximum rotational speed ω_{max}, since current was increasing rapidly between 20 and 30 Amps, relative to the propeller's rotational velocities,the values found in the table D.1 correspond to an average current of 25 Amps. A table of all identified Thrust and Drag coefficients can be seen in

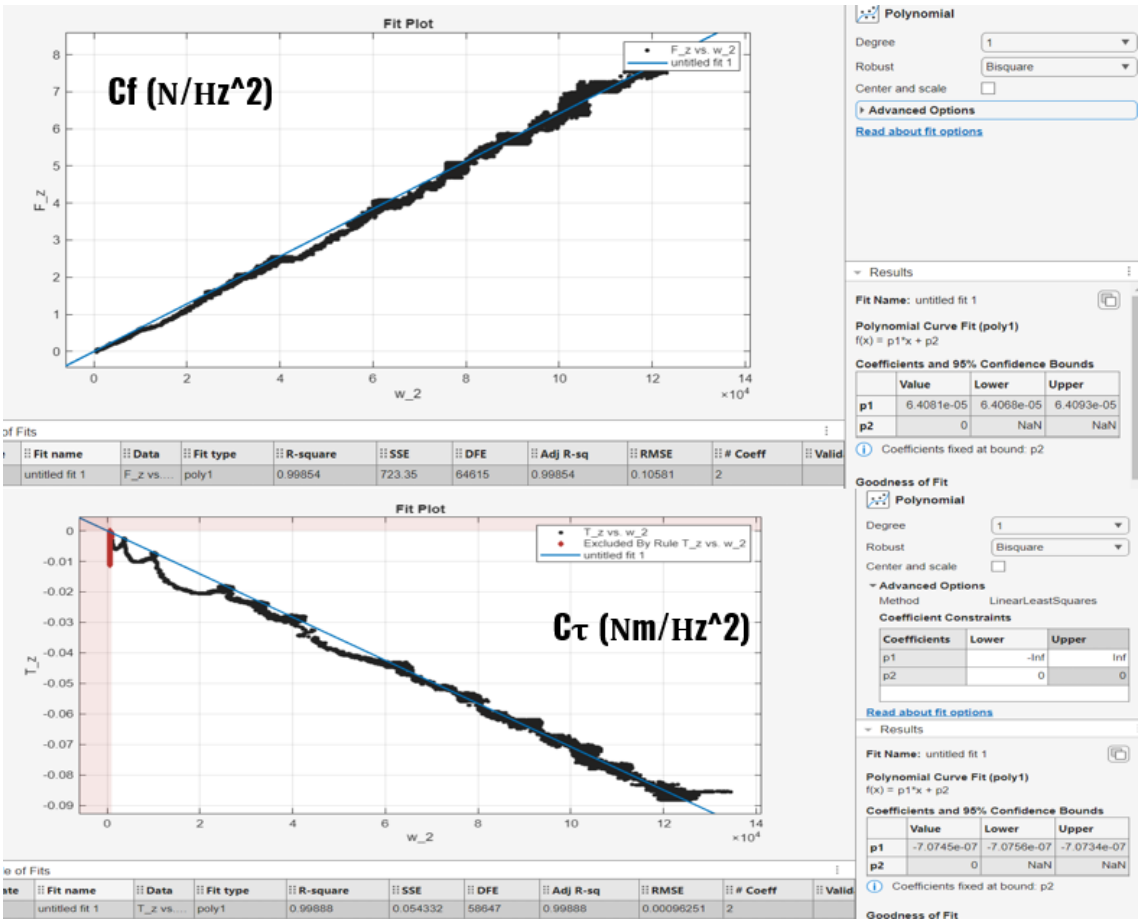


Figure 5.4: Example of fitting for a Propeller of a Set for c_f(up) and c_τ (down). Goodness of fit can be seen in the bottom.

appendix D.1. The identified thrust and drag coefficients, along with the maximum rotational speeds, will be used to enhance the accuracy of the input values in the optimization framework. This, in turn, will contribute to the development of more realistic and optimized designs for the AUTOASSESS use case.

AUTOASSESS Designs

Following the aforementioned experiment, the "GEMFAN Hulkie 5055-3" propellers were chosen (values in Appendix D.2).In order to make the final output design as realistic as possible, we estimated some components that would potentially be used to build the platform. The minimum vertical generated thrust is determined by the total estimated weight of the MRAV. In our case, we weighted in the lab the components and we present them in the table in Appendix D.2.

The maximum horizontal generated wrench, with 0 torques applied, was determined as $|F_h| = 6N$, for flying and contact inspection purposes. Although, the probe with which the contact is gonna be made requires much less force, it was determined like that in order to show the capabilities of the framework as well. Something like this has big impact on the Flight Endurance and Force Efficiency index.

To achieve design simplification and ease of construction, we utilized the Optimal Design software to generate multiple configurations based on selected parameters. Our focus was optimizing designs for the configurations "(A through D)-A" and "D-B," which correspond to specific tilt angles and rotor positions, respectively. Notably, the optimization for the "D-C" configuration was intentionally excluded from our analysis, because although in theory these designs are feasible, in reality, their control and manufacture are part of a larger work, out of the scope of this project.

For the "(A through D)-A" configuration modes, the End Effector (E.E.) was positioned parallel to the x-axis of the body frame. This configuration was selected to maintain design simplicity while maximizing the platform's dimensions to navigate the narrow corridors, but with also keeping a clearance of almost 2 cm from the walls, given that the arm lengths are uniform and relatively short. If the E.E. were positioned between rotors 1 and 2, forming a 30-degree angle with the x-axis (well-seen configuration in the literature), the arm lengths would need to be reduced (from 0.15m). This would result in a smaller platform, which would decrease the maximum payload the platform. That is because the platform, due to the orientation of the E.E. with respect to the body frame, would have to pass through the narrow corridors of the ballast tanks angled, meaning that the maximum y-axis radius of the platform would be 10.5 cm (with 15 cm arm length), which does not meet the volume constraints (10 cm radious). In general, since the optimization takes place after choosing the configuration of the E.E., the tilt angles will be optimised for its position. On the other hand if after the optimisation, one changes the position of the E.E. thus the point of contact, the platform most probably will not have the same wrench generation performance[8]. This is due to the fact that the wrench space is not a sphere and in different orientations has different force and torque generation capabilities. The framework optimises for these directions, by redesigning the force space.

For the "D-B" configuration, the E.E. was aligned parallel to the x-axis of the body frame as well. As the objective here is to minimize the y-axis component of the rotor positions, resulting in a design that is elongated along the x-axis and narrower along the y-axis, to pass through the narrow corridors of the ballast tanks, this E.E. alignment with the x-axis, was strategically selected. The parallel positioning of the E.E. in this case is crucial to avoid angular displacement, which would compromise the design's intended geometric characteristics.

The input parameters for the optimization, including the coefficients of thrust and drag for the propellers and the specifications of the motors, were based on data obtained from experiments conducted in the RAMs flight lab as mentioned before.

In the objective function term for assessing the control efficiency of the design over a specified trajectory, we followed a trajectory otherwise impossible for standard co-planar hexarotors and inspired by the trajectory described in Rajappa et al. (2015), used to assess reorientation while hovering and reacting to external force/torque disturbances and 6-DoF trajectory tracking. Note that a simpler trajectory than the one selected can introduce smaller tilt angles, more energy efficient platform (but with a compromise in maneuverabilily, e.g. the wrench space would be different). The selected trajectory involves the hexarotor maneuvering to an orientation of $-12°$ with respect to the X_B axis, $12°$ with respect to the Y_B axis, and $15°$ with respect to the Z_B axis, all while hovering in the fixed position $\mathbf{p} = (0,0,0)$. The initial conditions for the hexarotor were set as $\mathbf{p}(t_0) = \mathbf{0}$, $\dot{\mathbf{p}}(t_0) = \mathbf{0}$, $\mathbf{W}_{RB}(t_0) = \mathbf{I}_3$, and $\boldsymbol{\omega}_B(t_0) = \mathbf{0}$. The desired trajectory was defined as $\mathbf{p}_d(t) = \mathbf{0}$ and $\mathbf{R}_d(t) = \mathbf{R}_X(\phi(t))\mathbf{R}_Y(\theta(t))\mathbf{R}_Z(\psi(t))$, with $\phi(t)$, $\theta(t)$, and $\psi(t)$ following a smooth profile. In general simpler trajectories can be used such as simple hovering, or sinusoidal trajectories for the θ and ϕ angles like in the aforementioned paper. Here we decided to use this trajectory to optimize our design in a manuever that only a fully actuated hexarotor could do. The results and proposed designs are the following 3 in 5.1 table.

[8]Exactly as we saw for the baseline design.

Table 5.1: Comparison of all three designs with their respective alpha, beta angles, and arm lengths

Motor	Design 1		Design 2		Design 3	
	Alpha (°)	**Beta (°)**	**Alpha (°)**	**Beta (°)**	**Alpha (°)**	**Beta (°) | Arm Length (m)**
1	25.65	-14	-0.024	29.024	27.247	-13.748 / 0.186
2	-25.65	14	-9.3025	-35.786	-28.530	0.817 / 0.129
3	25.65	-14	3.2414	20.0347	32.097	-13.134 / 0.135
4	-25.65	14	0.02416	-29.024	-27.247	13.746 / 0.186
5	25.65	-14	9.3027	35.787	28.531	-0.820 / 0.129
6	-25.65	14	-3.2415	-20.0366	-32.097	13.131 / 0.135
Arm Length (m)		0.150	0.150		-	

Table 5.2: Metrics for all three designs (Control Effort at the bottom for the aforementioned trajectory)

Metrics	**Design 1**	**Design 2**	**Design 3**
Rank of Allocation Matrix	6	6	6
Maximum Endurance (min)	11.4	11.6	11.15
Optimal Speed for Maximum Endurance (m/s)	11.45	11.45	11.46
Flight Time for Maximum Range (min)	9.5	9.5	9.2
Optimal Speed for Maximum Range (m/s)	20.5	20.5	20.54
Maximum Range (km)	11.5	11.8	11.3
Force Efficiency Index	0.86	0.88	0.85
Condition Number	6.5	8.75	6.04
Min. Guaranteed Force (N)/Torque (Nm)	2.9 / 1.25	3 / 0.25	3.14 / 1.17
Balanced Distribution of Thrusts	2.330e-4	1.88e-4	2.336e-4
Control Effort	1.83e+6	1.7e+6	1.75e+6

The three designs with their corresponding force and torque spaces can be found in Appendix D.8, D.9 and D.10. For all three of them, simulations of "reorienting while hovering" and "hovering with external disturbance (Max attainable)" using the Simulation Script were done and presented in Appendix D.3.3. The results for all three designs show stable designs, able for 6D trajectory tracking and as the requirements imposed, able to withstand disturbances up to their desired wrenches (here 6N).

All three designs are considered optimal within the context of the AUTOASSESS use case, as they satisfy the key requirements, including energy efficiency (force efficiency index for all designs almost 90%), the ability to navigate within the dimensional constraints of the ballast tank, maintaining a full rank of the allocation matrix, and providing sufficient force (6N in the x-axis) and torque for contact inspections and payload requirements. In terms of the latter, all three designs can lift a maximum of up to 5 times their weight (maximum $\tilde{5}0$ N), making them ideal candidates for carrying a variety of sensors and equipment, if one considers their compact size.

Design 1 is characterized by simplicity and symmetry, with fixed arm lengths and symmetric motor angles. It offers a good balance of performance metrics, including an endurance of 11.4 min, a force efficiency index of 0.86, and a low condition number of 6.5, indicating stable and manageable control. The balanced approach of Design 1 makes it a solid all-round choice, particularly in scenarios where stability and simplicity are prioritized.

Design 2 outperforms the other designs in terms of endurance (11.6 minutes), making it the best choice for missions requiring extended flight time and coverage. However, this comes at the cost of a higher condition number (8.75), suggesting that control stability is more compromised, but not necessarily challenging. Despite its high force efficiency index of 0.88, it's limited

minimum guaranteed torque generation capabilities (that comes in the yaw) may compromise it in situations where abrupt maneuvering is critical. Note that in this design we effectively redesigned the force space by adding more desired wrenches (-3N and 3N on y-axis and -3N on x-axis), making the minimum guaranteed generated force significantly improved (3N) than the design in 5.1.

Design 3 offers flexibility with variable arm lengths and asymmetric motor configurations. While it provides slightly lower endurance and range compared to Design 2, it compensates with a better condition number (6.04), indicating more stable control. Its force efficiency index is slightly lower at 0.85, but the design's adaptability and its adequate omnidirectional wrench generation capabilities make it a strong candidate for the task.

In summary, while all three designs are optimal for the task, they cater to different operational priorities. Design 1 is ideal for straightforward, stable operations; Design 2 excels in endurance and energy efficiency; and Design 3 offers a balanced approach with a focus on control stability and adaptability, making it potentially the best choice for tasks requiring a blend of efficiency and maneuverability. With larger battery[9], smaller desired wrench and optimizing of different maneuver, their endurance can skyrocket. Given the ease of manufacture, we chose to proceed with Design 1, and its Gazebo simulation will be presented in the next paragraph.

Gazebo Simulation of the proposed AUTOASSESS "Design 1" For the simulation of this design, we chose to do a flight mission through a narrow corridor of 40x60cm and contact and push of 5N on the "hull", demonstrating its capabilities. The results can be seen here:

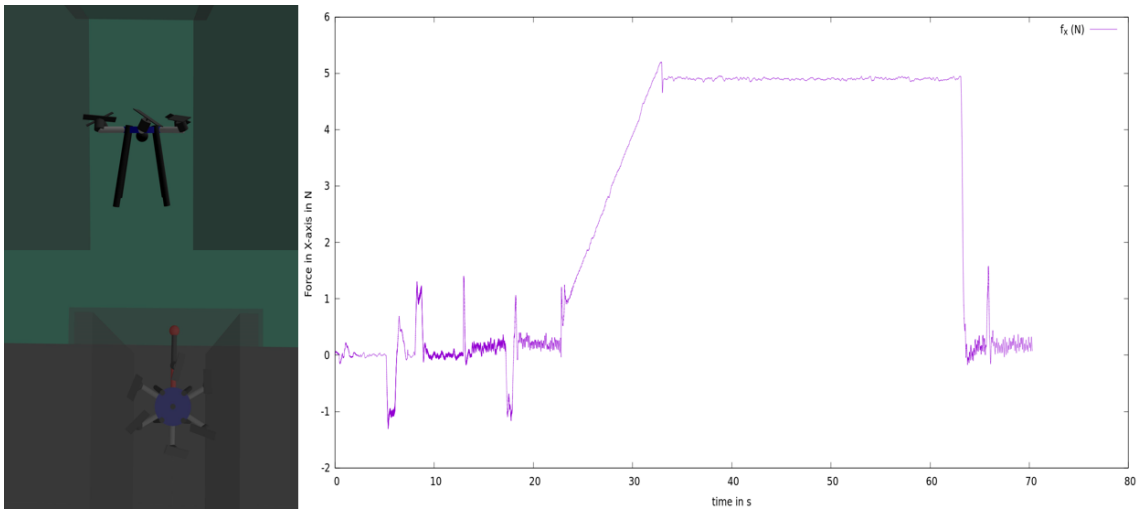


Figure 5.5: Contact and push of 5N of the proposed Design 1, for the AUTOASSESS use case

It is evident that the first contact occurs at 24 seconds. Following this, the drone gradually increases its force, reaching 5N with a slight overshoot, and then maintains this force with minimal oscillations for 30 seconds. Afterward, it detaches from the wall and returns to a waypoint. The small spikes of approximately $\pm 1N$ before and after contact corresponds to waypoints the system follows from takeoff until reaching the contact location. Each Gazebo simulation for a new design requires comprehensive tuning of the "uavatt" and "uavpos", controllers from scratch, involving numerous tests. These tests begin with basic hovering, progress to flight missions, and culminate in contact missions. As the final step before the physical realization of the design, Gazebo simulations incorporate all the parameters of a real drone, including potential noise and interactions between modules, which can deviate from

[9] Easy to carry due to their payload lifting capabilities

the ideal conditions simulated in the "Simulation" script. Having said that, so far Gazebo simulations only validate the framework proposed through its designs.

Implementation of the proposed AUTOASSESS "Design 1"
The proposed "Design 1" (see Appendix D.8) was finally implemented and can be seen in Fig. 5.6. It adheres exactly to all the parameters proposed by the framework. Successful hover tests have already been conducted, with flight and point contact tests to be performed in the near future.

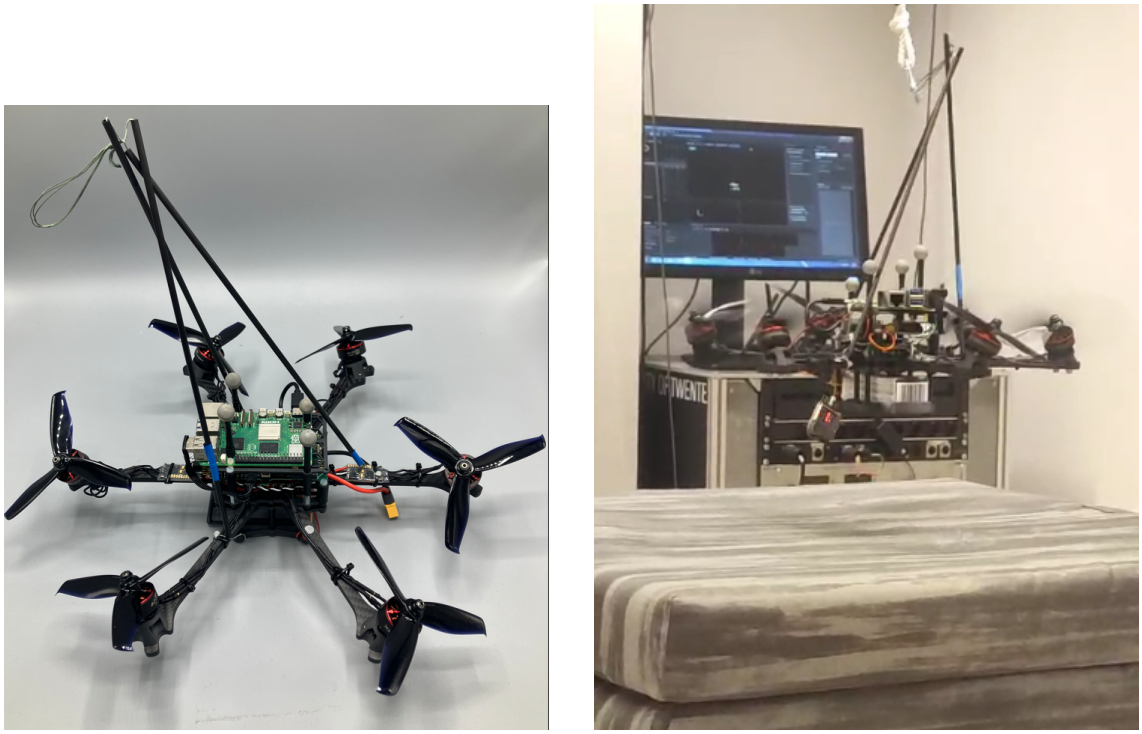


Figure 5.6: Optimised Design 1, proposed for the AUTOASSESS project (left). Hover Test (right).

The vehicle consists of a PAPARAZZI CHIMERA flight controller, RASPBERRY PI as high-level computer handling communication with the PC, 6 brushless motors 'Xnova Lightning V2' 2208-2500 with 2500KV, 6 GEMFAN HULKIE 5055S propellers, one 4x1 ESC AK32 V3 35A and two 35A ESCs. As battery for the tests, a four-cell 2300 mAh LiPo[10] was used.

[10]Battery : https://droneshop.nl/tattu-2300mah-14-8v-75c-4s-lipo-battery-xt60

6 Discussion

6.1 Addressing the Research Questions

6.1.1 RQ1

As mentioned in section 3 when addressing this research question, it is essential to acknowledge that there is no singular, definitive answer. Instead, a constellation of factors plays a crucial role in the development of each design, depending on the specific tasks at hand. Following an extensive review of the existing literature, it has been established that the metrics delineated in Section 3.2 will serve as the primary criteria for evaluating the suitability of a fixed-tilted MRAV design for a generic set of tasks. These metrics, along with their mathematical formulations, have been meticulously developed to provide a comprehensive assessment framework. The aforementioned section 3.2, elaborates on these primary metrics, ensuring clarity and precision in their definition and application. This detailed exposition facilitates a robust evaluation of fixed-tilted MRAV designs, effectively providing a benchmark of metrics to consider when proceeding with such applications. By adhering to these established criteria, the suitability of a fixed-tilted MRAV design for a specific task or set of tasks can be comprehensively assessed, thereby enhancing the overall efficacy and reliability of the design process.

6.1.2 RQ2

The question of how to propose a design based on task requirements is, indirectly but continuously, addressed throughout this research. While the straightforward answer might point to the theoretical and software framework developed, the actual approach runs deeper.

To be more specific, the methodology we proposed is optimization-driven, beginning with the recognition that the final design must be the result of a well-structured optimization process, formulated as a nonlinear programming (NLP) problem that holistically incorporates all task requirements and systems limitations. Serving as the cornerstone of this process is a thorough understanding of the task requirements and their mathematical representation, as well as knowledge of the individual modules (that the system is comprised of) limitations. These requirements and limitations serve as key inputs for the optimization problem and output a system that is reliable, realistic, and tailored to the task. With the addition of the careful selection of objective function components, that capture critical performance aspects of the system, along with system and task constraints and validation metrics, the software effectively binds these elements and allows for a systemic optimization, that outputs a design. Due to the formulation of the design problem as an NLP problem, the last step in this generalised procedure is the comparison of optimized designs, by the designer, and selection of the optimal, as mentioned in the Theoretical Framework section.

6.1.3 RQ3

This question is addressed in Section 3.1.1, where the mathematical representation of task requirements and constraints is elaborated. In this subsection, we analytically define a task and its mathematical representation, including the components it comprises, such as the task wrench. The derivation of the mathematical formulation of the task requirements is presented in a structured and clear manner, supporting the theoretical foundation of the developed software. This approach ensures that the theoretical concepts are effectively translated into practical applications, thereby facilitating the development of robust and optimized MRAV designs.

6.1.4 RQ4

As demonstrated throughout this thesis and the accompanying software, achieving any required-task-related improvements involves multiple factors. Redesigning the force space can fulfill certain requirements but may fall short in others. For instance, elongating the force space in

a specific direction to meet a task wrench requirement may fulfill that particular requirement but significantly reduce flight endurance due to increased internal forces caused by large tilting angles in that direction. Improvements are fundamentally a tradeoff between different task requirements. Ultimately, the optimal design for a set of tasks represents an optimal balance between these requirements. Focusing solely on redesigning the force space addresses only one aspect of the problem and in scenarios with multiple task requirements that may be orthogonal to each other, such as generating a wrench in a specific direction (assuming full actuation) versus maximizing endurance (where improving one can compromise the other), even the most efficient redesign of the force space can only partially achieve the required improvements. Thus, as this research suggests, a holistic approach that considers various perspectives and balances the tradeoffs is necessary to attain the optimal design for MRAVs. Let's see that in action. We consider the aforementioned proposed "Design 2" (see Appendix D.9) and an improvement of "generation of 1Nm torque around + z-axis while hovering" or to a desired wrench of $\mathbf{w_{des}} = [0\ 0\ 1\ 0\ 0\ 1]^T$. Moreover, for a system of radious $\sim 0.2m$, this torque can be translated to a tangential force [1], applied at the edge of the system of $\sim +5N$, which is sufficiently larger than the 3N in the positive y-axis. Thus in this targeted redesign, we expect larger tilt angles especially $\boldsymbol{\alpha}$, to achieve the improvement of a larger generation of force in the xy-plane, but also a decrease in the endurance due to the increase of internal forces. The redesigning and results can be seen in appendix E.1.

One can easily observe, that the results align with our expectations. The configuration of the tilt angles is changed to account for the torque around z with significantly larger $\boldsymbol{\alpha}$ tilt angles. Moreover, the endurance metric went down, from 11.6 mins to 11 mins, validating the increased presence of internal forces, as well as the control effort increased from 1.7e+6 to 1.8e+6. The wrench space was effectively redesigned with a noticable increase of both the minimum guaranted force and torque generation in all directions with the first increasing from 3N to 3.6N and the latter from 0.25Nm to 1.28Nm and a max attainable torque in the z-axis of 1.8 Nm, proving not only the effective redesign of the wrench space with this method but also validating the previous claims, these of trade-offs.

6.1.5 RQ5

An appropriate design procedure, that was followed as well in this research, to redesign the force space, not only in the specific case of AUTOASSESS, involves the knowledge of the task, the careful selection of the cost functions and constraints like the ones that have been derived in this research and especially by introducing, as done here through their mathematical representation, the desired wrenches ($\mathbf{w}_{\text{des}}$) as optimization targets. By strategically adding and adjusting $\mathbf{w}_{\text{des}}$, we can define the boundaries of the wrench space accurately. The more $\mathbf{w}_{\text{des}}$ are specified, and the more accurately they reflect the desired force space, the closer the final design will align with the task specifications. Multiple simulations and examples like in Section 6.1.4 validates this procedure, which allows for stretching or compressing the force and torque space, effectively redesigning it according to specific requirements. This ability to tailor the force space to the needs of a particular task is a significant advantage of this systematic procedure, which efficiently translates task requirements into an optimized design. Summarizing the procedure in steps:

- Accurate knowledge of the tasks requirements
- Careful selection of the cost functions and constraints
- Setting desired wrenches as optimization targets
- Evaluate the result based on the metrics[2]

These steps falls under the umbrella of the holistic approach that is introduced by the framework that this research contributed.

[1] on the xy-plane

[2] especially minimum guaranted wrench generation

7 Conclusion and future work

This thesis effectively achieved both of it's primary goals: first being the derivation of a generalised, systematic, design procedure from the formal requirements of a task, to an optimized Multirotor Aerial Vehicle (MRAV) configuration; and second through the use of this framework, the proposal and implementation of an optimized design for a unidirectional-thruster fixed-tilted propeller aerial robot, tailored for ballast tank contact inspections as part of the AUTOASSESS project. At the core of this approach is the formulation of the design problem, as a constrained multivariable nonlinear optimization problem (NLP), allowing the holistic consideration of task requirements through constraints or components of a unified objective function, derived from a combination of modified cost functions from the literature. The task requirements, serve as input to the MATLAB-based software, developed using CasADi, with IPOPT employed as the NLP's solver. The software outputs optimized MRAV designs tailored to the tasks and with a carefully selected set of performance metrics, the user is able to evaluate and compare between them, in order to make informed decisions about the optimal configuration, based on task requirements and design performance.

The theoretical framework presented in this thesis successfully provided a robust mathematical representation of a task, incorporating key requirements inspired by the AUTOASSESS project, such as endurance, wrench generation, and size constraints. It also delivered a proof of the convexity of the wrench space for generic unidirectional-thruster MRAVs, which facilitated the fast generation of the force and torque spaces by the software. Furthermore, the necessary metrics when working with fixed-tilted MRAVs were established, and the design problem was formulated as NLP problem. Consequently, the software framework developed, demonstrated wide flexibility and realism in terms of input parameters and design variables to be optimized. It provided extensive configuration options (COTA-CORP) and the ability to design the wrench space of the final MRAV by incorporating desired wrenches. The software offered a comprehensive range of output information about the optimized design, including visualizations of solution convergence, the design itself, the force and torque spaces and more. Additionally, it visualised the minimum guaranteed force and torque generation of the design, as well as various performance metrics. The framework also featured simulation capabilities through a custom-developed simulation software that allowed for waypoint trajectories and disturbance scenarios. Moreover, the software's ability to generate SDF files enabled quick model creation in Gazebo for further simulations of the optimized design.

The framework combined, contributes to the literature, as a systematic, unified framework, that directly ties task requirements, to design decisions and generates optimized unidirectional-thruster MRAVs, regardless of the number of actuator units. The robustness of it, was validated through a series of simulations and real-world experiments, where optimized MRAV designs successfully met the strict size, maneuverability, wrench generation, and endurance requirements for confined ballast tank spaces. The final implementation and testing of the AUTOASSESS proposed design, further reinforced the effectiveness and practicality of the framework.

Future work will focus on further exploring the weights of the objective function, to refine the balance between the various components. Moreover, there will be conducted additional flight and contact tests on the real drone developed for the AUTOASSESS project, to validate its performance in real-world conditions. Tuning the controllers for asymmetrical MRAV designs will be another priority, ensuring better adaptability for diverse configurations. Finally, the aim is to continue enhancing the framework's generality and robustness, allowing it to support a wider range of tasks and MRAV designs with increased flexibility and efficiency.

A
Appendix: Proof of convexity of the wrench set

Given an input set convex

$$A = \{\boldsymbol{u} \in \mathbb{R}^n \mid u_{min} \preceq \boldsymbol{u} \preceq u_{max}\}$$

and

$$\mathbf{F} : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

$$\boldsymbol{u} \mapsto \boldsymbol{w} = \mathbf{F}\boldsymbol{u}$$

linear map

$$J(\boldsymbol{w}) = \{\boldsymbol{u} \in \mathbb{R}^n \mid \boldsymbol{u} = \mathbf{F}^{\dagger}\boldsymbol{w} + \mathbf{N}\boldsymbol{\lambda}, \boldsymbol{\lambda} \in \mathbb{R}^l\}$$

and l=dim(Ker(**F**)). We assume here $m \leq n$ and $l = n - m$ (full rank)

For a set

$$\mathbb{B} = \{\boldsymbol{w} \in \mathbb{R}^m \mid \boldsymbol{w} = \mathbf{F}\boldsymbol{u}, \forall\, \boldsymbol{u} \in \mathbb{A}\}$$

Proposition: B is convex, i.e.

$$\forall\, \boldsymbol{w_1}, \boldsymbol{w_2} \in \mathbb{B}, \boldsymbol{w}(\boldsymbol{u}) = \alpha \boldsymbol{w_1} + (1-\alpha)\boldsymbol{w_2} \in \mathbb{B}, \forall\, a \in [0,1]$$

Proof: First, we will show convexity of $\mathbb{A}$:

To prove that the set

$$\mathbb{A} = \{\boldsymbol{u} \in \mathbb{R}^n \mid u_{min} \preceq \boldsymbol{u} \preceq u_{max}\}$$

is convex, we need to show that for any two vectors $\boldsymbol{u}_1, \boldsymbol{u}_2 \in A$ and any $\alpha \in [0,1]$, the convex combination $\alpha\boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2$ is also in $\mathbb{A}$.

Given:

$$u_{min} \preceq \boldsymbol{u}_1 \preceq u_{max}$$

and

$$u_{min} \preceq \boldsymbol{u}_2 \preceq u_{max}$$

We need to show:

$$u_{min} \preceq \alpha\boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \preceq u_{max}$$

First, let's show the lower bound:

$$u_{min} \preceq \alpha\boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2$$

Since $\boldsymbol{u}_1 \in A$ and $\boldsymbol{u}_2 \in A$, we have:

$$u_{min} \preceq \boldsymbol{u}_1 \implies \alpha u_{min} \preceq \alpha\boldsymbol{u}_1$$

$$u_{min} \preceq \boldsymbol{u}_2 \implies (1-\alpha)u_{min} \preceq (1-\alpha)\boldsymbol{u}_2$$

Adding these inequalities:

$$\alpha u_{min} + (1-\alpha)u_{min} \preceq \alpha\boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2$$

$$u_{min} \preceq \alpha\boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2$$

Next, let's show the upper bound:

$$\alpha \boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \preceq u_{max}$$

Since $\boldsymbol{u}_1 \in A$ and $\boldsymbol{u}_2 \in A$, we have:

$$\boldsymbol{u}_1 \preceq u_{max} \implies \alpha \boldsymbol{u}_1 \preceq \alpha u_{max}$$

$$\boldsymbol{u}_2 \preceq u_{max} \implies (1-\alpha)\boldsymbol{u}_2 \preceq (1-\alpha) u_{max}$$

Adding these inequalities:

$$\alpha \boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \preceq \alpha u_{max} + (1-\alpha) u_{max}$$

$$\alpha \boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \preceq u_{max}$$

Combining the results, we have:

$$u_{min} \preceq \alpha \boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \preceq u_{max}$$

Thus, $\alpha \boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \in \mathbb{A}$, proving that $\mathbb{A}$ is convex.

Then for:

$$\boldsymbol{w_1} \in \mathbb{B} \Leftrightarrow \exists \boldsymbol{\lambda}_1 \mid \mathbf{F}^\dagger \boldsymbol{w_1} + \mathbf{N}\boldsymbol{\lambda}_1 = \boldsymbol{u_1} \in \mathbb{A}$$

$$\boldsymbol{w_2} \in \mathbb{B} \Leftrightarrow \exists \boldsymbol{\lambda}_2 \mid \mathbf{F}^\dagger \boldsymbol{w_2} + \mathbf{N}\boldsymbol{\lambda}_2 = \boldsymbol{u_2} \in \mathbb{A}$$

$\mathbb{A}$ is convex thus:

$$\boldsymbol{u} = \alpha \boldsymbol{u}_1 + (1-\alpha)\boldsymbol{u}_2 \in \mathbb{A} \implies \mathbf{F}\boldsymbol{u} \in \mathbb{B}$$

We write $\boldsymbol{u}$ explicitly as

$$\boldsymbol{u} = \alpha \mathbf{F}^\dagger \boldsymbol{w_1} + (1-\alpha)\mathbf{F}^\dagger \boldsymbol{w_2} + \mathbf{N}(\alpha \boldsymbol{\lambda_1} + (1-\alpha)\boldsymbol{\lambda_2}) \in \mathbb{A}$$

$$\mathbf{F}\boldsymbol{u} = \alpha \mathbf{F}\mathbf{F}^\dagger \boldsymbol{w}_1 + (1-\alpha)\mathbf{F}\mathbf{F}^\dagger \boldsymbol{w}_2 + \mathbf{F}\mathbf{N}(\alpha \boldsymbol{\lambda_1} + (1-\alpha)\boldsymbol{\lambda_2}) \in \mathbb{B}$$

$$\mathbf{F}\mathbf{N}(\alpha \boldsymbol{\lambda_1} + (1-\alpha)\boldsymbol{\lambda_2}) = \mathbf{0}$$

Since $\mathbf{F}$ is full rank and $m \leq n$ we have that $\mathbf{F}\mathbf{F}^\dagger = \mathbf{I}_m$

Ultimately:

$$\mathbf{F}\boldsymbol{u} = \alpha \boldsymbol{w_1} + (1-\alpha)\boldsymbol{w_2} \in \mathbb{B}$$

Thus B convex.

B
Appendix: Theoretical Framework

B.1
Appendix: Selection of design variables

In order to prove that $\nabla \mathbf{T} \neq 0$, we need to show that at least one of the partial derivatives in each component vector is not 0. This is done by showing that the sets $\mathbb{V}_{T_i}$ and $\mathbb{W}_{T_i}$ are not constant with respect to the respective variables $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}$. Let us omit (i) for the moment and focus in the task T.

$$\mathbf{T} = (T_1, T_2) = (\mathbb{V}_T, \mathbb{W}_T)$$

for $T_1 = \mathbb{V}_T(\boldsymbol{\lambda}, \boldsymbol{l_{x_i}})$, we differentiate the expression representing the set, in this case $\|\mathbf{r}_i\|$, where i=x,y,z. Let us denote that with $V_T(\boldsymbol{\lambda}, \boldsymbol{l_{x_i}})$. then it is easy to show that:

$$\frac{\partial T_1}{\partial \boldsymbol{\lambda}} = \frac{\partial V_T(\boldsymbol{\lambda}, \boldsymbol{l_x})}{\partial \boldsymbol{\lambda}} \neq 0$$

$$\frac{\partial T_1}{\partial \boldsymbol{l_x}} = \frac{\partial V_T(\boldsymbol{\lambda}, \boldsymbol{l_x})}{\partial \boldsymbol{l_x}} \neq 0$$

$$\frac{\partial T_1}{\partial \boldsymbol{\alpha}} = \frac{\partial T_2}{\partial \boldsymbol{\beta}} = 0$$

For $T_2 = \mathbb{W}_{T_i}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x})$, the expression representing the set is $\mathbf{w}_{T_i} = \mathbf{F}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}) \cdot \mathbf{u}$. It is easy to show that:

$$\nabla T_2 = \nabla(\mathbf{F}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}) \cdot \mathbf{u}) \neq 0$$

Thus, our optimization design variables would be $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}, \boldsymbol{l_x}$

B.2
Appendix: Method 1

Method 1 assumes that the effective thrust needed to counteract the drone's weight is distributed equally among all propellers. This simplification provides same results as the one of the paper when all tilt angles are zero and gives very good results when they are equal, but may introduce inaccuracies when tilt angles vary.

Total Thrust Calculation: The total thrust T_h required to counteract the weight is calculated using the following formula:

$$\boldsymbol{T_h} = \frac{mg}{\sum(\cos(\boldsymbol{\alpha}) \cdot \cos(\boldsymbol{\beta}))}$$

where m is the mass of the drone, g is the acceleration due to gravity, and α_i and β_i are the tilt angles of each propeller. The T_h is the combined effective thrust required to counteract the weight of the object. Each propeller contributes a portion of this based on it's orientation. The combined effective thrust T_h of each propeller must be distributed among the different orientations to result in the effective vertical thrust required to counteract the weight.

Vertical Thrust Component: The vertical thrust component for each propeller $T_{i,z}$ is determined by multiplying T_h with the cosine of the tilt angles:

$$T_{i,z} = T_h \cos(\alpha_i) \cos(\beta_i)$$

because the quantity of interest to us is the power consumption of rotor when at hover, we plug into the induced velocity of each rotor and power consumption the whole MRAV at hover ,the newly computed effective thrust:

$$u_{i_h} = \sqrt{\frac{T_h}{2\rho\pi r_p^2}} \quad \text{(m/s)}$$

and in the calculation of power consumption at hover for the whole MRAV:

$$P_{h,i} = \sum \left(\frac{u_{i,h} T_h}{\eta_P} \right)$$

B.3 Appendix: Minimum Guaranteed Control Force and Torque with unidirectional propellers

We consider u_i the unit vector representing the orientation of the i-th rotor and the convex force space of the unidirectional-thrust MRAV. Then similarly to Park et al. (2018) and inspired by Bosscher et al. (2006), we define the guaranteed minimum force generated for any orientation, from all rotors, for unidirectional MRAVs as:

$$F_{\min} = \min_{i,j} \sum_{k=1}^{\text{np}} \begin{cases} \frac{mg}{2} \cdot \frac{|(\mathbf{u}_i \times \mathbf{u}_j) \cdot \mathbf{z}|}{\|\mathbf{u}_i \times \mathbf{u}_j\|}, & \text{if } (\mathbf{u}_i \times \mathbf{u}_j) \cdot \mathbf{u}_k < 0 \\ \frac{c_f \cdot t_{\max}}{2} \cdot \frac{|(\mathbf{u}_i \times \mathbf{u}_j) \cdot \mathbf{u}_k|}{\|\mathbf{u}_i \times \mathbf{u}_j\|}, & \text{if } (\mathbf{u}_i \times \mathbf{u}_j) \cdot \mathbf{u}_k \geq 0 \end{cases}$$

with F_{min} being the minimum distance from the origin to the closest plane spanned by (u_i, u_j) of the attainable convex force space, along its normal vector $(u_i \times u_j)$. Because, we consider unidirectional thrust, if the plane is "under" the origin, the contribution of the gravitational force (mg) on the robot is considered. The maximum rotational speed squared, is considered as t_{max}. Similarly, we define the guaranteed minimum torque generated for any orientation, by all rotors as:

$$T_{\min} = \min_{i,j} \sum_{k=1}^{\text{np}} \frac{c_f \cdot t_{\max}}{2} \cdot \frac{|(\boldsymbol{\tau}_i \times \boldsymbol{\tau}_j) \cdot \boldsymbol{\tau}_k|}{\|\boldsymbol{\tau}_i \times \boldsymbol{\tau}_j\|}$$

C
Appendix: Software Implementation

C.1
Appendix: CasADi

CasADi is an open-source software framework designed for nonlinear optimization and optimal control, particularly suited for problems involving dynamic systems. At its core, CasADi provides a symbolic framework that enables the efficient formulation and solution of complex optimization problems. One of the key features of CasADi is its support for automatic differentiation, which allows for the exact computation of derivatives, a critical component in optimization algorithms. This capability is particularly valuable in the context of constrained multivariable optimization, where the accuracy and efficiency of derivative calculations directly impact the convergence and performance of the solver.

CasADi's integration with advanced and off-the-self solvers, such as the aforementioned IPOPT, further enhances its capability to tackle large-scale nonlinear optimization problems and one of the reasons that we chose to work with it. These solvers are optimized for performance and are adept at handling complex constraints and large variable sets, such as in our case. Additionally, CasADi is designed to exploit sparsity in matrices—a common characteristic of large-scale optimization problems—thereby improving both computational speed and memory efficiency.

Appendix: CasADi for solving the Design Problem

Using IPOPT options within CasADi allows for fine-tuning of the optimization process, enhancing both the accuracy and efficiency of solving nonlinear problems.

The "print level" option in IPOPT is crucial for debugging and monitoring the optimization process. By setting print level = 5, for example, users can obtain detailed information about each iteration of the solver, including the objective function value, the norm of the primal and dual infeasibilities, and other relevant statistics. This verbose output is particularly helpful for fine-tuning the solver parameters or diagnosing convergence issues. Understanding the solver's behavior at each step can guide in making informed adjustments to other solver options, to improve performance and achieve the desired results.

Tolerance parameters might play the most significant role in determining the accuracy and convergence criteria of the optimization process. For example, "tol" sets the overall tolerance for the optimality conditions, ensuring that the solution meets a specified accuracy. The "$dual\_inf\_tol$" controls the acceptable level of dual infeasibility, which is essential for maintaining the feasibility of the Lagrange multipliers associated with the constraints. Similarly, the "$constr\_viol\_tol$" defines the tolerance for constraint violations, ensuring that the final solution adheres to the specified constraints within an acceptable margin.

Additionally, the "$acceptable\_constr\_viol\_tol$" option allows for a more relaxed constraint violation tolerance during intermediate iterations, which can be particularly useful when dealing with difficult problems where strict feasibility is challenging to maintain throughout the optimization process. By providing a slightly higher tolerance during the initial stages, the solver can focus on finding a feasible region more efficiently and then gradually tighten the constraints to achieve the final desired accuracy.

In summary, the choice of CasADi for solving our Design Problem while developing a generalized, systematic, design procedure from task requirements to final MRAV configuration, over other methods, was driven by its superior efficiency, advanced solver capabilities, flexibility in problem

formulation, and the ability to effectively handle the complexities inherent in constrained multivariable optimization problems. These factors collectively make CasADi the optimal tool for our needs, ensuring both high performance and reliable results.

C.2 Appendix: Optimal Design

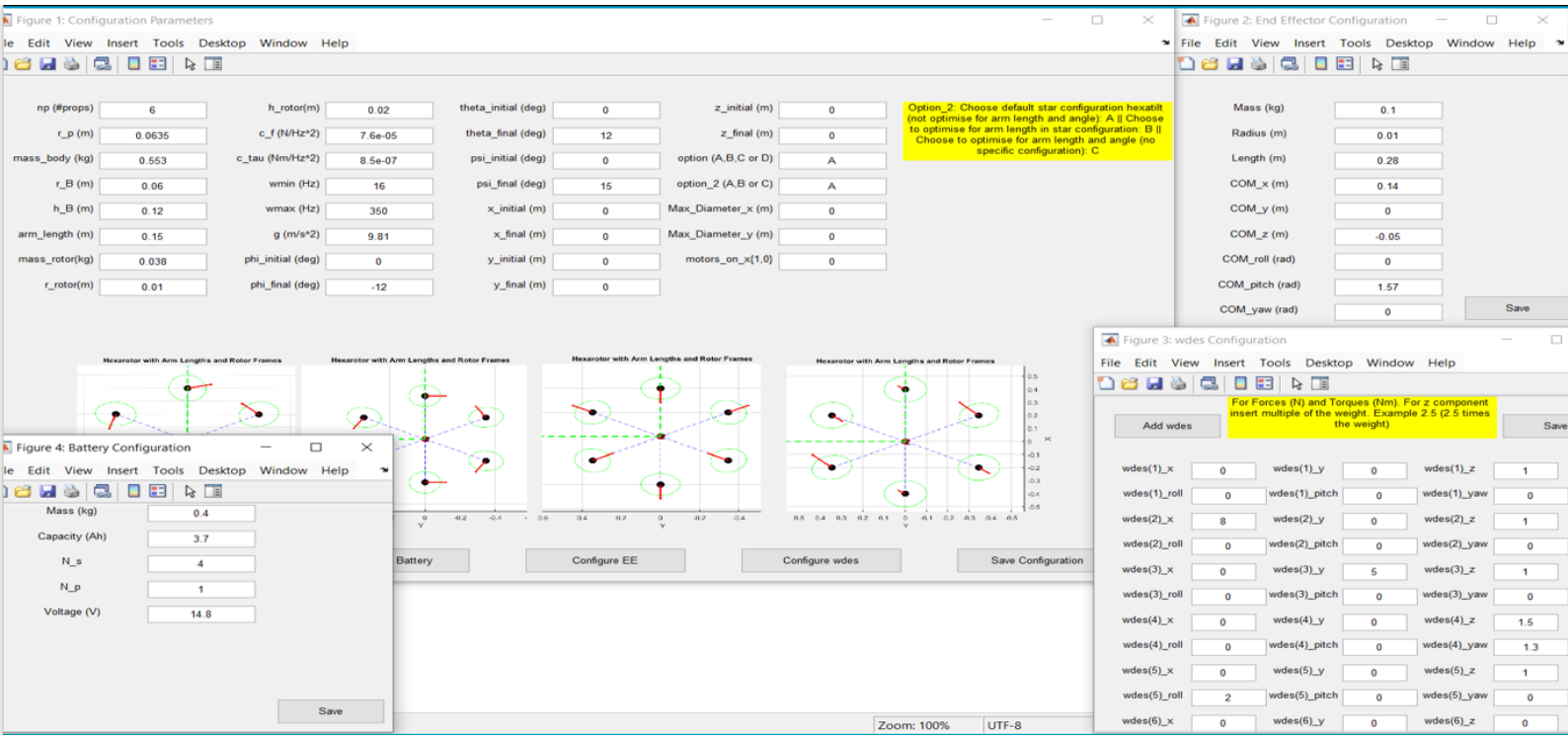


Figure C.1: Configuration GUI

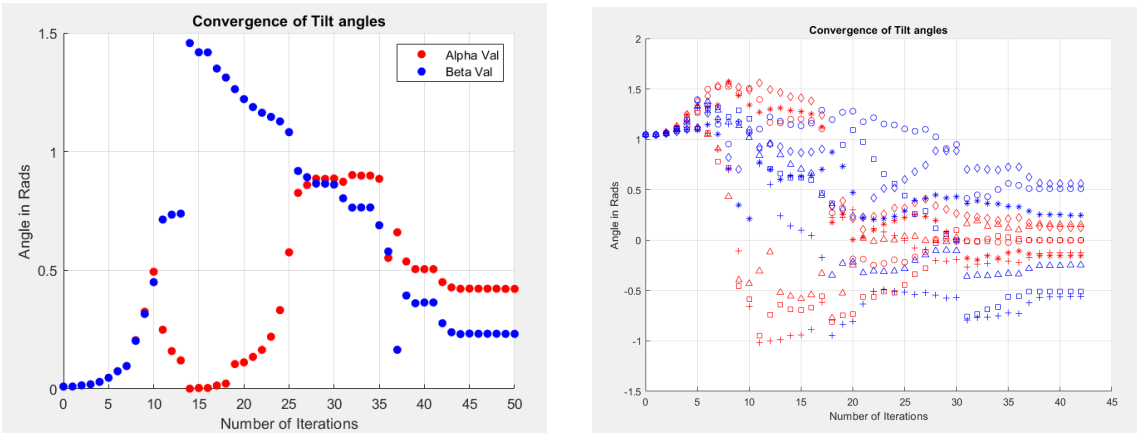


Figure C.2: Example: Convergence of solution for Configuration option for tilt angles A (left) and D (right). Blue is Beta value, red is Alpha. (Configuration Option for position of motors A)

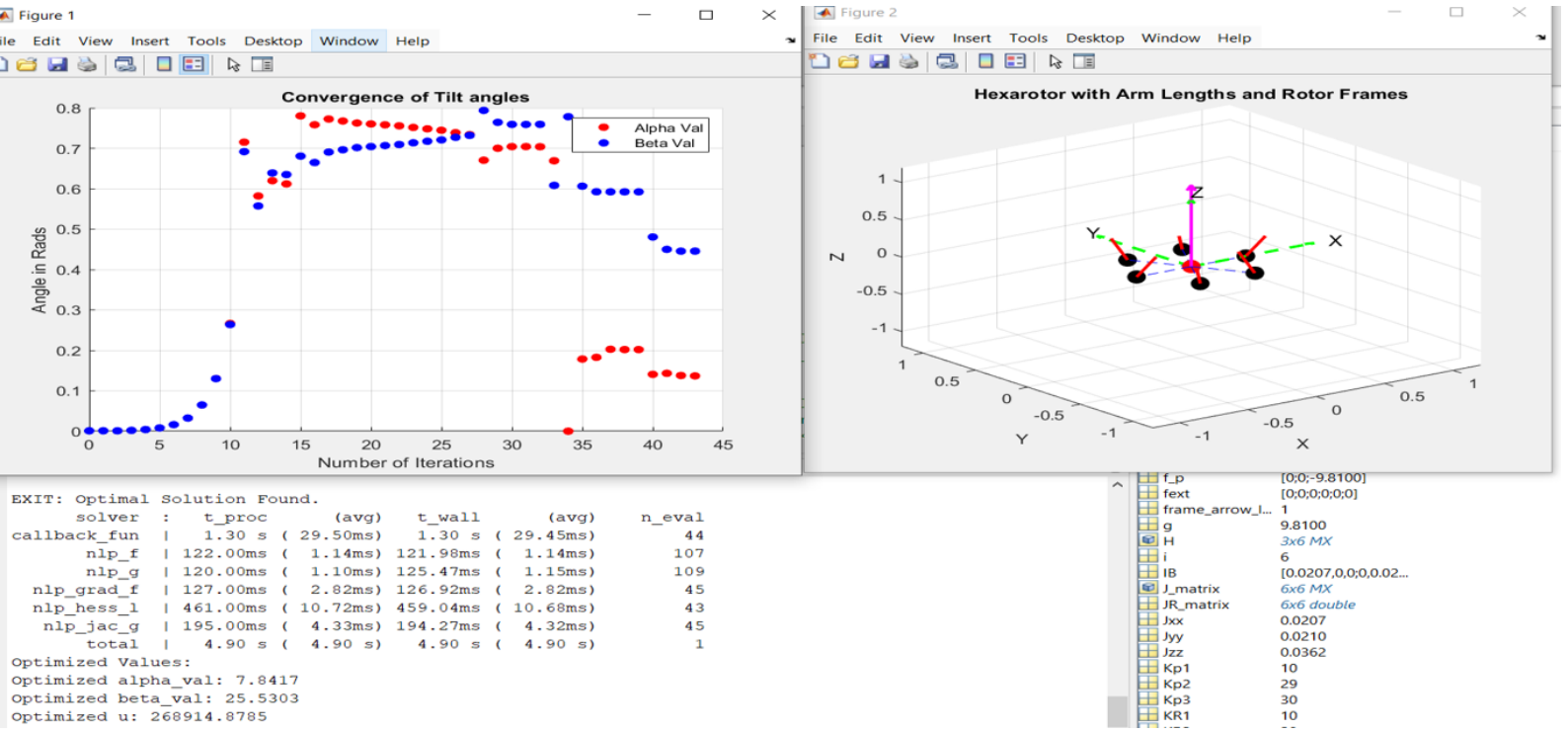


Figure C.3: Example: Output of Optimal Design script for a simple case

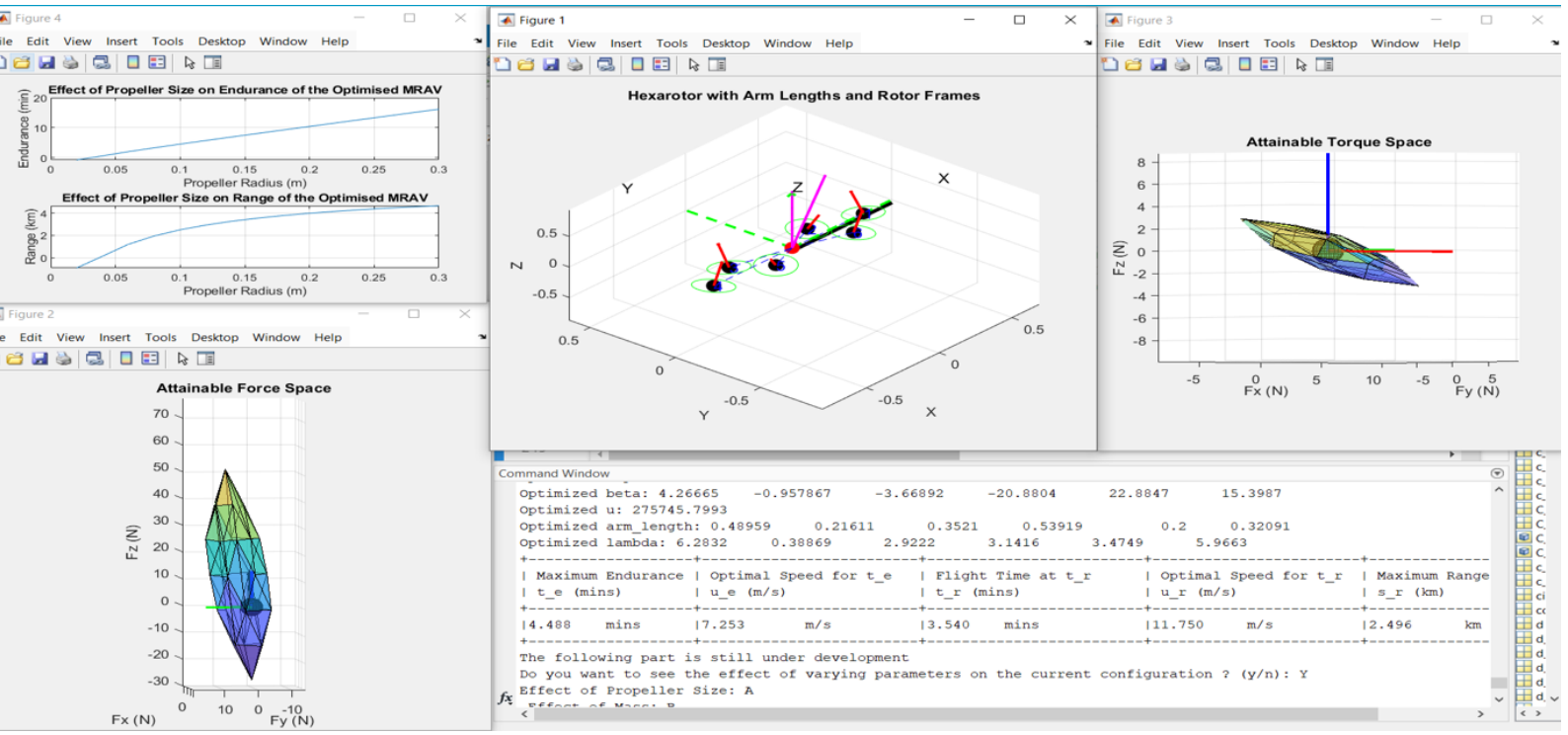


Figure C.4: Example: Output of the Optimal Design for a more complex case

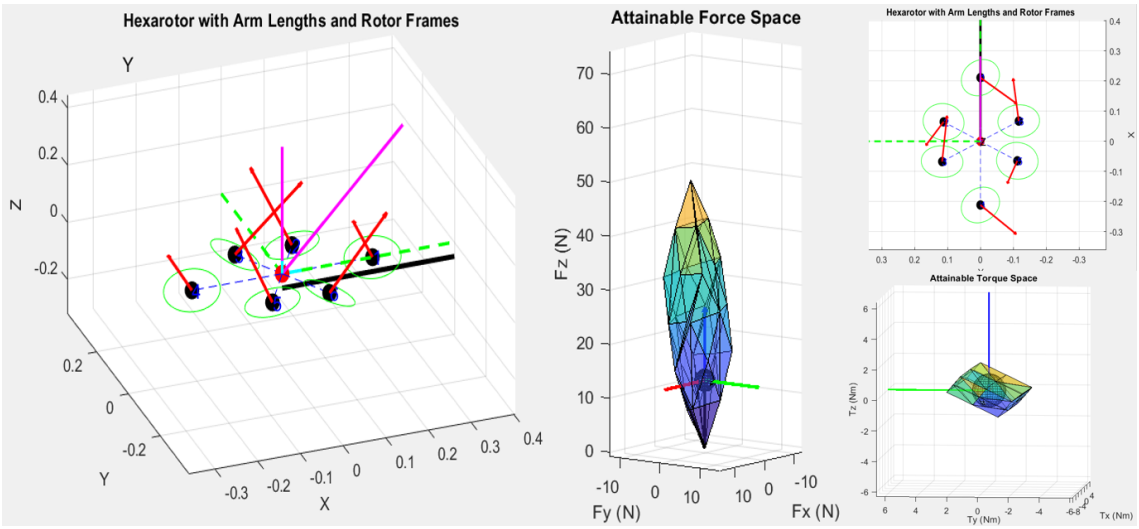


Figure C.5: Configuration Options Set (D-B), $\mathbf{w_{des}} = [601000]^T$, Volume Constraint y axis diameter$\leq 0.4m$ and x axis diameter$\leq 0.5m$. Axes of hexarotor plots are in meters.

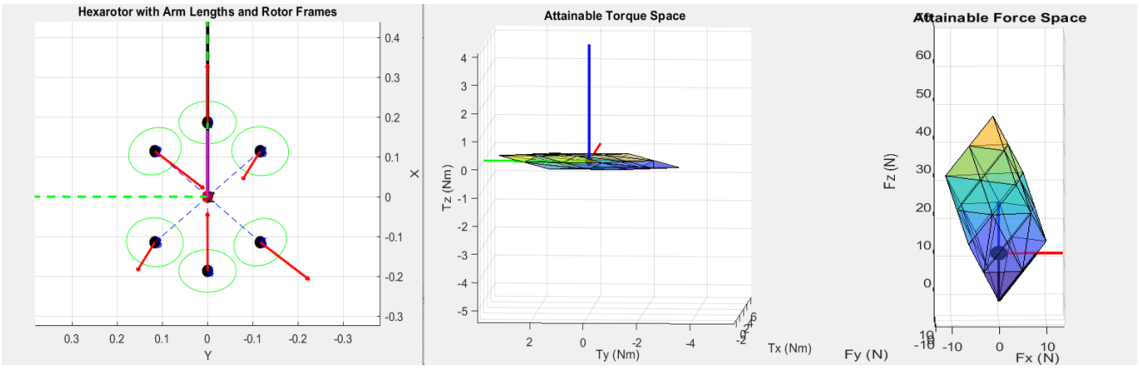


Figure C.6: Configuration Options Set (D-C), $\mathbf{w_{des}} = [601000]^T$, Volume Constraint y axis diameter$\leq 0.4m$ and x axis diameter$\leq 0.5m$ and constraint motors 1 and 6 to x axis. Notice the elongated torque space and their small omnidirectional capabilities of the design. Axes of hexarotor plots are in meters.

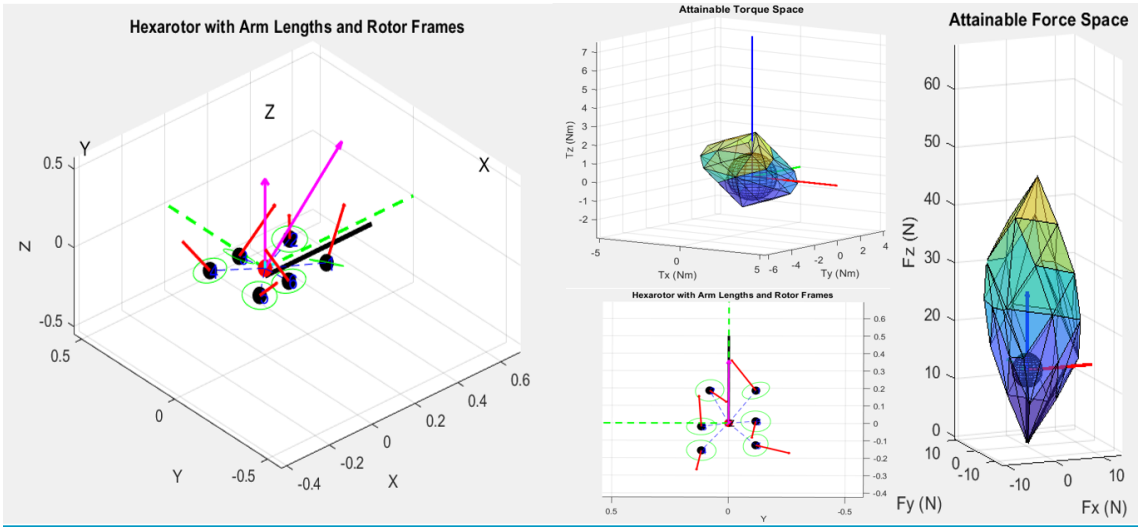


Figure C.7: Configuration Options Set (D-C), $\mathbf{w_{des}} = [701000]^T$, Volume Constraint y axis diameter$\leq 0.4m$ and x axis diameter$\leq 0.5m$. Axes of hexarotor plots are in meters.

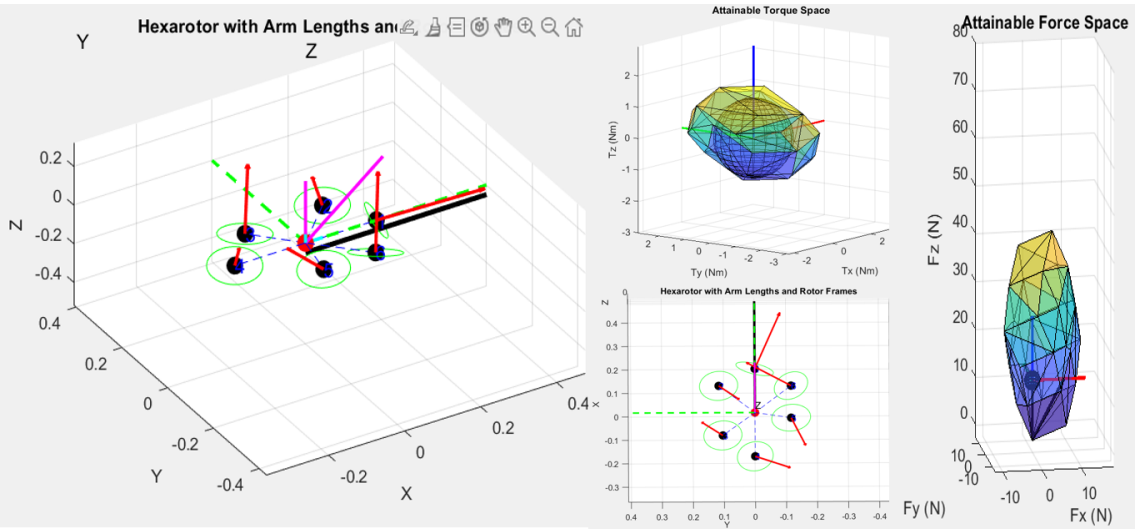


Figure C.8: Configuration Options Set (D-C), $\mathbf{w_{des}} = [901000]^T$, Volume Constraint y axis diameter≤ 0.4 and x axis diameter≤ 0.5. Notice how we redesign the force space to achieve the desired wrench, by elongating it in the x axis. Of course we can add more desired wrenches for example $\mathbf{w_{des}} = [-301000]^T, [031000]^T, [0-31000]^T$ and reshape the force space with larger omnidirectional capabilities of 3N (sphere of radious 3). It is worth noticing also the fact that the motor 1, for big forces on x is has a big beta angle. Axes of hexarotor plots are in meters.

C.3 Appendix: Simulation Script

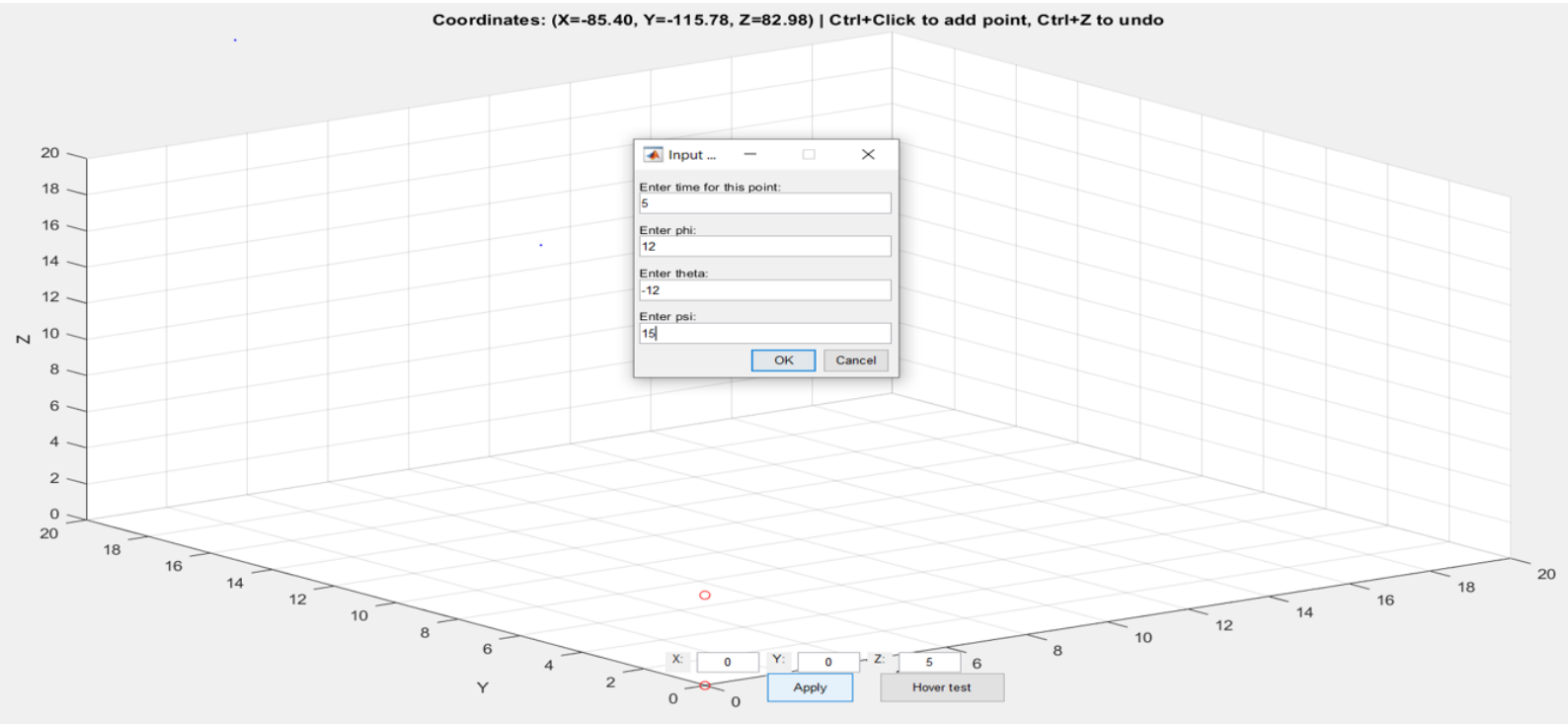


Figure C.9: GUI with manual coordinate input, hover test button etc

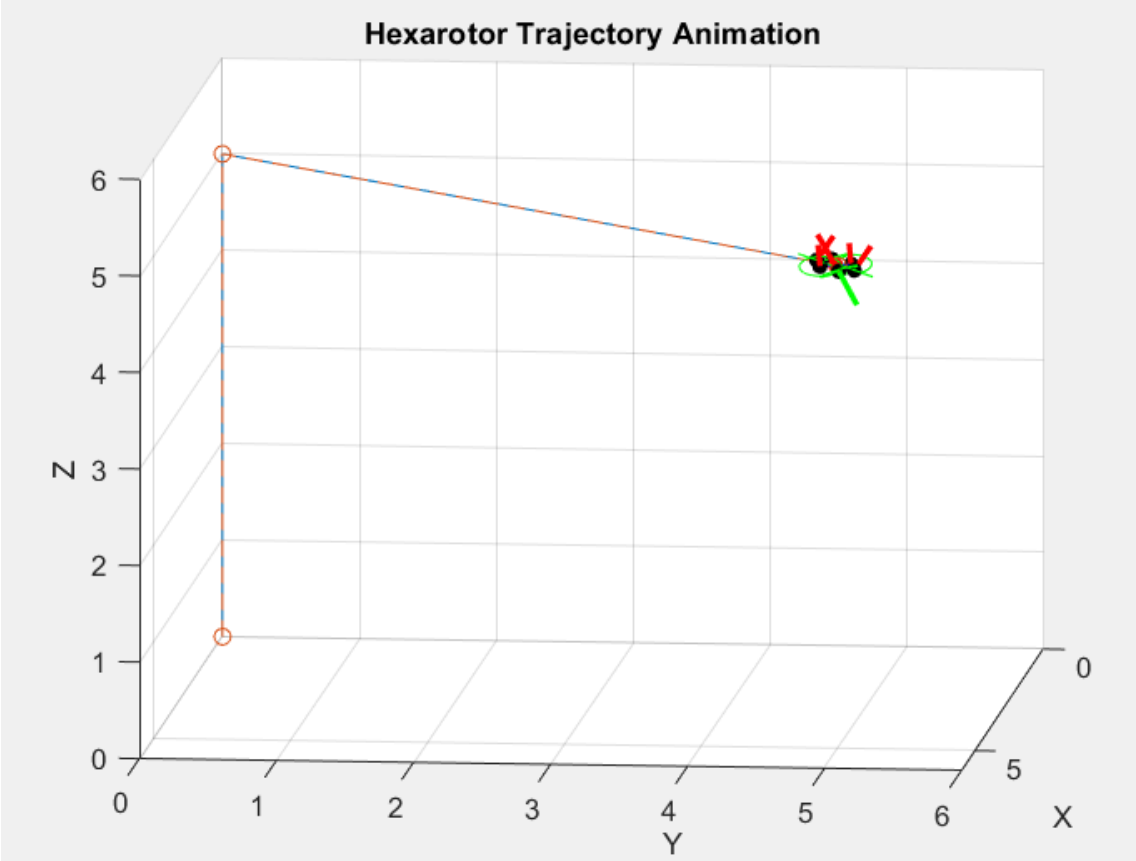


Figure C.10: Animation screenshot with axes units in meters.

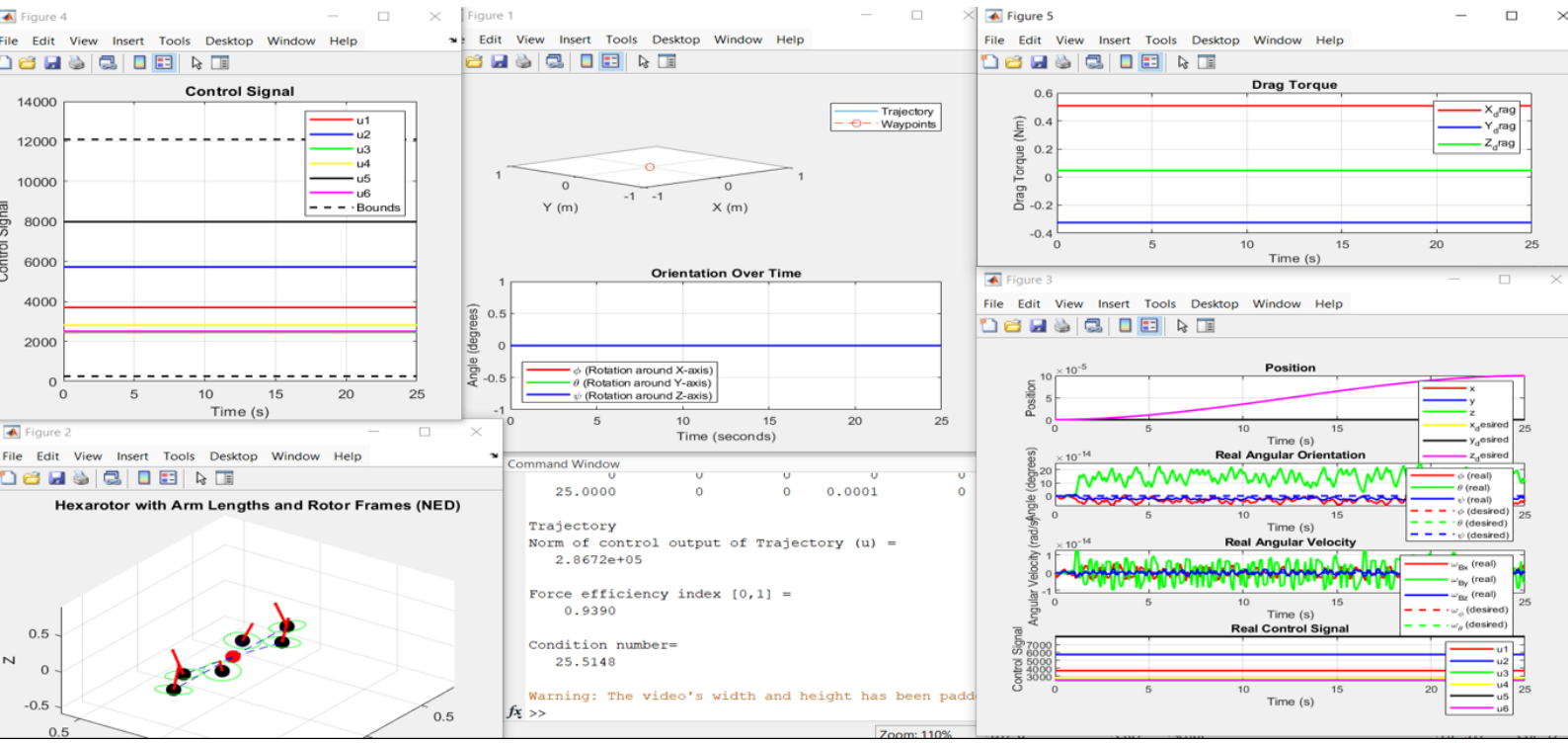


Figure C.11: Example: Hover simulation of an optimised design

C.4 Appendix: Example Simulation

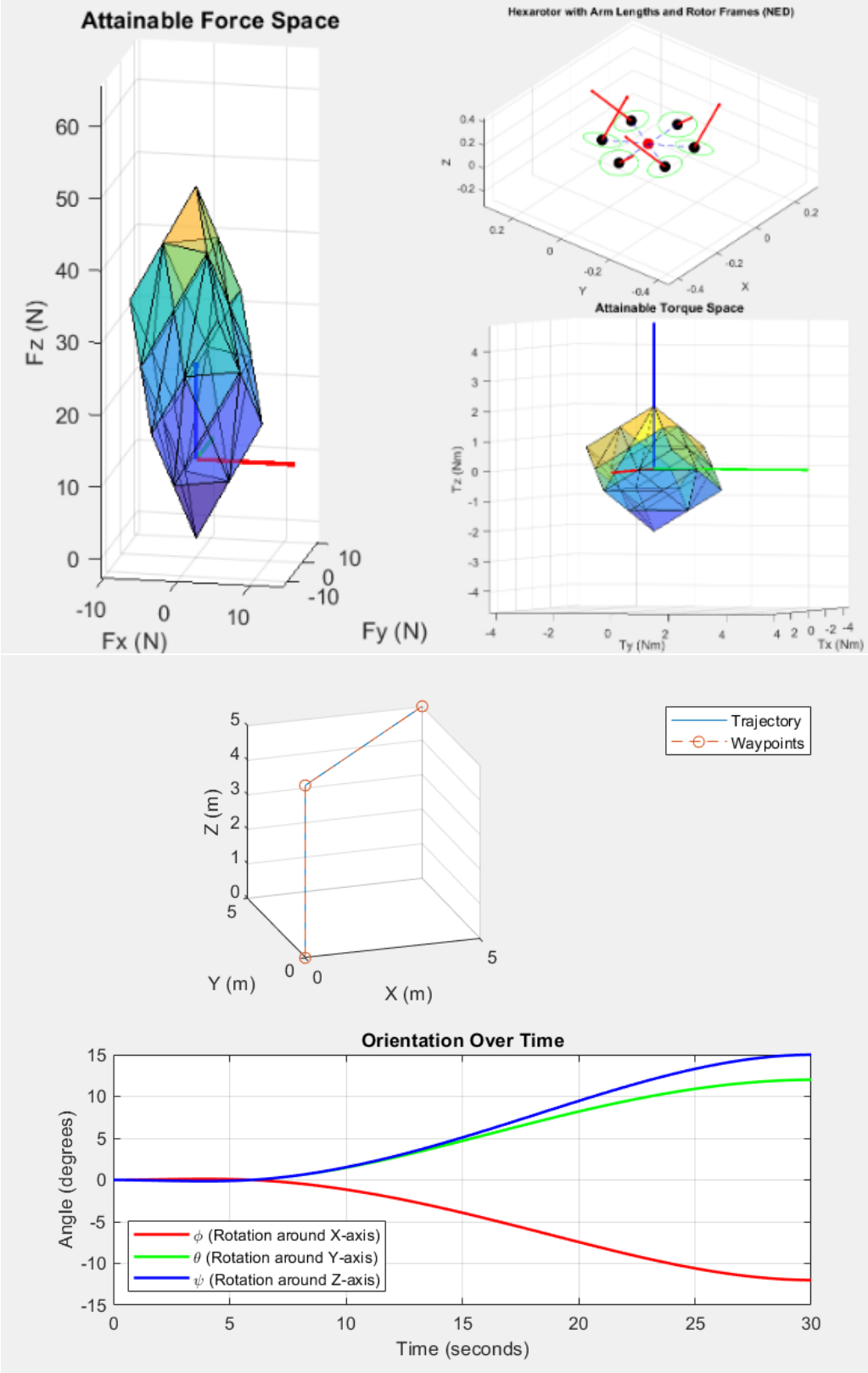


Figure C.12: MRAV (top) and trajectory to be followed by the MRAV (bottom). Axes of MRAV plots are in meters.

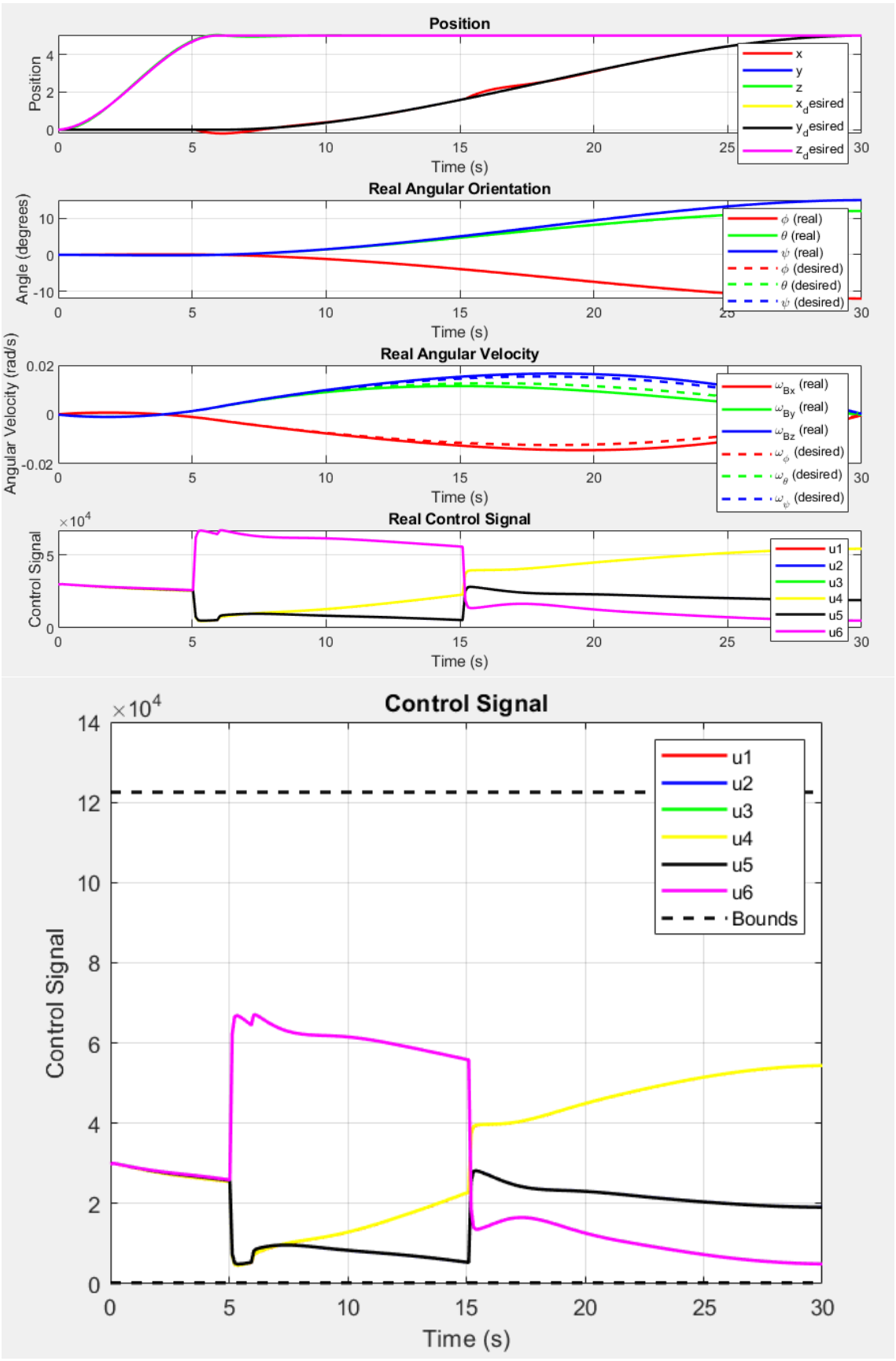


Figure C.13: Trajectory tracking with external disturbance of 4 N in x axis (top) and Control Signal (Hz^2) with motor limits (dashed lines) (bottom)

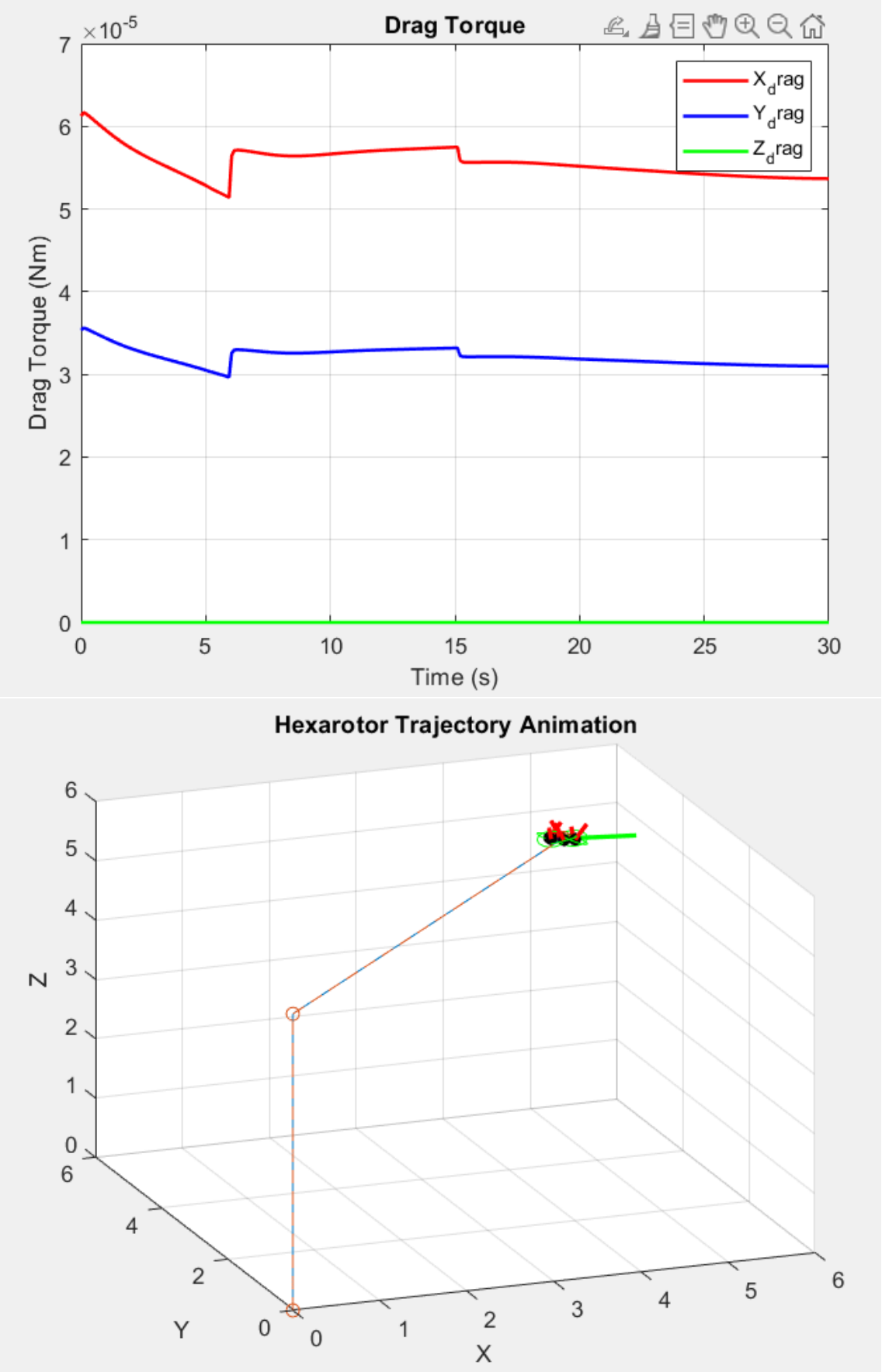


Figure C.14: Drag Torque experienced (top) and Trajectory Tracking Animation (bottom). Axes are in meters.

C.5 Appendix: Baseline Comparison

- Tilt-hex: External Disturbance of magnitude of 6N on the E.E. (30 deg rotation w.r.t. x-axis)

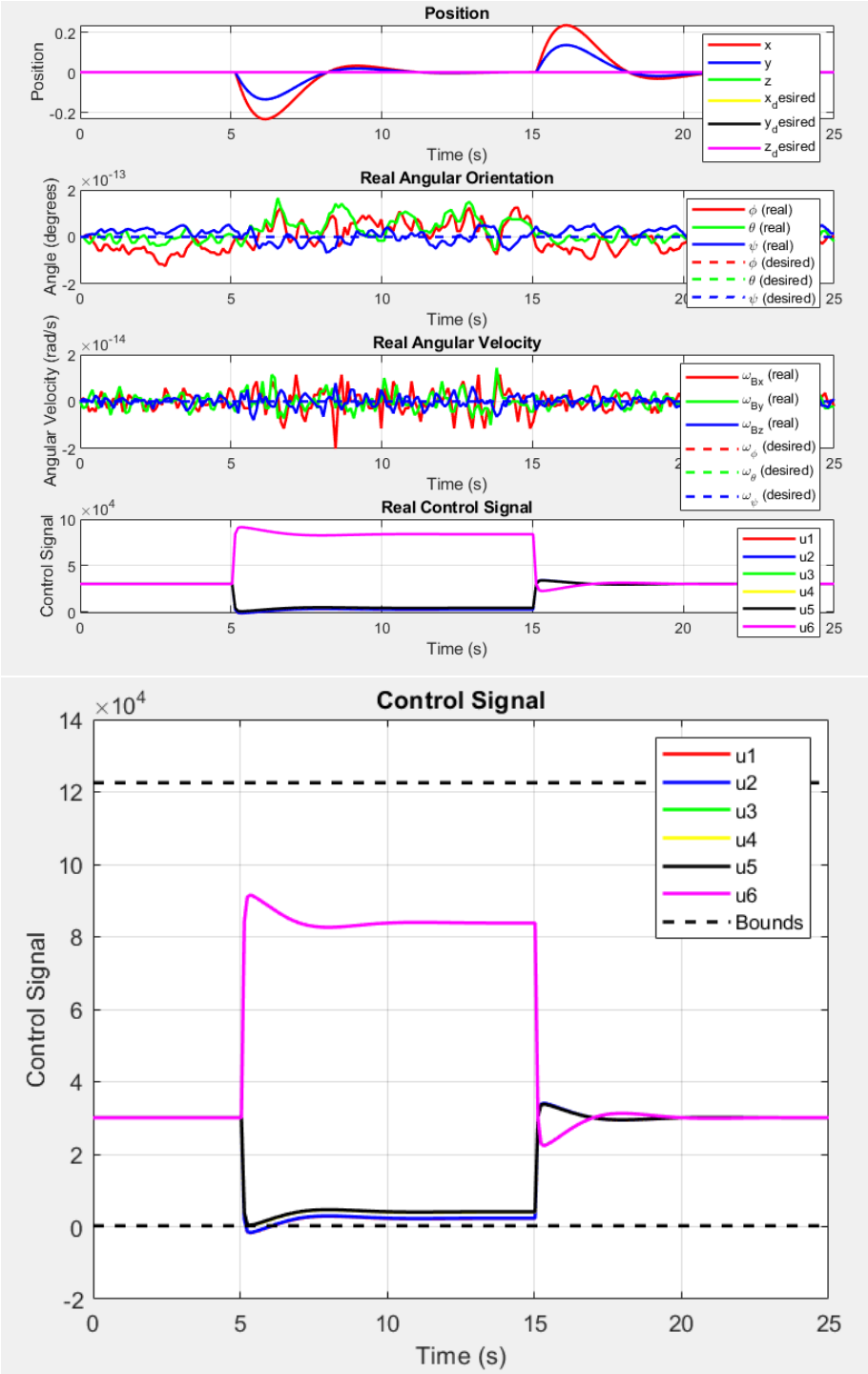


Figure C.15: Constant External Disturbance of magnitude of 6N applied parallel and on the tip of the E.E. that is rotated by 30 deg. Time of application of force between 5-15 s . Down is the control signal Hz^2-time(s) plot and up are the tracked trajectory (position in m). With black dashed lines are the motor limits.

- Tilt-hex: External Disturbance of magnitude of 6N on the E.E. (parallel to the x-axis).

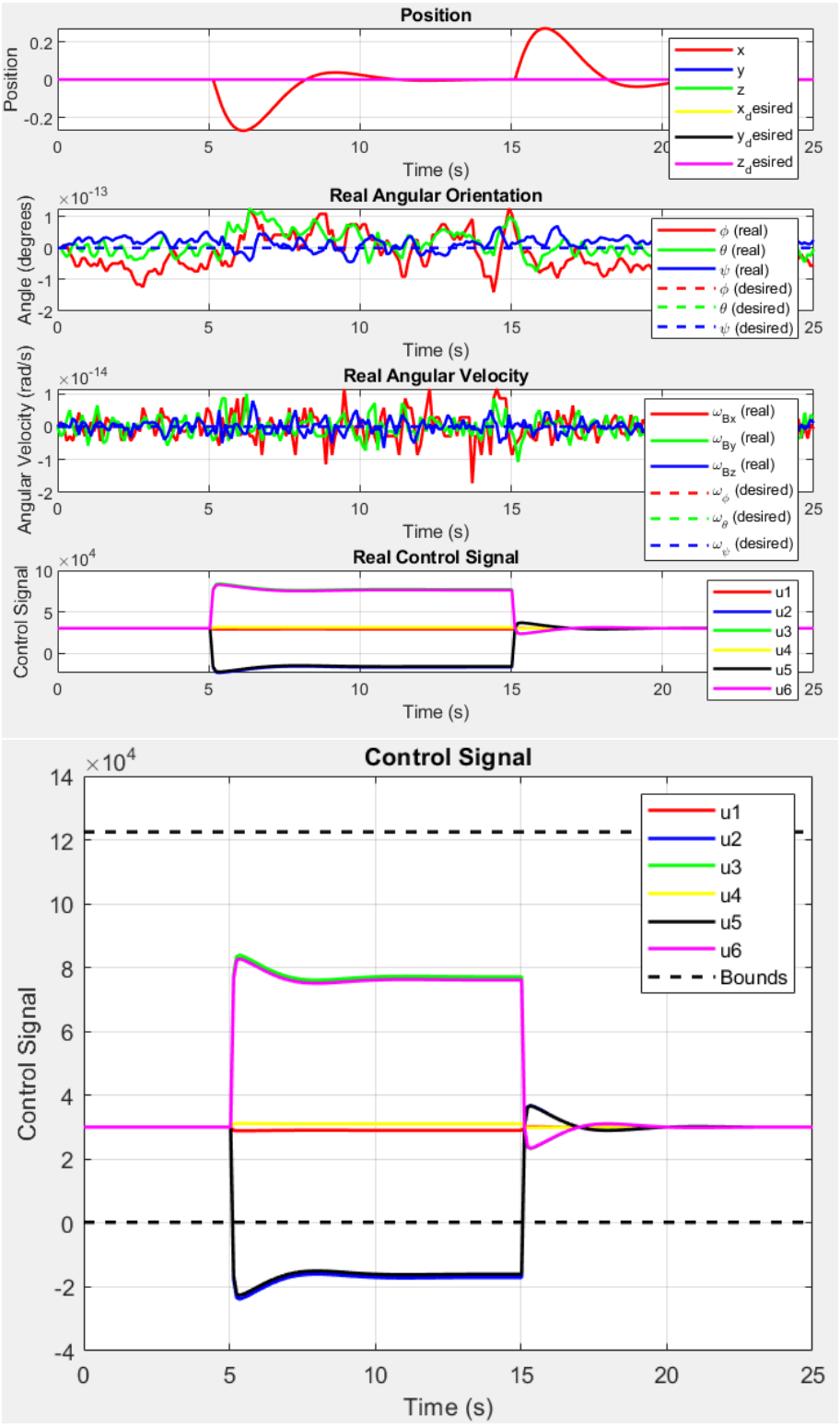


Figure C.16: Constant External Disturbance of magnitude of 6N applied parallel and on the tip of the E.E. (parallel to the x-axis of the Body Frame). Time of application of force between 5-15 s . Down is the control signal Hz^2-time(s) plot and up are the tracked trajectory (position in m). With black dashed lines are the motor limits.

D
Appendix: Simulations and Results

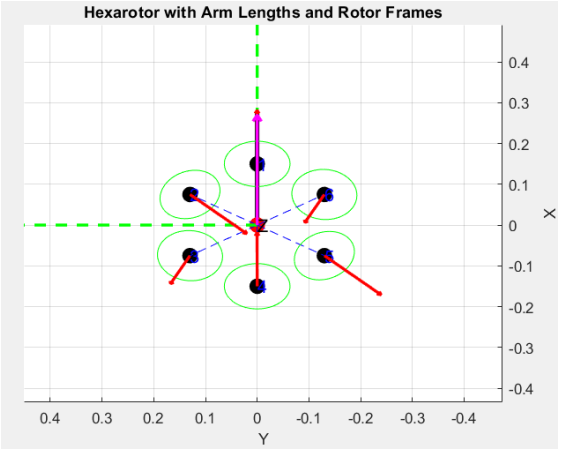
D.1
Appendix: Gazebo Simulation

- Design (D-A)

```
Norm of control output of Trajectory (u) =
   2.2075e+06

Force efficiency index [0,1] =
    0.9083

Condition number=
   11.6305
```




```
Optimized alpha: -0.055047     -7.0976     -8.8735    0.055046      7.1025      8.8694
Optimized beta: 29.0414     -32.0103      14.2756     -29.0414      32.0098     -14.2764
Optimized u: 1723339.3609
Optimized arm_length: 0.15        0.15        0.15        0.15        0.15        0.15
Optimized lambda: 0      1.0472      2.0944      3.1416      4.1888       5.236
+------------------+-------------------------+-------------------------+-----------------------+-----------------+
| Maximum Endurance | Optimal Speed for t_e   | Flight Time at t_r      | Optimal Speed for t_r | Maximum Range   |
| t_e (mins)        | u_e (m/s)               | t_r (mins)              | u_r (m/s)             | s_r (km)        |
+------------------+-------------------------+-------------------------+-----------------------+-----------------+
|11.582    mins      |11.372       m/s          |9.552    mins            |20.401     m/s         |11.692      km
+------------------+-------------------------+-------------------------+-----------------------+-----------------+
```

Figure D.1: Metric of the design (top right), Tilt angles and Endurance (bottom), Top view (axis in m) of the design (left)

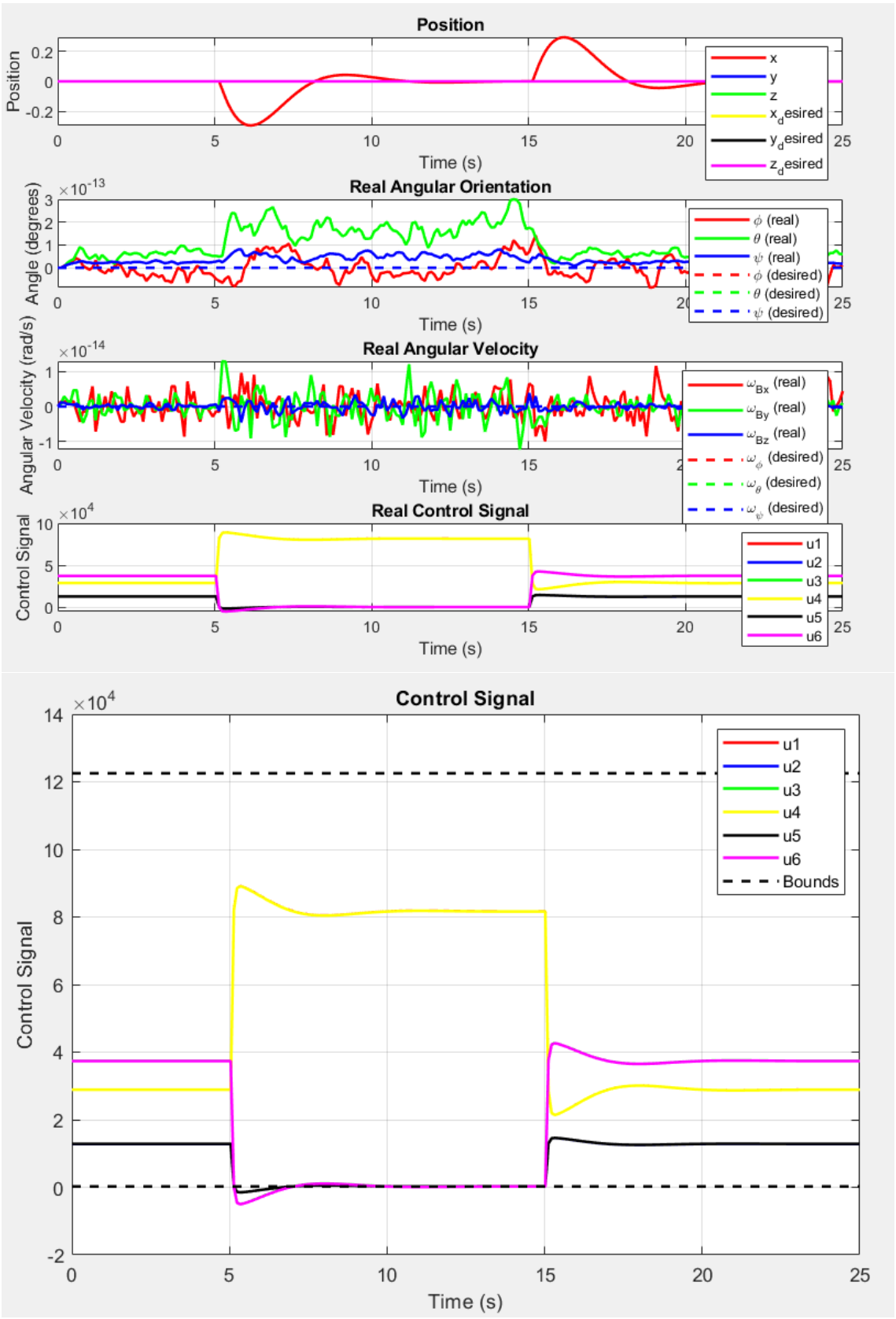


Figure D.2: External Disturbance of 6N in the x axis while hovering. Tracked trajectory (top) and control signal in Hz^2 (bottom).

- Design (D-C)

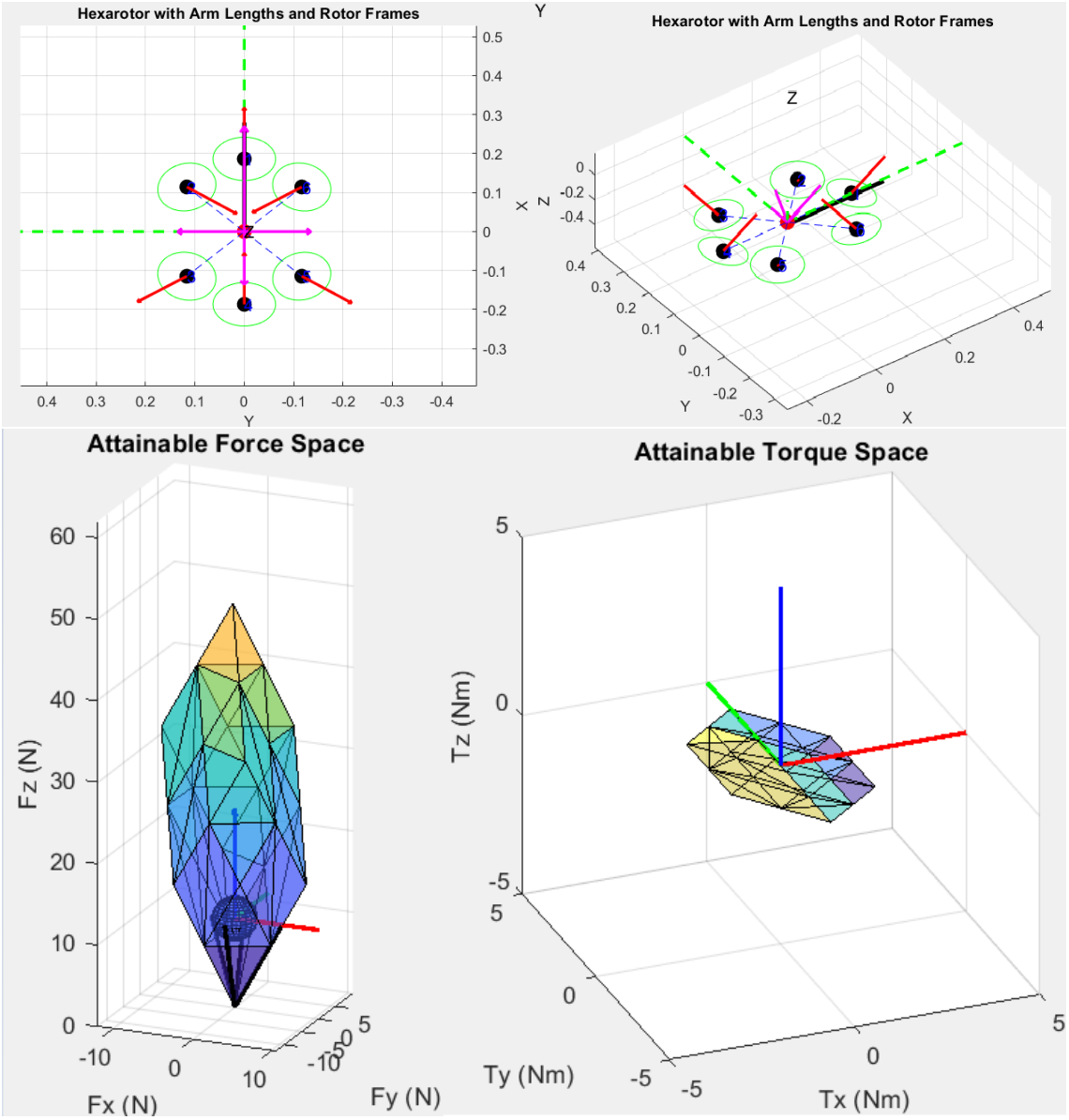


Figure D.3: Design (D-C) in m (top), Force (left) and Torque (right) Space (bottom)

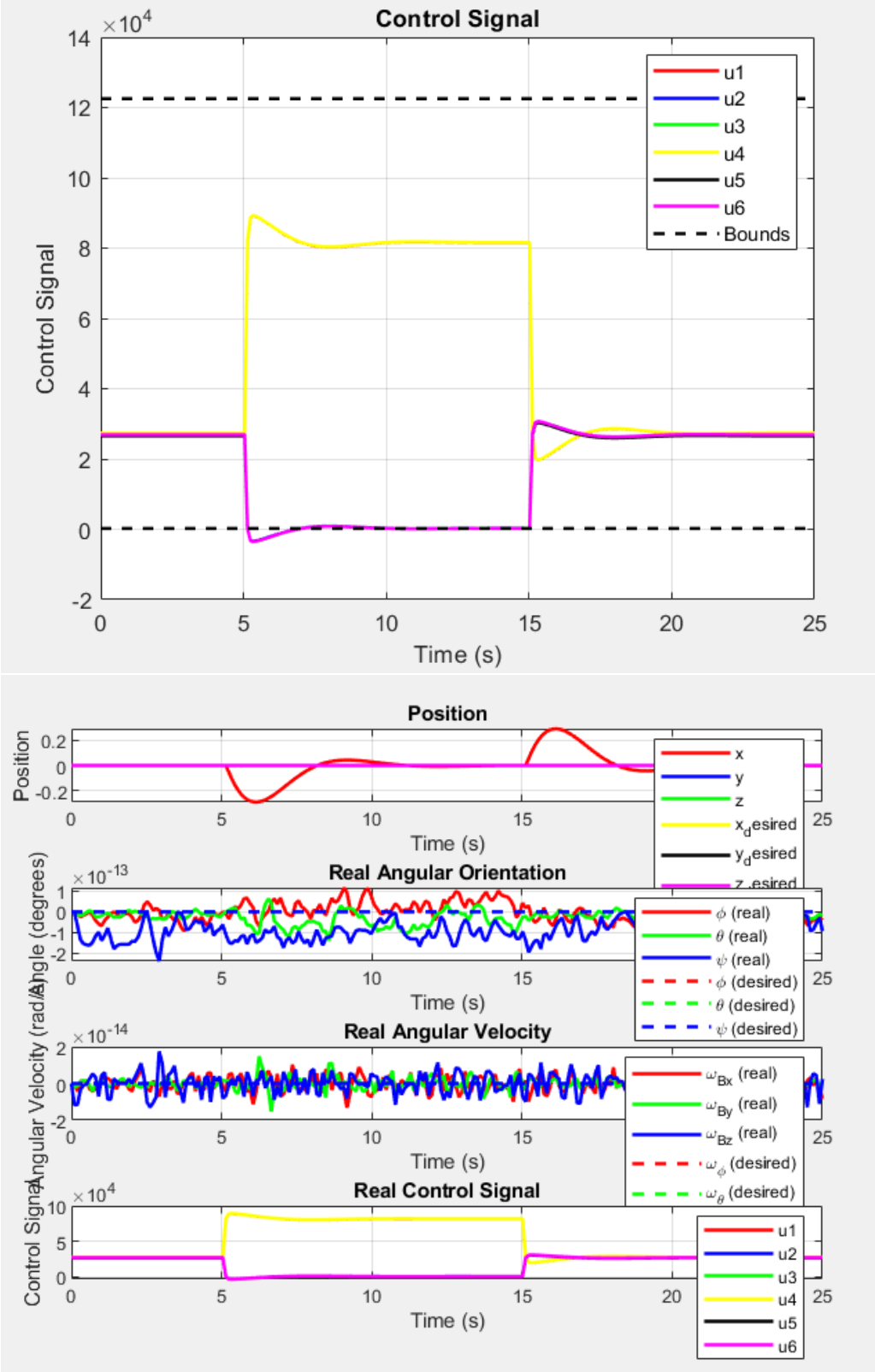


Figure D.4: Control signal plot in Hz^2, of the simulator script for external disturbance 6N in the x-axis (top) and tracked trajectory (bottom)

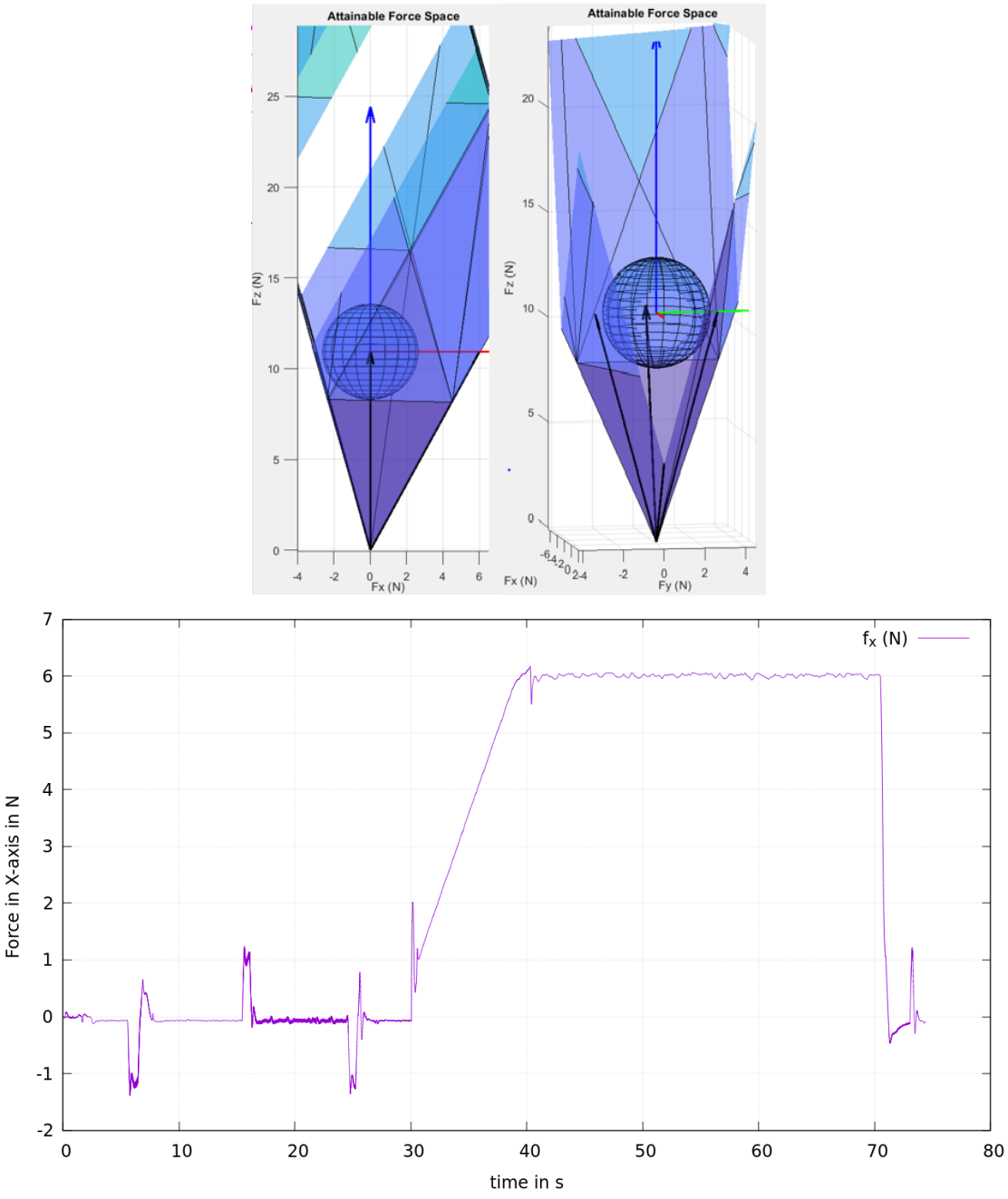


Figure D.5: Force space zoom (top). The desired forces (part of the desired wrenches) can be seen with black arrows shaping the boundaries of the force space. On the bottom the Gazebo simulation force plot from the logs of the uavatt can be seen, with the contact taking place from 24s-54s, with a 6N magnitude.

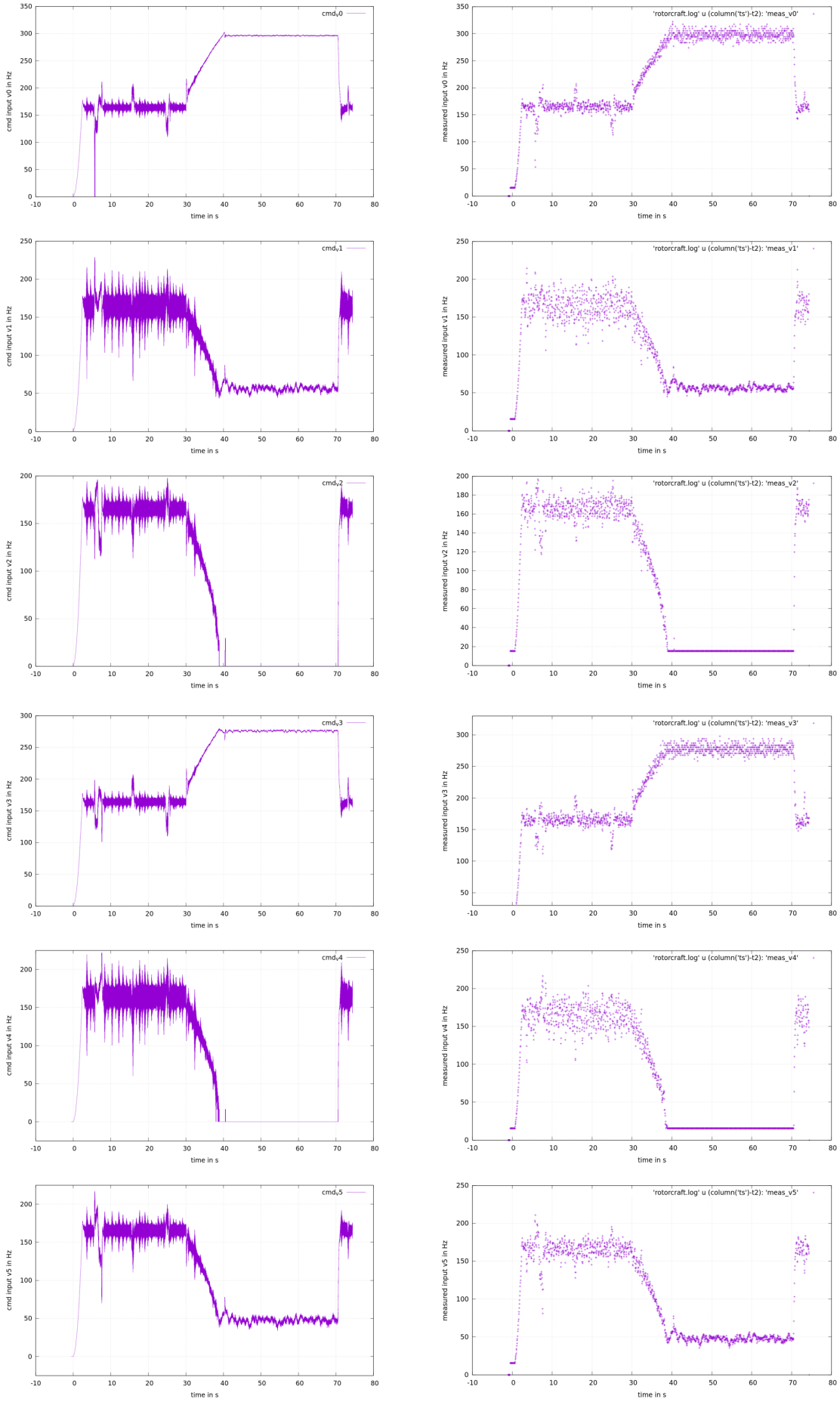


Figure D.6: Commanded input (left) and Measured velocities (right) (all in Hz)

D.2 Appendix: Coefficients Experiment

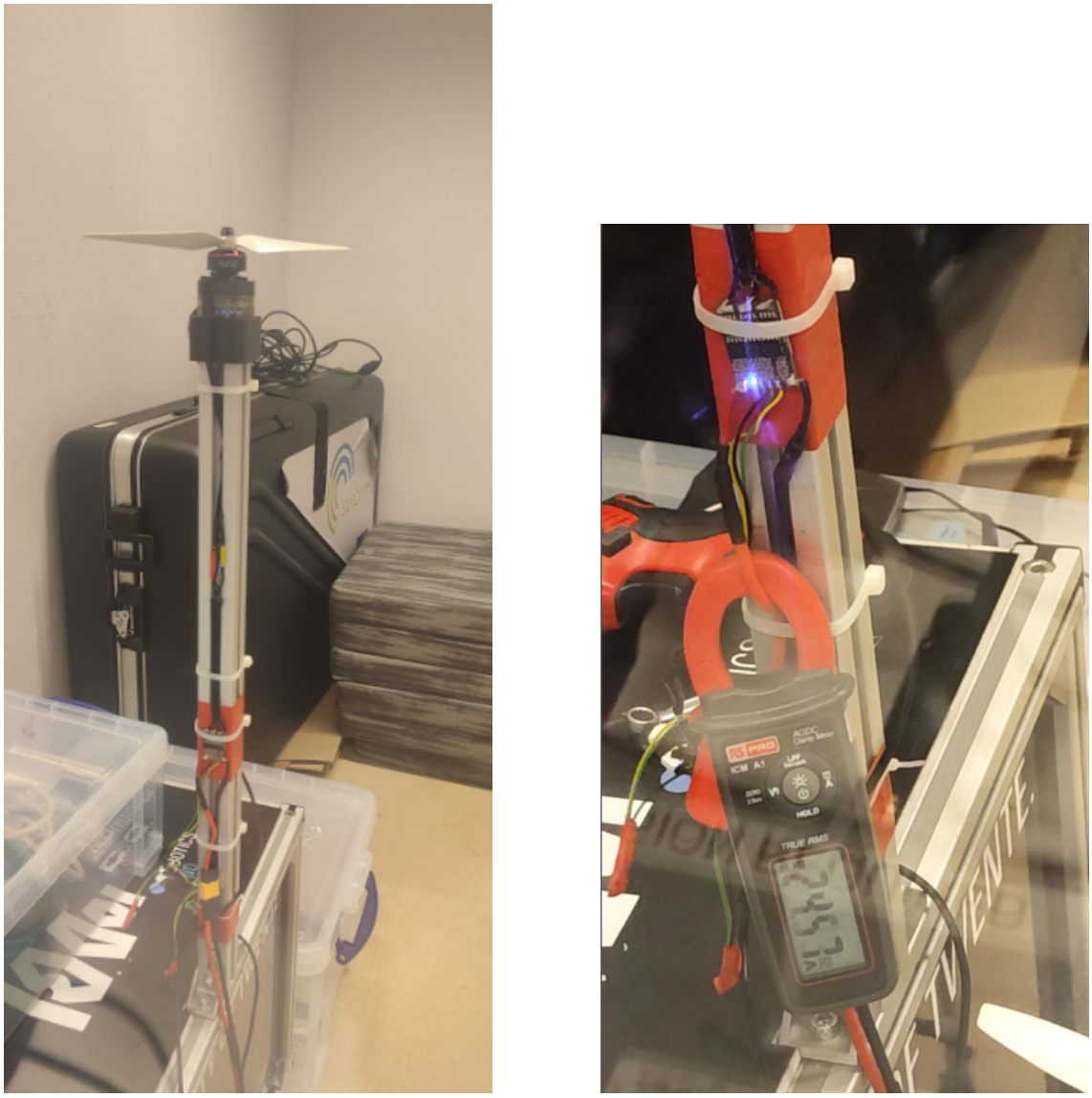


Figure D.7: Test Bench (left) and Clamp for DC current measurement (right)

Table D.1: Table of the Computed Thrust and Drag Coefficients for 6 sets of propellers

Propellers	**Thrust Coeff.** $\frac{N}{Hz^2}$	**Drag Coeff.** $\frac{Nm}{Hz^2}$	$\omega_{max} Hz$
DP 5x4x4V1S PC (Green)	5.86e-5	6.06e-7	>350
T5143S (Gray)	5.45e-5	6.2e-7	>360
Hulkie 5055-3 (Gemfan)	7.61e-5	8.53e-7	>360
Duct 4x4x6	6.4e-5	9.32e-7	>360
WinDancer 5043-3 (Gemfan)	6.4e-5	7.12e-7	>360
DP 4.8x3.4x4 (Grey)	6.15e-5	7.1e-7	>360

D.3 Appendix: AUTOASSESS usecase

D.3.1 Appendix: Component Weights

Table D.2: Components and their respective weights

Component	Weight (kg)
Flight Controller + Onboard computer + electronics	0.090
Motor-Propeller combination (per combination)	0.045
ESCs-Cables (4 in 1 and 2 singles)	0.080
Arm + tilt adaptor (per combination)	0.027
2 plates	0.030
2 Protection (each)	0.030
Battery (small 2.3 Amps)	0.216
Battery (big 3.7 Amps)	0.311
End Effector	0.110

D.3.2

Appendix: Proposed Designs for AUTOASSESS use case (Axes of MRAV plots in meters)

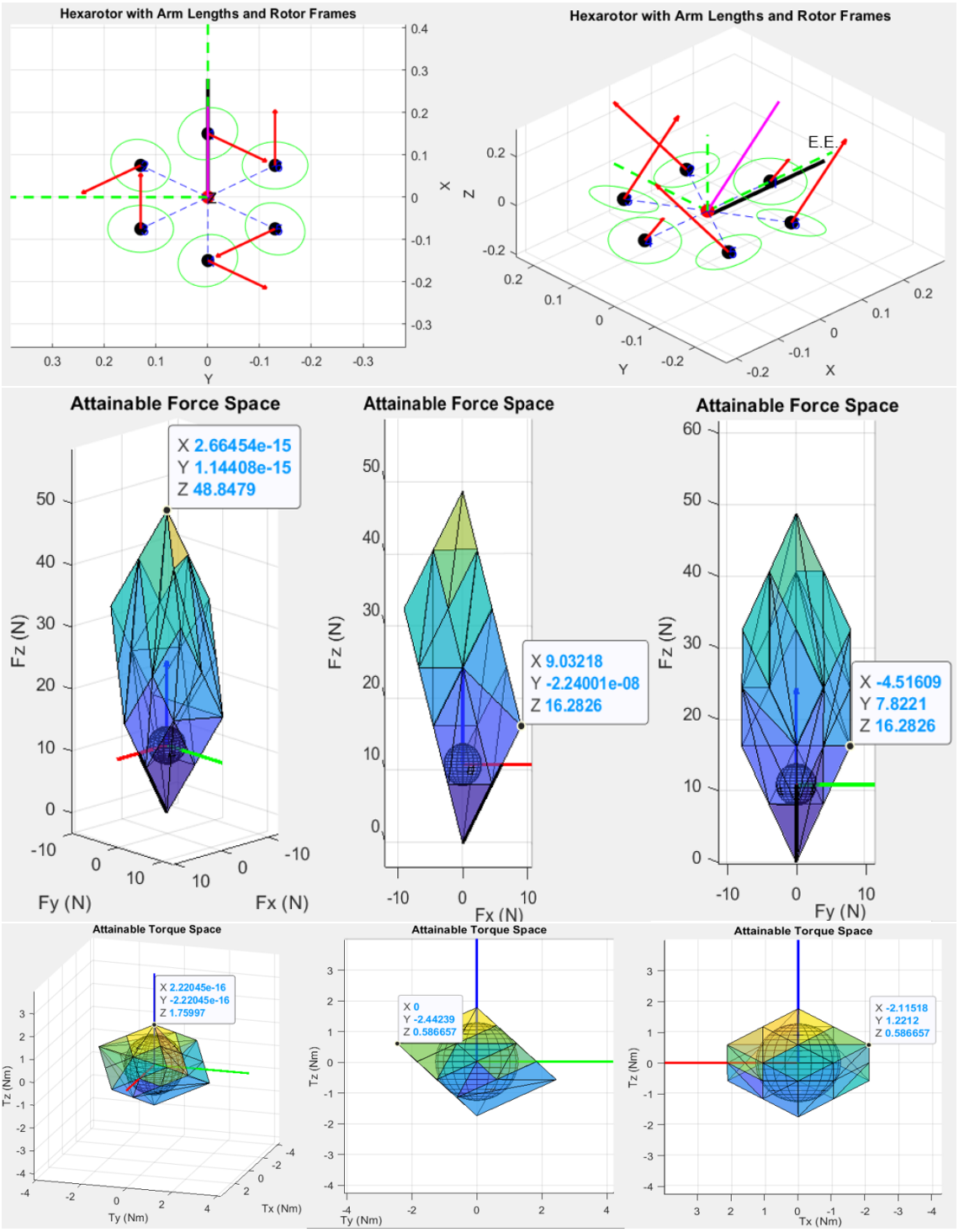


Figure D.8: Design 1

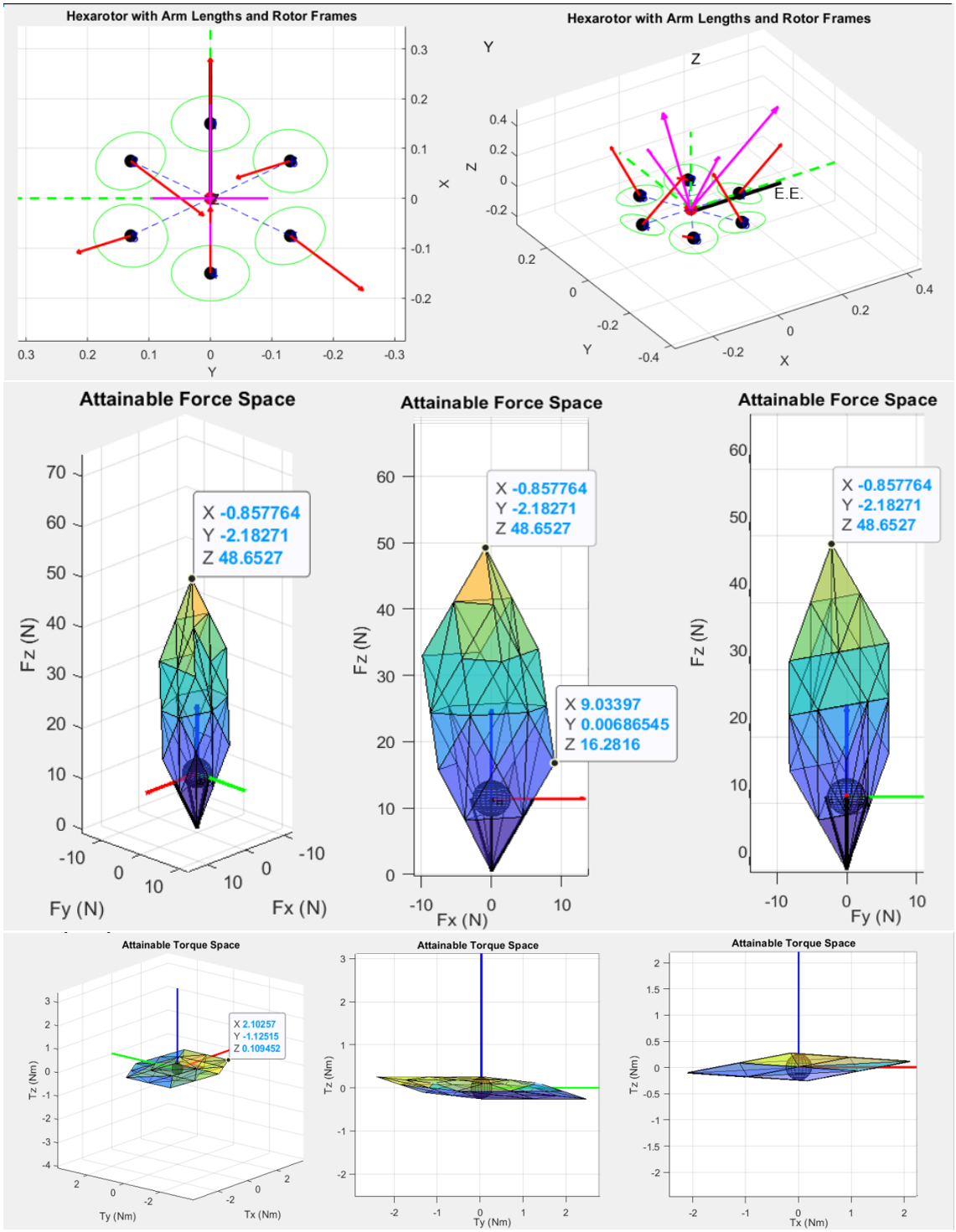


Figure D.9: Design 2

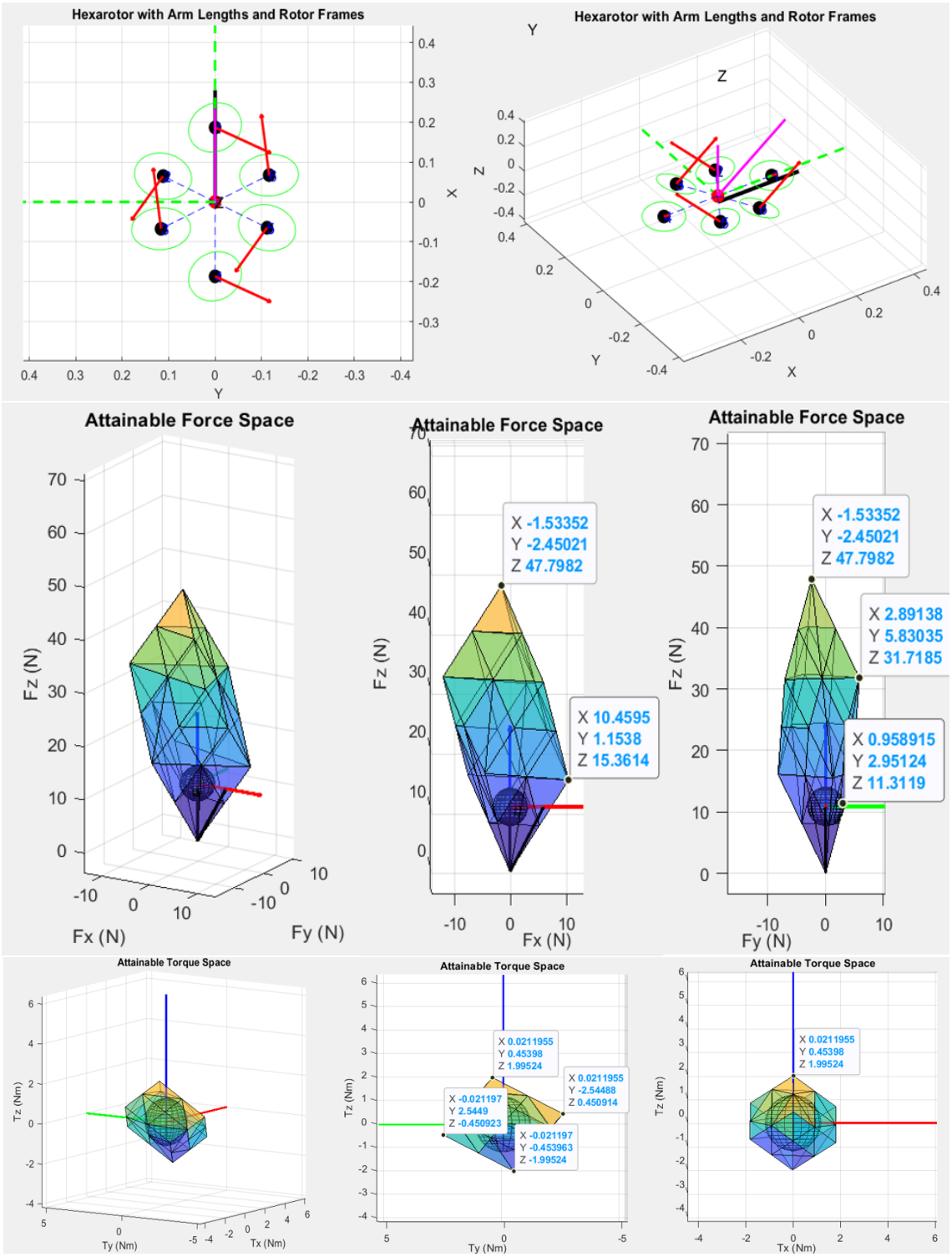


Figure D.10: Design 3

D.3.3

Appendix: Proposed Designs 6D trajectory tracking and External Disturbance Application

- Design 1: Reorienting while hovering

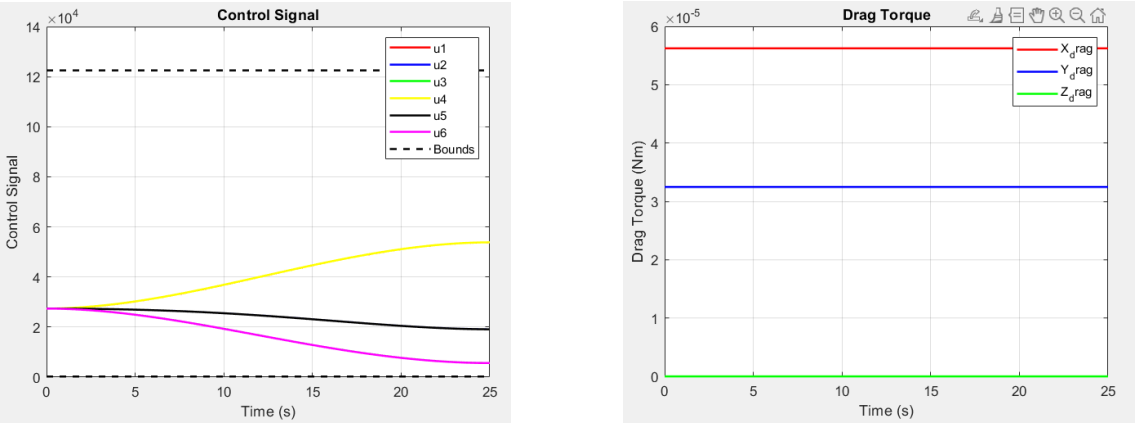


Figure D.11: Control Signal (Hz^2) (Left) and drag torque experienced (Right) over the trajectory

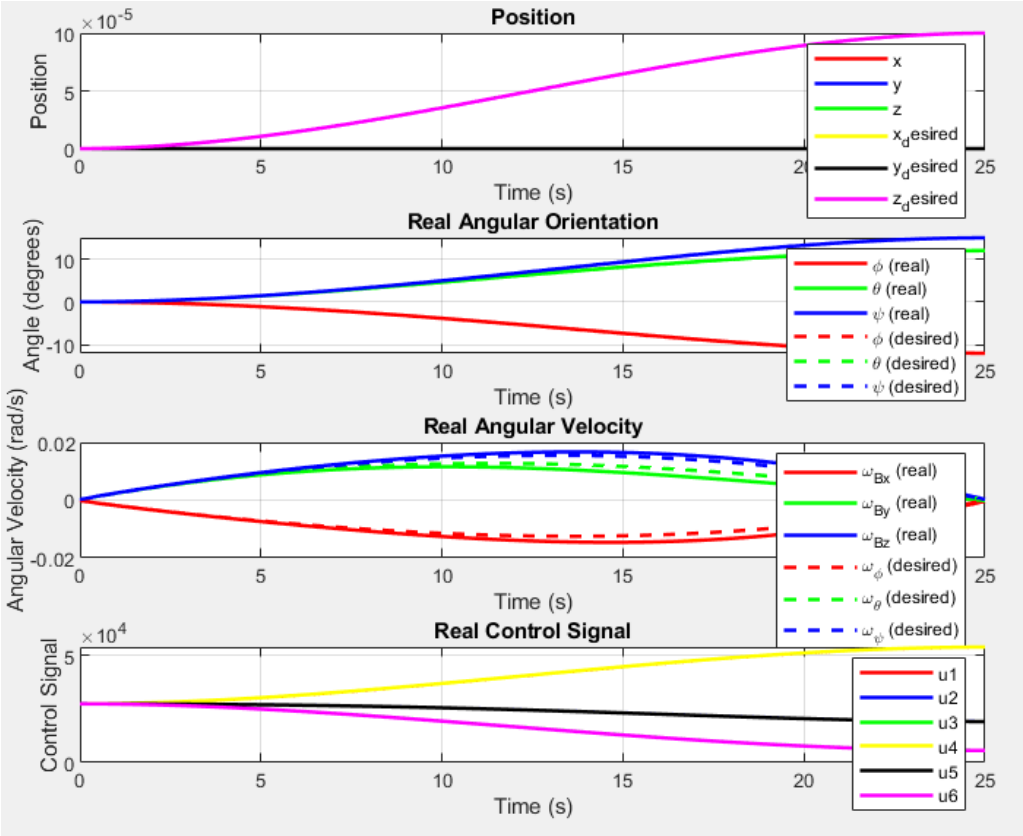


Figure D.12: Tracked position (m), orientation, angular velocity and Control Signal (Hz^2)

- Design 1: Hovering with External Disturbance (Max attainable)

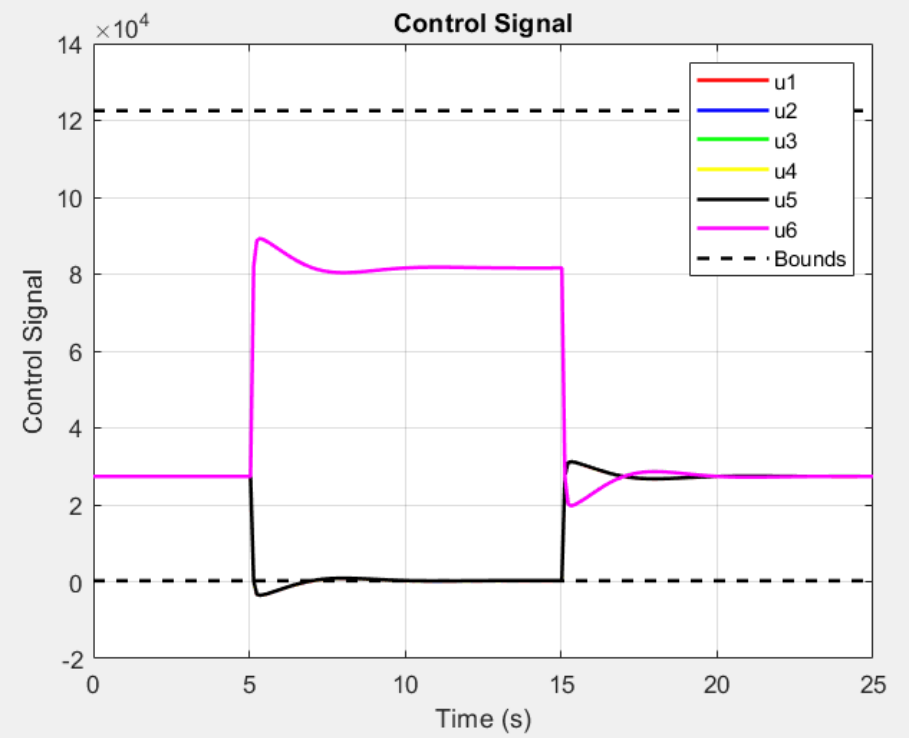


Figure D.13: Control Signal (Hz^2) over the trajectory

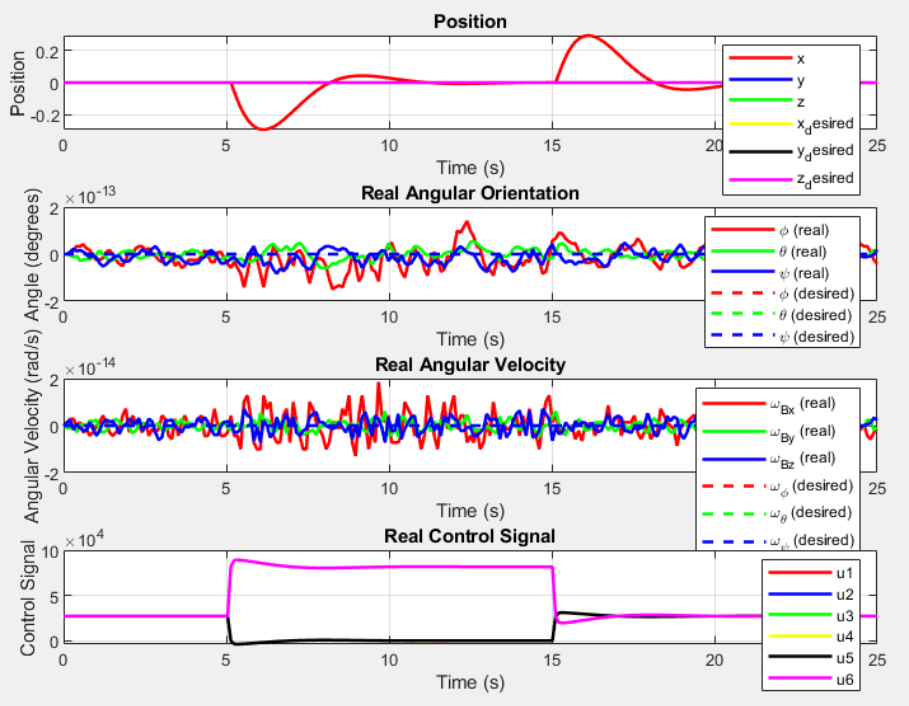


Figure D.14: Tracked position (m), orientation, angular velocity and Control Signal (Hz^2)

- Design 2: Reorienting while hovering

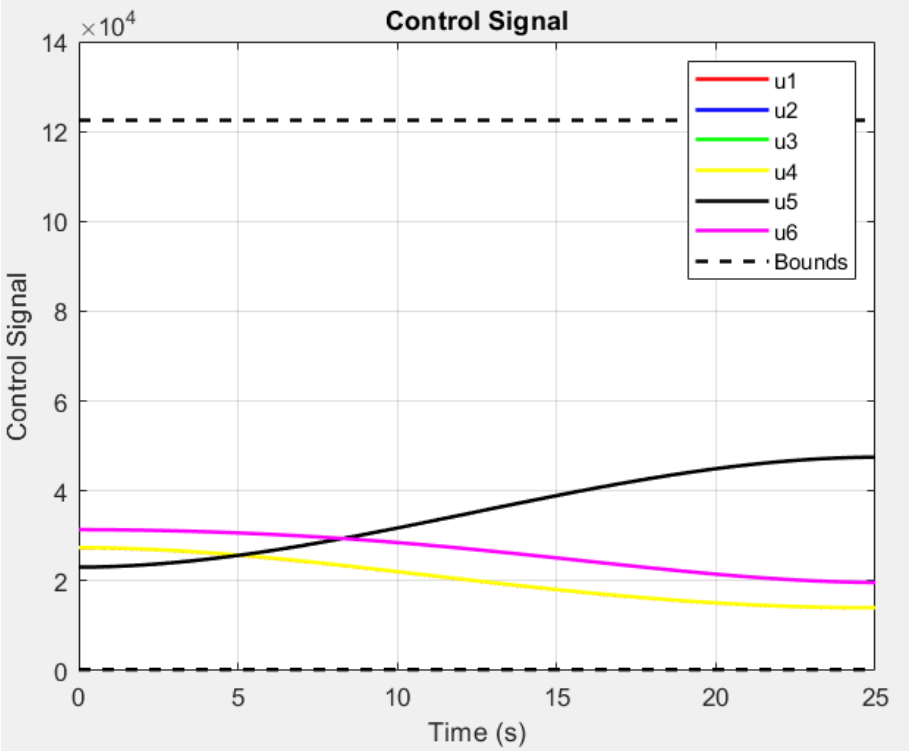


Figure D.15: Control Signal (Hz^2) over the trajectory

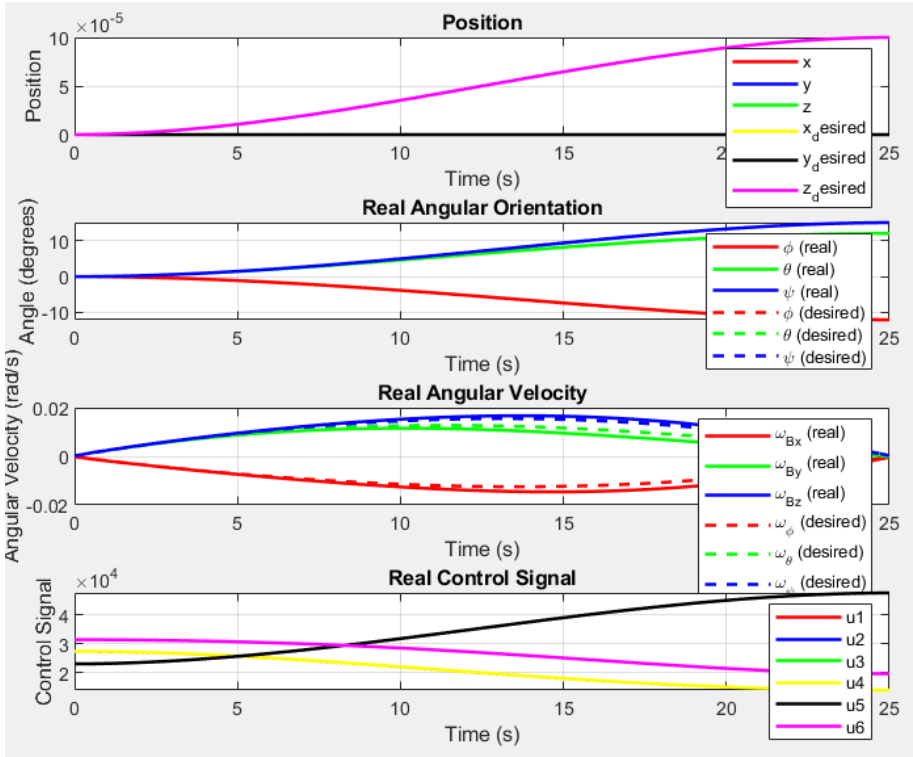
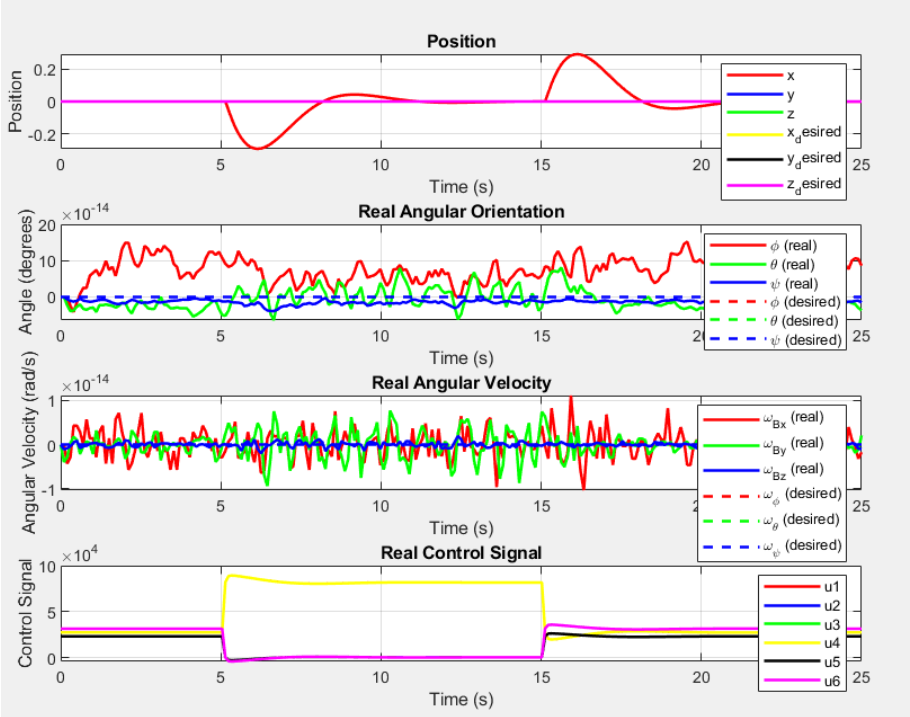


Figure D.16: Tracked position (m), orientation, angular velocity and Control Signal (Hz^2)

- Design 2: Hovering with External Disturbance (Max attainable)



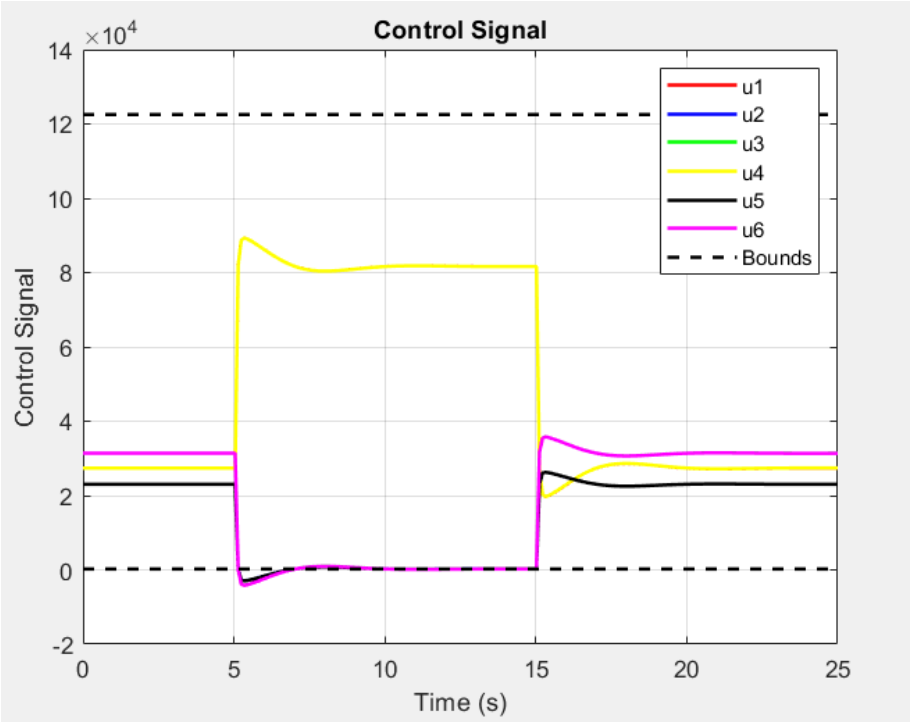


Figure D.17: Tracked position (m), orientation, angular velocity and Control Signal (Hz^2) (Top) and Control Signal (Hz^2) over the trajectory (Bottom)

- Design 3: Reorienting while hovering

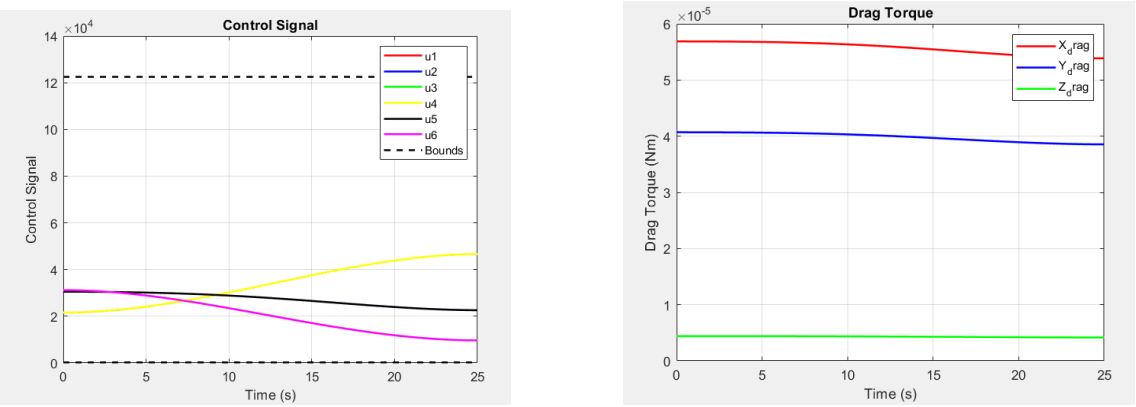


Figure D.18: Control Signal (Hz^2) (Left) and Drag Torque (Right) over the trajectory

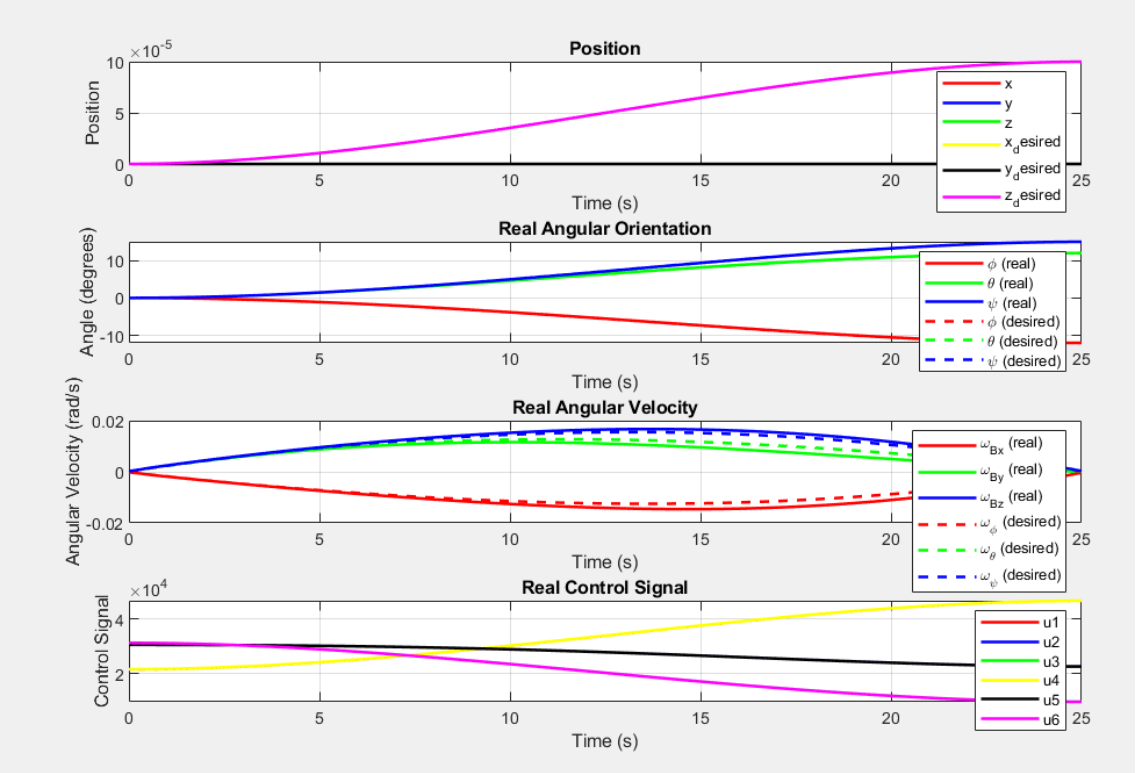


Figure D.19: Tracked position (m), orientation, angular velocity and Control Signal (Hz^2)

- Design 3: Hovering with External Disturbance (Max attainable)

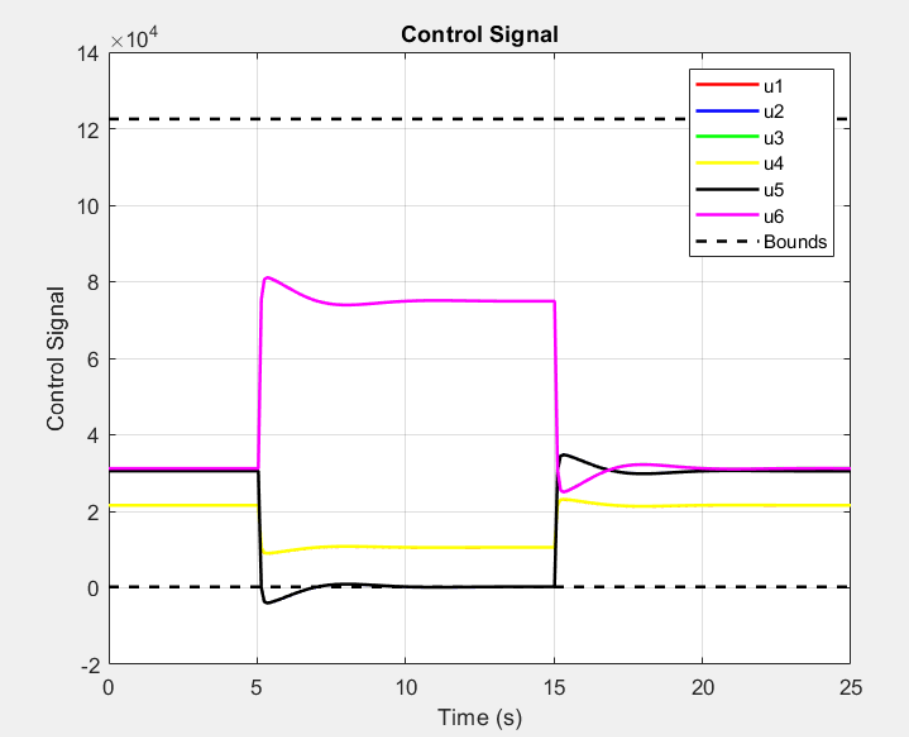


Figure D.20: Control Signal over the trajectory (Hz^2)

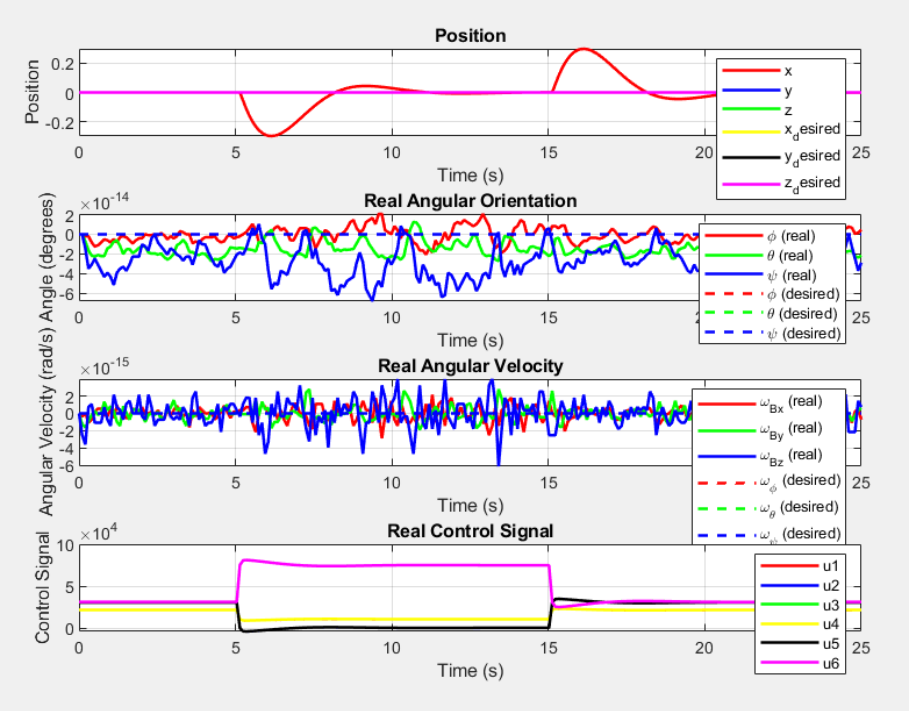


Figure D.21: Tracked position (m), orientation, angular velocity and Control Signal (Hz^2)

E
Appendix: Discussion

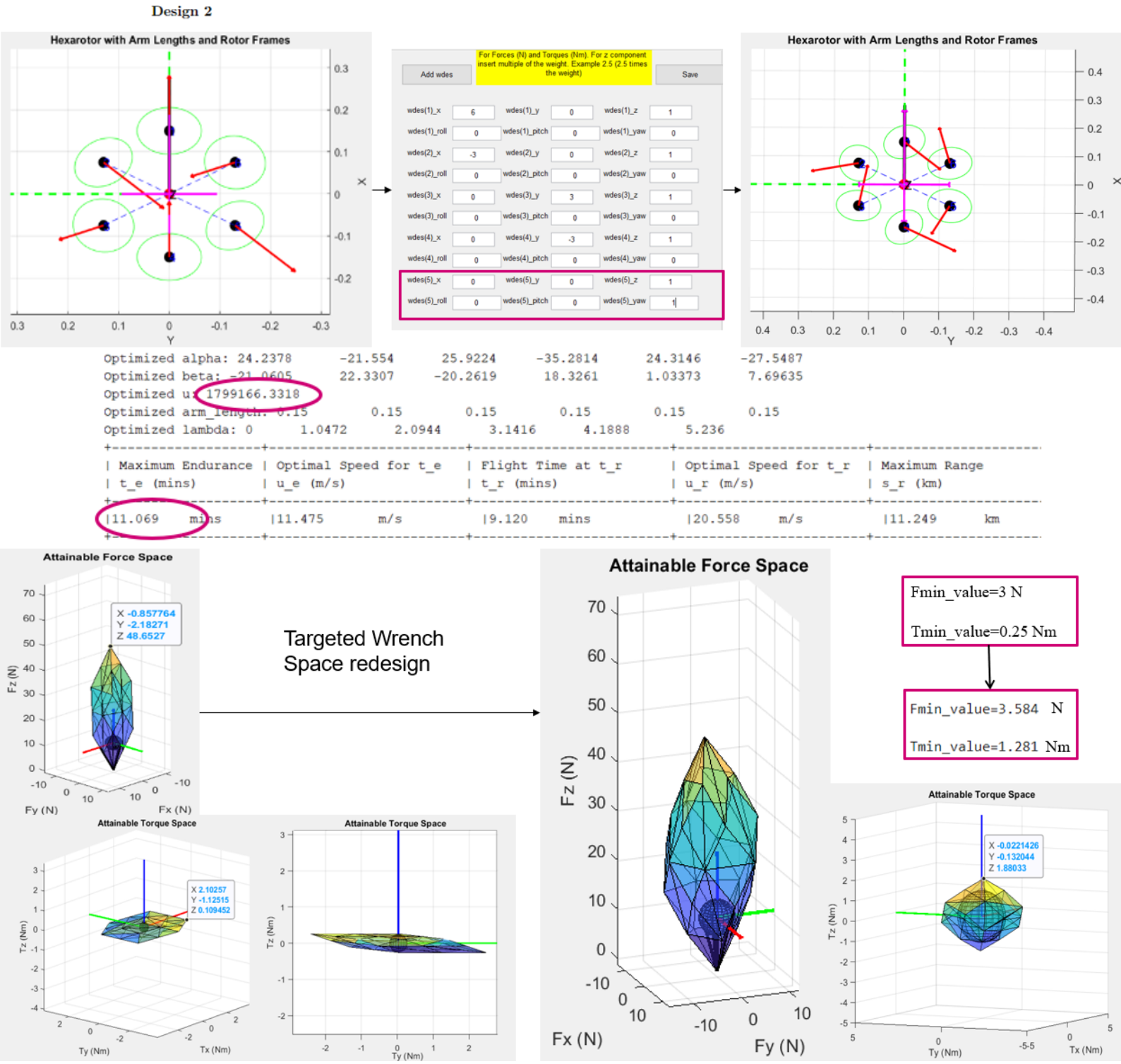


Figure E.1: Targeted Redesign based on the required improvement of 1Nm in z-axis (top). Output of Optimal Design script with larger control effort and lower endurance (middle). Old wrench space (bottom left) and new larger redesigned Wrench space (right bottom), with Fmin and Tmin significantly larger and max torque around z-axis 1.8 Nm.

F
Appendix: Block Schemes

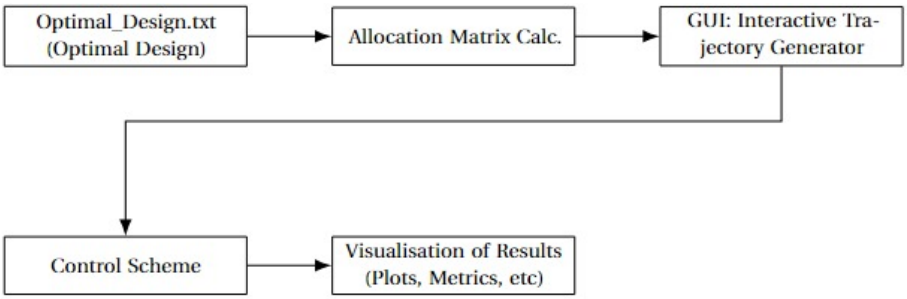


Figure F.1: Simulator Script: Block Scheme

Bibliography

Andersson, J. A., J. Gillis, G. Horn, J. B. Rawlings and M. Diehl (2019), CasADi: a software framework for nonlinear optimization and optimal control, *Mathematical Programming Computation*, **vol. 11**, pp. 1–36.

AUTOASSESS (2024a), About Autoassess.
https://autoassess.eu/about/

AUTOASSESS (2024b), Autoassess – AI and robotics for safe vessel inspection.
https://autoassess.eu/

Bauersfeld, L. and D. Scaramuzza (2022), Range, Endurance, and Optimal Speed Estimates for Multicopters, **vol. 7**, no.2, pp. 2953–2960, ISSN 2377-3774, doi:10.1109/lra.2022.3145063.

Bonnin-Pascual, F., A. Ortiz, E. Garcia-Fidalgo and J. P. Company-Corcoles (2019), A reconfigurable framework to turn a MAV into an effective tool for vessel inspection, *Robotics and Computer-Integrated Manufacturing*, **vol. 56**, pp. 191–211, ISSN 0736-5845, doi:10.1016/j.rcim.2018.09.009.

Bosscher, P., A. Riechel and I. Ebert-Uphoff (2006), Wrench-feasible workspace generation for cable-driven robots, **vol. 22**, no.5, pp. 890–902, ISSN 1552-3098, doi:10.1109/tro.2006.878967.

Brescianini, D. and R. D'Andrea (2016), Design, modeling and control of an omni-directional aerial vehicle, in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, doi:10.1109/icra.2016.7487497.

Casadi.org (2018), CasADi - Docs.
https://web.casadi.org/docs/

CORDIS, c. (2014), Inspection Capabilities for Enhanced Ship Safety, https://cordis.europa.eu/project/id/605200.

CORDIS, c. (2017), Marine INspection rObotic Assistant System, https://cordis.europa.eu/project/id/233715.

Faessler, M., A. Franchi and D. Scaramuzza (2017), Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories, **vol. 3**, no.2, pp. 620–626.

Gazebo (2024), Gazebosim.
https://gazebosim.org/home

Github.io (2020), Ipopt: Documentation.
https://coin-or.github.io/Ipopt/index.html

Hamandi, M., Q. Sable, M. Tognon and A. Franchi (2021a), Understanding the omnidirectional capability of a generic multi-rotor aerial vehicle, in *2021 Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO)*, IEEE, pp. 1–6, doi:10.1109/airpharo52252.2021.9571051.

Hamandi, M., K. Sawant, M. Tognon and A. Franchi (2020), Omni-Plus-Seven (O7+): An Omnidirectional Aerial Prototype with a Minimal Number of Unidirectional Thrusters, in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, pp. 754–761, doi:10.1109/icuas48674.2020.9214065.

Hamandi, M., F. Usai, Q. Sablé, N. Staub, M. Tognon and A. Franchi (2021b), Design of multirotor aerial vehicles: A taxonomy based on input allocation, **vol. 40**, no.8–9, pp. 1015–1044, ISSN 1741-3176, doi:10.1177/02783649211025998.

Ikeda, T., S. Yasui, M. Fujihara, K. Ohara, S. Ashizawa, A. Ichikawa, A. Okino, T. Oomichi and T. Fukuda (2017), Wall contact by octo-rotor UAV with one DoF manipulator for bridge inspection, in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, doi:10.1109/iros.2017.8206398.

Jiang, Q., D. Mellinger, C. Kappeyne and V. Kumar (2011), Analysis and Synthesis of Multi-Rotor Aerial Vehicles, in *Volume 6: 35th Mechanisms and Robotics Conference, Parts A and B*, ASMEDC, IDETC-CIE2011, pp. 711–720, doi:10.1115/detc2011-47114.

Kiso, K., T. Ibuki, M. Yasuda and M. Sampei (2015), Structural optimization of hexrotors based on dynamic manipulability and the maximum translational acceleration, in *2015 IEEE Conference on Control Applications (CCA)*, IEEE, pp. 774–779, doi:10.1109/cca.2015.7320711.

Lightning, X. (2024), Xnova Lightning V2 2208 2500KV Set.
https://www.rcheli-store.de/Quadcopters/FPV-Racer/FPV-Racter-Motors/Xnova-Lightning-V2-2208-2500KV-Set.htm?shop=rcheli_en&SessionId=&a=article&ProdNr=CPD-Xn-Light-08-25-V2N+Set&t=15578&c=32452&p=32452

Mathworks (2024), Create Default waypointTrajectory.
https://nl.mathworks.com/help/nav/ref/waypointtrajectory-system-object.html

Nava, G., Q. Sable, M. Tognon, D. Pucci and A. Franchi (2020), Direct Force Feedback Control and Online Multi-Task Optimization for Aerial Manipulators, **vol. 5**, no.2, pp. 331–338, ISSN 2377-3774, doi:10.1109/lra.2019.2958473.

Nikou, A., G. C. Gavridis and K. J. Kyriakopoulos (2015), Mechanical design, modelling and control of a novel aerial manipulator, in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, volume 83, IEEE, pp. 4698–4703, doi:10.1109/icra.2015.7139851.

Openrobots (a), genom3.
https://git.openrobots.org/projects/genom3

Openrobots (b), Telekyb3 - Documentation.
https://git.openrobots.org/projects/telekyb3/pages/index

Openrobots (c), uavatt-genom3.
https://git.openrobots.org/projects/uavatt-genom3/pages/README

Openrobots (d), uavpos-genom3.
https://git.openrobots.org/projects/uavpos-genom3/pages/README

Park, S., J. Lee, J. Ahn, M. Kim, J. Her, G.-H. Yang and D. Lee (2018), ODAR: Aerial Manipulation Platform Enabling Omnidirectional Wrench Generation, **vol. 23**, no.4, pp. 1907–1918, ISSN 1941-014X, doi:10.1109/tmech.2018.2848255.

Rajappa, S., M. Ryll, H. H. Bülthoff and A. Franchi (2015), Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers, in *2015 IEEE international conference on robotics and automation (ICRA)*, IEEE, pp. 4006–4013.

Rashad, R., P. Kuipers, J. Engelen and S. Stramigioli (2017), Design, Modeling, and Geometric Control on SE(3) of a Fully-Actuated Hexarotor for Aerial Interaction, doi:10.48550/ARXIV.1709.05398.

Ryll, M., D. Bicego and A. Franchi (2016), Modeling and control of FAST-Hex: A fully-actuated by synchronized-tilting hexarotor, in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, doi:10.1109/iros.2016.7759271.

Ryll, M., G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale and A. Franchi (2017), 6D physical interaction with a fully actuated aerial robot, in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, doi:10.1109/icra.2017.7989608.

Strawson, J., P. Cao, T. Bewley and F. Kuester (2021), Rotor Orientation Optimization for Direct 6 Degree of Freedom Control of Multirotors, in *2021 IEEE Aerospace Conference (50100)*, volume abs 1801 4581, IEEE, pp. 1–12, doi:10.1109/aero50100.2021.9438375.

Tognon, M. and A. Franchi (2018), Omnidirectional Aerial Vehicles With Unidirectional Thrusters: Theory, Optimal Design, and Control, **vol. 3**, no.3, pp. 2277–2282, ISSN 2377-3774,

doi:10.1109/lra.2018.2802544.

Valyon, J. and G. Horvath (2007), A Sparse Robust Model for a Linz-Donawitz Steel Converter, in *2007 IEEE Instrumentation & Measurement Technology Conference IMTC 2007*, IEEE, pp. 1–6, ISSN 1091-5281, doi:10.1109/imtc.2007.379018.

Voliro.com (2023), Wind Turbine LPS drone inspection case study. https://voliro.com/case-studies/wind-turbine-lps-drone-inspection

Wächter, A. and L. T. Biegler (2006), On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Mathematical programming*, **vol. 106**, pp. 25–57.

Xu, J. and D. Saldaña (2023), Finding Optimal Modular Robots for Aerial Tasks, in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, volume 100, IEEE, pp. 11922–11928, doi:10.1109/icra48891.2023.10160555.