POLITECNICO
MILANO 1863

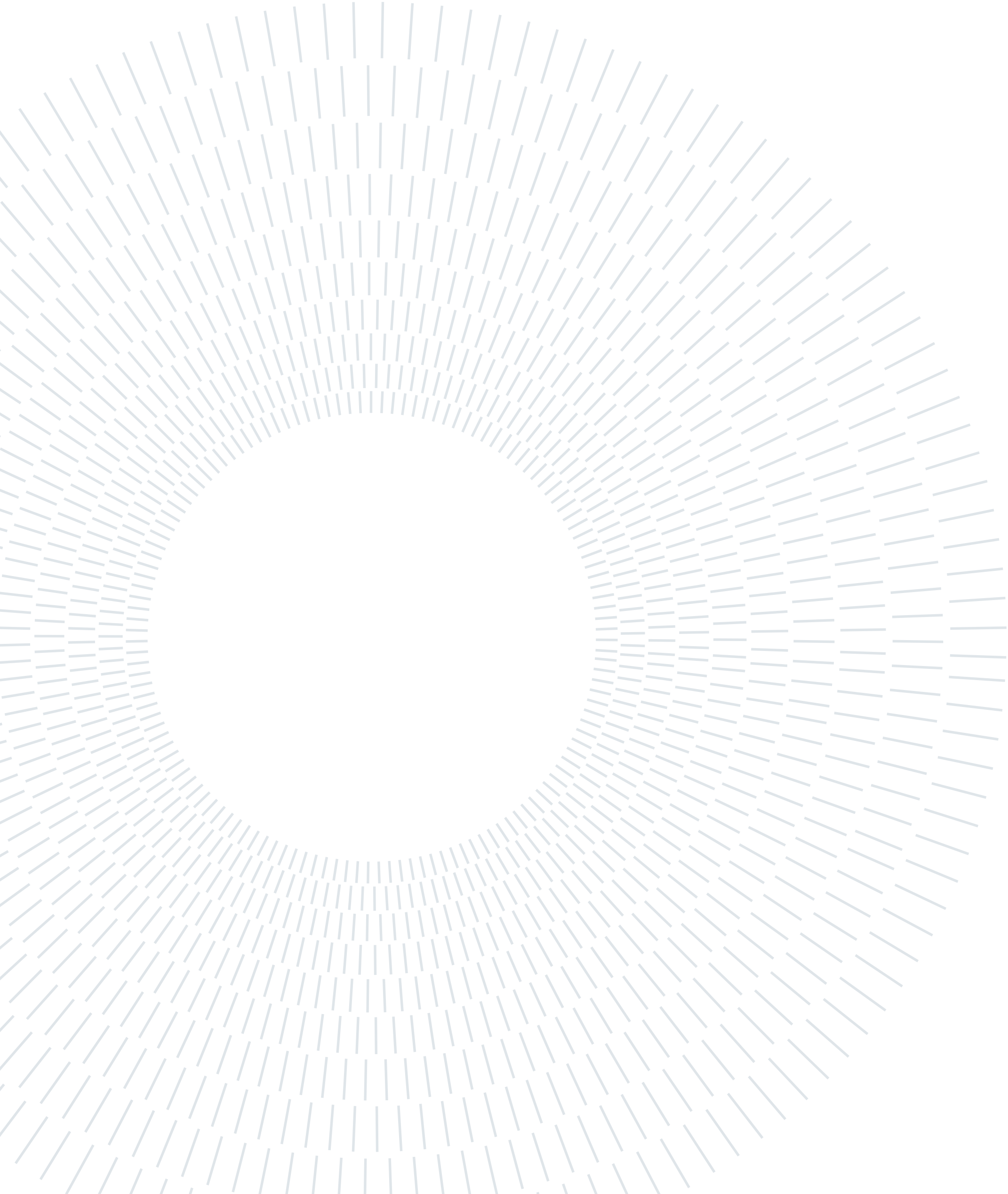
SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Object-oriented modelling and dynamic inversion for quadrotor simulation and control

TESI DI LAUREA MAGISTRALE IN
AERONAUTICAL ENGINEERING - INGEGNERIA AERONAUTICA

Author: **Atefeh Esmaeilzadeh Rostam**

Student ID: 10774972
Advisor: Prof. Marco Lovera
Co-advisors: Jan Brugård, Wolfram Mathcore
Academic Year: 2022-2023



Abstract

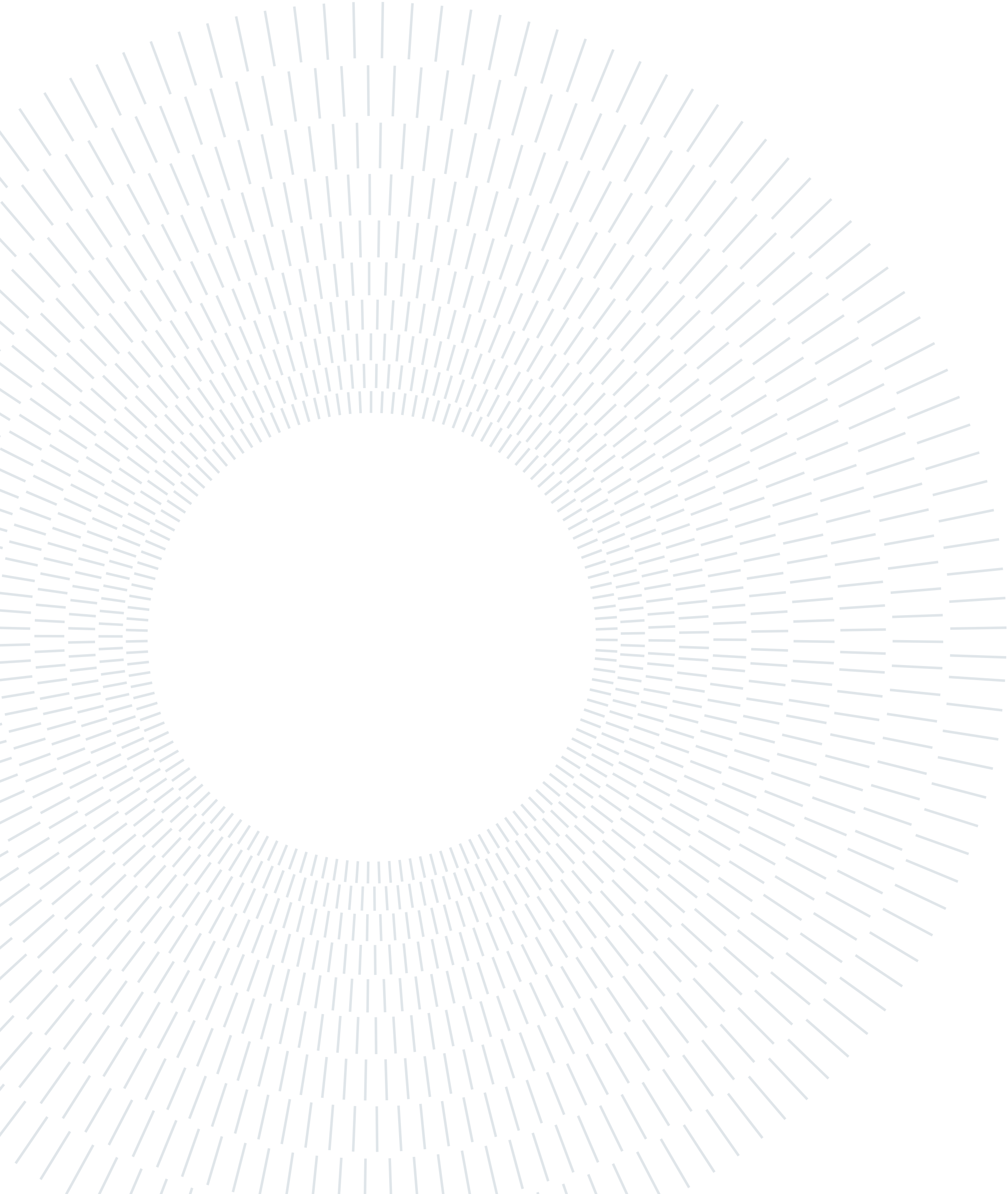
The objective of this research is to develop a model of a quadrotor and its components using the Wolfram System Model and Modelica language to create new blocks and utilities. Subsequently, a classical linear controller is implemented on System Modeler. This involves establishing a theoretical background on quadrotor kinematics and dynamics, as well as developing the model of its components such as mass body, electric motors, sensors and propellers.

The goal for designing the quadrotor and controller for System Modeler is to integrate a multi-rotor package into the aircraft library for the upcoming system modeler release. This requires a user-friendly model with intuitive interfaces and visualizations. Thus, a classical linear cascade PID controller emerges as a reasonably suitable choice for the conceptual design stage. Despite the highly nonlinear nature of quadrotor dynamics, a linear controller can be effective to a certain extent. Therefore, a nonlinear approach, specifically the Incremental Non-linear Dynamic Inversion (INDI) control algorithm, is investigated for attitude control.

The INDI algorithm is based on the canceling out the nonlinearity of the system, by using the dynamic of the system which allows to use the linear controller to achieve the desired behavior in the quadrotor. In contrast to classical nonlinear dynamic inversion, INDI does not rely on the precise system model, thus enhancing system robustness against uncertainty. However, it requires state derivative estimation and the suppression of high-frequency noise using a low-pass filter. The INDI method is simulated using MATLAB and Simulink, and the results are compared with those of a cascade PID controller.

The simulation results indicate that the performance of the INDI method slightly improved compared to the original PID controller especially for higher velocity and for more aggressive trajectories while requiring less computational power and fewer parameters to tune.

Keywords: WSM, INDI, Attitude control, Position control



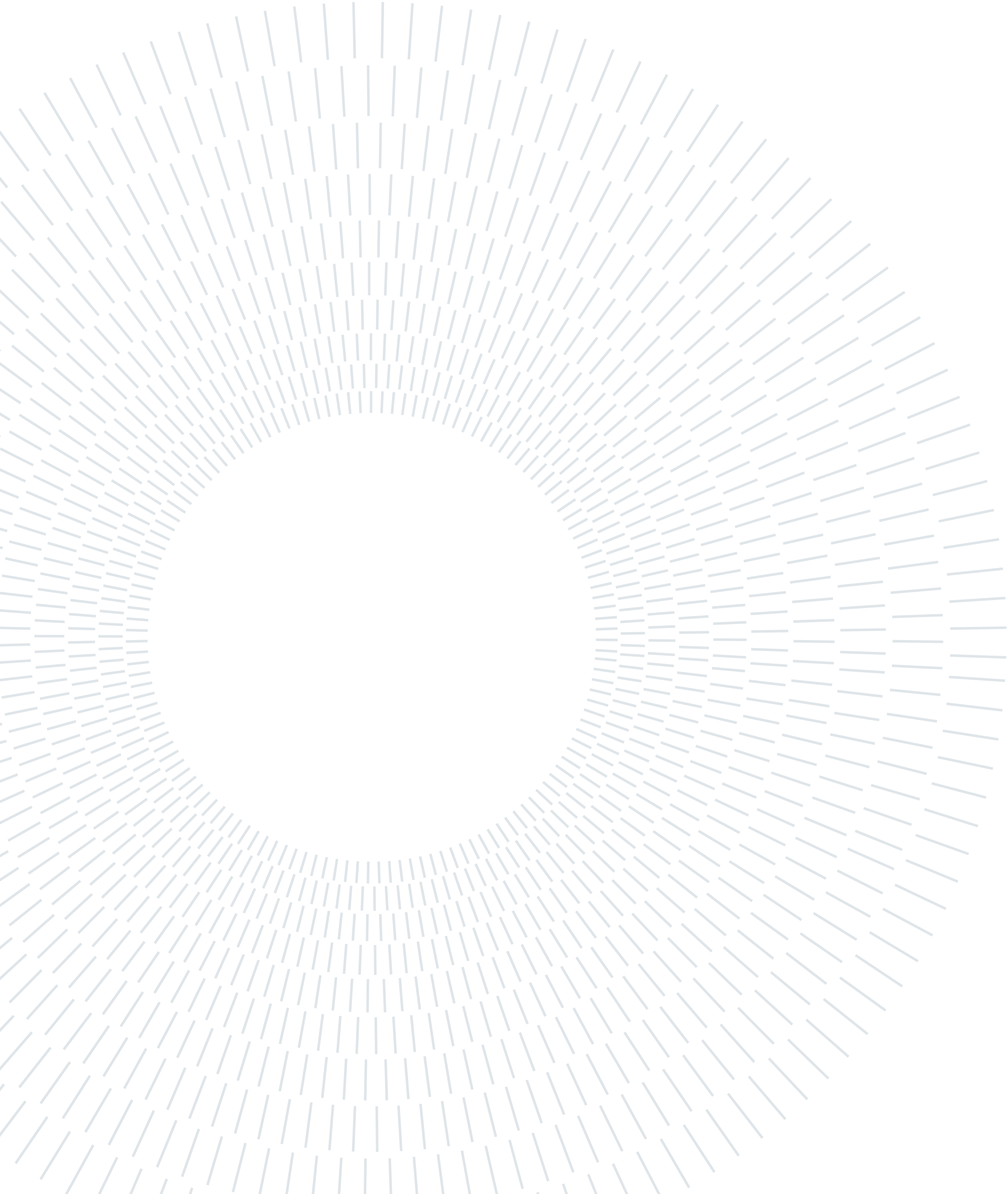
Abstract in lingua italiana

L'obiettivo di questa ricerca è sviluppare un modello di una quadratica e delle sue componenti utilizzando il modello del sistema Wolfram e il linguaggio Modelica per creare nuovi blocchi e funzioni. Successivamente, su System Modeler viene implementato un controller lineare classico. Ciò comporta la creazione di un background teorico sulla cinematica e dinamica quadratica, nonché lo sviluppo del modello dei suoi componenti come corpi di massa, motori elettrici, sensori ed eliche. L'obiettivo per la progettazione del quadritore e del controller per System Modeler è integrare un pacchetto multi-rotore nella libreria dell'aereo per la prossima versione del system modeler. Ciò richiede un modello user-friendly con interfacce e visualizzazioni intuitive. Pertanto, un classico controller PID lineare in cascata emerge come una scelta ragionevolmente adatta per la fase di progettazione concettuale.

Nonostante la natura altamente non lineare della dinamica quadratica, un controllore lineare può essere efficace fino a un certo punto. Pertanto, per il controllo dell'assetto viene studiato un approccio non lineare, in particolare l'algoritmo di controllo Incrementale Non Lineare Inversione Dinamica (INDI). L'algoritmo INDI si basa sull'annullamento della non linearità del sistema, utilizzando la dinamica del sistema che consente di adoperare il controller lineare per ottenere il comportamento desiderato nel quadrotore. A differenza della classica inversione dinamica non lineare, INDI non si basa sul modello di sistema preciso, migliorando così la robustezza del sistema contro le incertezze. Tuttavia, richiede la stima della derivata dello stato e la soppressione del rumore ad alta frequenza utilizzando un filtro passa-basso. Il metodo INDI viene simulato utilizzando MATLAB e Simulink e i risultati vengono confrontati con quelli di un controller PID in cascata.

I risultati della simulazione indicano che le prestazioni del metodo INDI sono leggermente migliorate rispetto al controller PID originale, soprattutto per velocità più elevate e per traiettorie più aggressive, richiedendo allo stesso tempo meno potenza di calcolo e meno parametri da regolare.

Parole chiave: WSM, INDI, controllo della direzione, controllo della posizione



Contents

1 | Introduction

This chapter provides an introductory overview of the thesis topic, including its background, the motivation driving the research, objectives and an outline of the structure of the thesis report.

1.1. Background

The inception of quadrotors can be traced back to the early 20th century when Étienne Oehmichen, a French engineer, developed one of the earliest manned quadrotor helicopters, the "Oehmichen No. 2," in the late 1920s. However, it wasn't until the late 20th and early 21st centuries that quadrotor technology truly began to advance.

With developments in electronics, materials, and control systems, modern quadrotor drones emerged as versatile and highly maneuverable aerial platforms. Their applications span a wide range of fields, including aerial photography, surveillance, agriculture, and research. Quadrotors offer a multitude of advantages that make them invaluable tools for research across various disciplines. Their exceptional maneuverability, coupled with versatility in payload options, enables them to perform diverse tasks such as environmental monitoring, disaster response, and wildlife observation with precision. Moreover, their accessibility and relative affordability make them attainable for researchers with varying budgets, while their inherent safety features ensure minimal risk to both operators and the surrounding environment.

Facilitating real-time data acquisition, quadrotors enable researchers to promptly analyze dynamic scenarios and make informed decisions. Additionally, their ease of experimentation makes them ideal platforms for testing new technologies and algorithms, fostering innovation in fields such as robotics and artificial intelligence. Overall, the capabilities of quadrotors empower researchers to explore new frontiers and address pressing challenges with efficiency and effectiveness.

One of the primary advantages of quadrotors is their utility as a test-bed for experimenting with various control strategies. Due to their maneuverability and versatility and being unmanned, quadrotors provide an ideal platform for researchers and engineers to develop, test, and refine control algorithms. This capability allows for experimentation with different linear non-linear, and adaptive control methodologies By testing control strategies on quadrotors, researchers can gain valuable insights into their performance in real-world scenarios, leading to advancements in autonomous navigation, stability control, and overall flight performance.

The process of designing and simulating a quadrotor typically begins with developing a model tailored the specific objectives, application, and desired level of complexity. Then, various controllers are designed based on the operational mission and design criteria. And then optimization of the design can be undertaken to align with the specified requirements. This research uses two platforms for modeling and simulating the quadrotor.

The first part, focuses on model-based design, where the quadrotor model is constructed using Wolfram System Modeler (WSM). WSM facilitates the modeling of the quadrotor by utilization of components available in the Modelica standard library, each containing its mathematical model. By interconnecting these blocks, the interactions between components can be accurately captured. Subsequently, a classical linear controller is designed. Parameterizing the components enables easy adjustment of parameters and facilitates simulation by importing various quadrotor data through an comfortable interface.

While linear controllers can be a useful starting point and offer insights into drone dynamics and control, they have limitations due to the highly nonlinear nature of quadrotor systems. Linear methods are effective only within a specific range around the trim point, restricting the operational range of the quadrotor. To address this constraint, nonlinear controllers are developed to expand the functionality of the quadrotor. Therefore, the second part introduces the development of a nonlinear controller algorithm for the ANT-R drone, which was created at the Aerospace System and Control Laboratory of Politecnico di Milano [3].

1.2. Thesis objectives

This thesis is divided into two primary sections. The first part involves developing a model of the quadrotor and its controller for the multirotor package of Aircraft library of the Wolfram System Modeler (WSM) with the Modelica language using the model based design approach. The objective is to create a model that closely resembles a real quadrotor while remaining user-friendly and easy to understand. This entails developing components, defining parameters, and creating an intuitive interface, all accompanied by comprehensive documentation.

In the second part, the focus shifts to studying the nonlinear controller method, specifically Incremental Nonlinear Dynamic Inversion (INDI) control. This part primarily concentrates on implementing INDI for attitude control of the quadrotor. The implementation is carried out using MATLAB and Simulink. The objectives can be listed as below:

- Modeling the nonlinear model of the quadrotor components and the full model on the WSM.
- Design cascade linear P-PID controller for quadrotor on WSM.
- Integration of the full model and create examples on WSM.
- Study nonlinear approach for quadrotor control: NDI and INDI.
- Implementation of the INDI. method on a nonlinear simulator using MATLAB and Simulink.
- Comparison of linear and nonlinear controllers.

1.3. Outline

This section provides a concise overview of the report's structure, outlining a brief summary of the content covered in each chapter.

- **Chapter 2:** This chapter studies reference frames, transformations between these frames, and provides preliminary information necessary for the mathematical modeling of quadrotor kinematics and dynamics.
- **Chapter 3:** This chapter begins with an introduction to the concept of object oriented modeling, model-based design and the corresponding modeling and simulation tool (WSM). It proceeds by detailing the breakdown of the components of the developed quadrotor, including the mass body of the quadrotor and the modeling of the propulsion system, which comprises the electric motor and propeller model and the required utilities. The chapter concludes by integrating all components to present the full model of the quadrotor.
- **Chapter 4:** This chapter introduces the methodology for designing the classical linear controller and provides a details of the implementation of attitude and position controllers within WSM. chapter 4 concludes with the integration of the controllers.
- **Chapter 5:** In this chapter, two flight scenarios are modeled within WSM, and the simulation results are presented. In the discussion section, the capabilities and limitations of the model are introduced.
- **Chapter 6:** In this chapter, a nonlinear control method called non-linear dynamic

inversion (NDI) and its incremental version are introduced. The mathematical model of the INDI for the quadrotor is derived, followed by a detailed explanation of its implementation using MATLAB and Simulink.

- **Chapter 7:** In this chapter, the simulation results for the applied control algorithm are presented, followed by a discussion comparing the outcomes with those of the linear control method.
- **Chapter 8:** The research concludes with a summary of outcomes and followed by suggestions for future development.

2 | Modeling of the quadrotor

This chapter outlines the mathematical depiction of the quadrotor, including the transformation matrix, as well as the kinematics and dynamics associated with it.
To accurately model the quadrotor and its reference frames, certain assumptions need to be taken into account.

- Quadrotor is considered rigid body.
- Propellers are rigid.
- Quadrotor frame is considered symmetric.
- Quadrotor center of the mass coincides with the geometrical center of the frame.
- Quadrotor has electrical power system which has been supplied by batteries which makes the mass and moment of inertia of the quadrotor to be constant.
- Earth considered to be flat and non-rotating.

2.1. Reference frames

The initial stage in the modeling of a mechanical object involves the selection of appropriate reference frames. For fixed reference frame or inertial reference frame North-East-Down (NED) arrangement was selected, in which x and y axes are align with the meridian and parallel lines, respectively, while the third component directed downward to the Earth's center. Therefore, the inertia reference frame can be denoted as $F_E = (O_E; N; E; D)$, where O_E represent the reference frame origin, and N, E and D constitute the orthogonal vectors indicating the North, East, and Down directions respectively.
The choice of the body reference frame holds significant importance as it impacts both the inertia and actuator matrix. To do so, a choice must be made between two configurations: the plus (+) and cross (×) configurations. Each configuration carries its own set of advantages and disadvantages. For this thesis, the cross configuration is selected. In this arrangement, the first axis aligns with the symmetry plane and points forwards.

Following the right-handed coordinate system, the second axis extends towards the right, and the third axis points downward. Hence, the body reference frame can be defied as: $F_B = (O_B; x_B; y_B; z_B)$, where O_B denotes the origin of the frame located at the Center of the Mass (CoM), and (x_B, y_B, z_B) corresponds to the three orthogonal vectors in the specified directions. Figure 2.1 depicts the quadrotor configuration in two aforementioned reference frames.

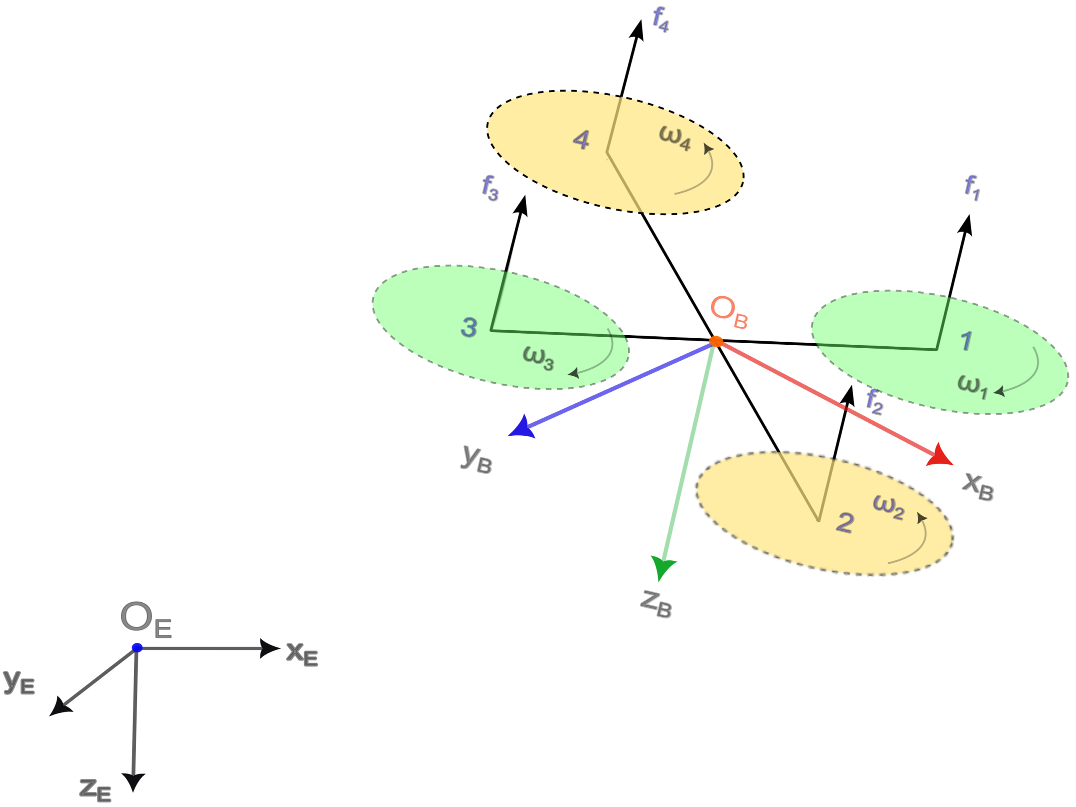


Figure 2.1: Quadrotor reference frames and the force and moments representations

2.1.1. Transformations

2.1.2. Euler angles

Euler angles are three angles to describe the orientation of a rigid body in the three dimensional Euclidean space. Moreover, they are introducing the orientation of a reference frame with respect to another reference frame through transformation of a coordinates of a point in a reference frame into another reference frame. Euler angles are indicated as ϕ, θ, ψ.

Using the to-from notation, a rotation matrix from system A to system B would be named R_{B-A}. Hence, any general coordinate vector P, in the A reference frame can be resolved in the B reference frame through Rotation matrix, R.

$$P_B = R_{B-A} P_A \tag{2.1}$$

Rotation matrices are indicated with $\mathbb{R}^3$ which can be the resultant of the rotation around one, two or three axes. Rotation matrices around an axis can be multiplied to attain an overall rotation matrix which represents a rotation from one reference frame to another one. Continuing with the well known quadrotor body to inertial rotation matrix, among all different sequences, the NASA standard rotation convention is used for the quadrotor. To do so, the first rotation occur around Z axis with the rotation angle of ψ, then a rotation of an angle θ around the new intermediate Y axis and finally a rotation of angle ϕ around the newer X axis.

$$R_X(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \tag{2.2}$$

$$R_Y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2.3}$$

$$R_Z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

Multiplying the three rotation matrix with aforementioned sequence for a body to earth transformation, results the so called Euler rotation matrix.

$$R_{BE}(\phi, \theta, \psi) = R_X(\phi) R_Y(\theta) R_Z(\psi) \tag{2.5}$$

$$R_{BE}(\phi, \theta, \psi) = \begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\theta)\sin(\psi) & -\sin(\theta) \\ \sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi) & \sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \sin(\phi)\cos(\theta) \\ \cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi) & \cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{2.6}$$

2.1.3. Quaternions

Quaternions, originating from the ingenuity of William Rowan Hamilton in 1843, expand upon the notion of complex numbers by introducing a four-dimensional framework. This innovation provides a robust methodology for expressing rotations within three-dimensional space.

Quaternions are widely employed across diverse domains because of their unique capability to represent spatial orientation and rotations without the limitations of gimbal lock. Among the various representations of quaternions, one method is through the use of unit quaternions. It presents the attitude matrix as a homogeneous quadratic function of the quaternion elements, eliminating the need for trigonometric calculations. Quaternions offer a more efficient means of describing rotations compared to the attitude matrix, as it contain only four components instead of nine and adhere to a single constraint—the norm constraint—rather than the six constraints enforced by orthogonality on the attitude matrix. Therefore, in quaternion representation, the initial three components represent a vector (ρ), while the fourth component represents a scalar as shown in equation (2.7).

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \rho \\ q_4 \end{bmatrix} \tag{2.7}$$

For a quaternion to be classified as a unit quaternion, it must have a magnitude of one, as illustrated below.

$$||q|^2 = ||\rho||^2 + ||q_4||^2 = 1 \tag{2.8}$$

In what follows some of the main mathematical features of quaternion will be presented which are helpful for the transformations between body and earth reference frames and the Euler angles.

The inverse of the unit quaternion is equivalent to its conjugate ($q^{-1} = \bar{q}$).

Quaternions make it simple to perform consecutive rotations by using quaternion multiplication.

$$R(q_1)R(q_2) = R(q_1 \otimes q_2) \tag{2.9}$$

Where $\otimes$ is the Hamiltonian product, which can be obtained by [4]:

$$q \otimes q' = \begin{bmatrix} q_4\rho' + q'_4\rho - \rho \times \rho' \\ q_4 q'_4 - \rho.\rho' \end{bmatrix} \tag{2.10}$$

In order to obtain the attitude matrix through quaternion, a homogeneous quadratic function should be defined as shown in equation (2.11):

$$R(q) = (q_4^2 - ||\rho||^2)I_3 + 2\rho\rho^T - 2q_4[\rho\times] \tag{2.11}$$

Where ρ^T is the transpose of the ρ, I_3 is a 3×3 matrix and the explicit demonstration of the cross product can be seen in equation (2.12):

$$[\rho\times] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{2.12}$$

Therefore, the attitude matrix reads:

$$R(q) = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_2q_1 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_3q_1 + q_2q_4) & 2(q_3q_2 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \tag{2.13}$$

Moreover, one could also derive the corresponding quaternion from an attitude matrix R using a modified of Shepperd's algorithm.

$$C_i = 4q_iq \qquad for \qquad i = 1, ..., 4 \tag{2.14}$$

Where q_i is the i-th component of the quaternion (q). Following equation (2.14), the C_i vectors can be obtained.

$$C_1 = \begin{bmatrix} 1 + R_{11} - R_{22} - R_{33} \\ R_{12} + R_{21} \\ R_{13} + R_{31} \\ R_{23} + R_{32} \end{bmatrix} \tag{2.15}$$

$$C_2 = \begin{bmatrix} R_{21} + R_{12} \\ 1 + R_{22} - R_{33} - R_{11} \\ R_{23} + R_{32} \\ R_{31} - R_{13} \end{bmatrix} \tag{2.16}$$

$$C_3 = \begin{bmatrix} R_{31} + R_{13} \\ R_{32} + R_{23} \\ 1 + R_{33} - R_{11} - R_{22} \\ R_{12} - R_{21} \end{bmatrix} \tag{2.17}$$

$$C_4 = \begin{bmatrix} R_{23} - R_{32} \\ R_{31} - R_{13} \\ R_{12} - R_{21} \\ 1 + R_{11} + R_{22} + R_{33} \end{bmatrix} \tag{2.18}$$

Where R_{ij}s are the components of the attitude matrix R.
Finally, to achieve the unit quaternion, the normalization should be done or each C_i.

2.2. Kinematics

The kinematics of a quadrotor, involves analyzing its movement independent of the forces causing it. Quadrotors can perform complex maneuvers because they have the capability to regulate thrust and torque individually at each rotor. Through adjusting the rotational speed of its rotors, a quadrotor can navigate in three-dimensional space, performing both translations and rotations. The mathematical equations governing quadrotor motion includes relations describing its position, velocity, and acceleration relative to time.

Considering unit quaternions for the quadrotor attitude representation for the second part of this project, the main quadrotor kinematic equation reads:

$$\dot{q} = \frac{1}{2} q \otimes \omega_b \tag{2.19}$$

Where ω_b is the angular speed in the body reference frame.

This equation provide a concise representation of the quadrotor's orientation dynamics in quaternion form. Quaternion-based representations offer advantages over Euler angles, such as reduced computational complexity since equation (2.19) is linear and avoidance of singularities.

To get the Euler Angles from quaternions, one can use equation (2.13).

It's worth noting that when dealing with small angles, the vector part of the quaternion closely approximates to half of the angles, denoted as $\rho \approx \frac{\phi}{2}, \frac{\theta}{2}, \frac{\psi}{2}$. Additionally, the fourth component of the quaternion, q_4, can be approximated to one in such cases ($q_4 = 1$).

2.3. Dynamics

One of the key parts of the modeling of a system to describe the physical behavior of the quadrotor is the dynamic model. The basis of the quadrotor dynamics calculations is Newton's second law for linear and angular motion.

2.3.1. Equation of motion

For linear (translational) motion the Newton second law states that the external forces (aerodynamic, gravity and thrust) acting on the quadrotor is equal to the time rate of change of the quadrotor's momentum as shown in equation (2.20), whereas, for angular motion the moments created by aerodynamic forces and thrust acting on the quadrotor need to be equal to the time rate of change in angular momentum as presented in equation (2.21).

$$F_b = \frac{d(m v_b)}{dt} \tag{2.20}$$

$$M_b = \frac{d(I \omega_b)}{dt} \tag{2.21}$$

Note that forces and moments of the quadrotor are defined in the body reference frames. Therefore, F_b and M_b are the force and moment in body reference frame. m is mass of the quadrotor, I is the second moment of Inertia, v_b is the velocity vector in the body reference frame and ω_b is the angular velocity vector in the reference frames.

The two equations (2.20) and (2.21), can be extended by assuming a consistent mass and inertia for the quadrotor as mentioned in the beginning of this chapter, given that quadrotors typically maintain a constant mass due to their battery power throughout the duration of their flight missions.

$$F_b = m\dot{v}_b + \omega_b \times (m v_b) \tag{2.22}$$

$$M_b = I\dot{\omega}_b + \omega_b \times (I \omega_b) \tag{2.23}$$

2.3.2. External forces and moments

The next step after writing the equations of motion is to define the forces and moments acting on the system. The main forces acting on the quadrotor are the gravitational force (F_g), which acts downward (in the positive direction of the third component of the inertia reference frame), and the thrust (T) generated by the propellers, which acts upward in the opposite direction to the third component of the body reference frame. Other forces, such as aerodynamic forces, can be modeled with a disturbance model rather than being separately included in the forces.

$$F_g = mgi_{z_e} \tag{2.24}$$

Where F_g is the gravitational force, m is the mass of the quadrotor and i_{z_e} is a unit vector showing the direction of the third component of the Earth reference frame (z_e).

$$T = -\sum T_i i_{z_b} \qquad for \qquad i = 1, ..., 4 \tag{2.25}$$

Where T is the total thrust of the quadrotor, T_i is the thrust of the each propeller and i_{z_b} is a unit vector showing the direction of the third component of the body reference frame (z_b). Force and moments of the rotors can be modeled as a quadratic relation of the rotor rotational speed.

$$T_i = k_\tau \omega_i^2 \tag{2.26}$$

$$Q_i = k_q \omega_i^2 \tag{2.27}$$

where ω_i is the rotation speed of each rotor, k_τ the rotor thrust coefficient, k_q the rotor torque coefficient. Figure 2.2 shows the force and monets acting on the quadrotor body. The summation of four upward forces generate the required lift force. Multiplying the lift force with arm length gives the resultant moment on the center of the mass. For a stable flight, The resultant moment in hover is equal to zero following the right hand rule, so, the quadrotor stays still in hover.

As you can see in the Figure 2.2, propeller 1 and 3 rotates clockwise while propeller 2 and 4 are rotating counter clockwise. The benefit of having two pairs of propellers rotating in the opposite direction is that they cancel out moment generates by the rotation of each rotors and prevents from the spin of the quadrotor, imagine if all of them were rotating in the same direction, according to the newton's third law the quadrotor would spin in the opposite direction of the rotation of the blades.

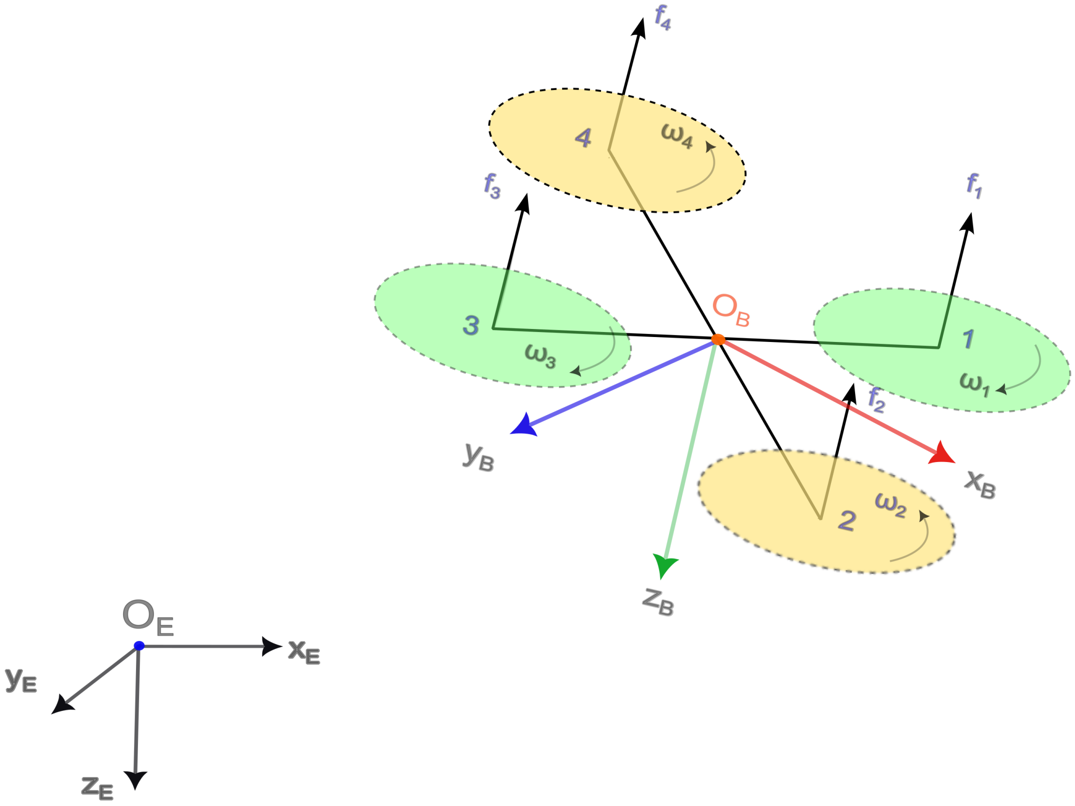
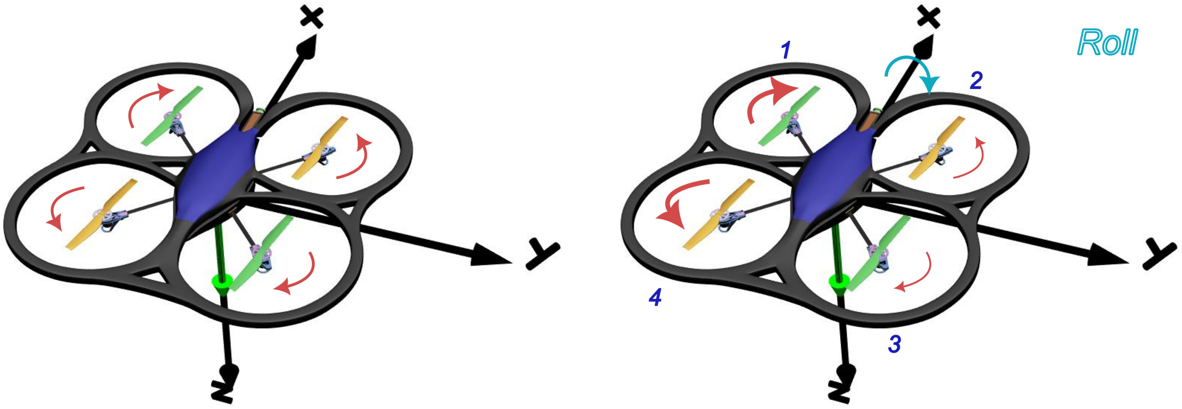
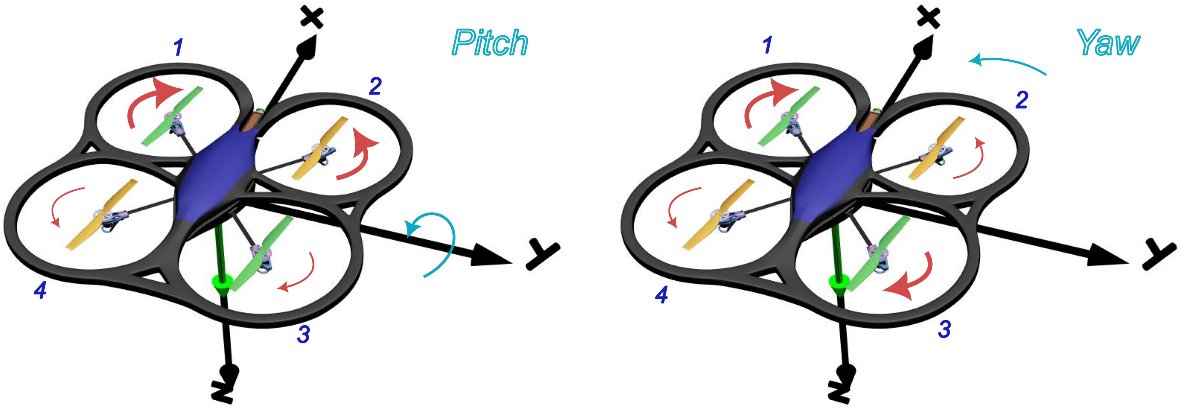


Figure 2.2: Quadrotor reference frames and the force and moments representations

Following this rotation arrangement, the pure vertical motion coming from throttle command can be obtained by increasing or decreasing the thrust of all propellers by the same amount 2.3a. As shown in the Figure 2.3b a positive roll motion can be obtained from the increase in the increment of the rotation speed of the propellers 2 and 3 and the same amount decrease of the propeller 1 and 4 which requires a direction change of the propeller 3 and four as well. A positive pitch results from the generating an incremental positive lift force in the positive direction of the propeller 1 and 2 and the negative force increment of the propeller 3 and 4 2.3c. Finally, yaw motion comes from the rotation of the all of the propellers in the same direction which makes the quadrotor itself to spin in the opposite direction 2.3d.

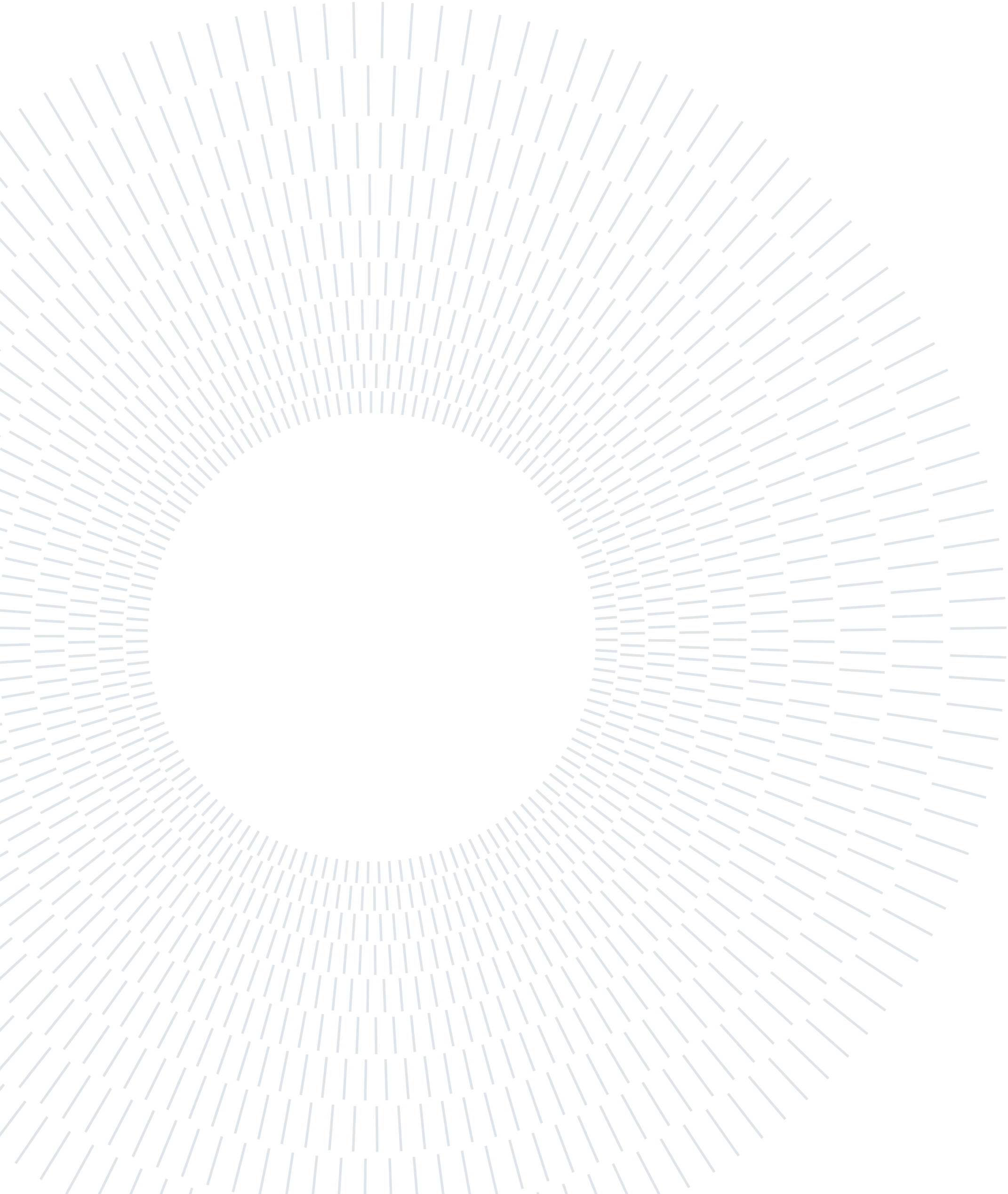


(a) Throttle (b) Roll motion



(c) Pitch motion (d) Yaw motion

Figure 2.3: Quadrotor motions



3 | Object-oriented modeling of the quadrotor

This chapter begins with an introduction to Wolfram System Modeler (WSM), a tool used for model-based design, and discusses the concept of model-based design. It then proceeds to outline the process of developing the components of a quadrotor and subsequently integrating these components into a complete quadrotor system.

Object-oriented modeling is a methodology used in software engineering to design and represent systems by modeling them as a collection of interacting objects. It is based on the principles of object-oriented programming, which organizes software design around data, or objects, rather than functions and logic.

3.1. Introduction of the WSM

Wolfram System Modeler, developed by Wolfram MathCore, is a model-based design platform for engineering as well as life-science modeling and simulation based on the Modelica language. It provides an interactive graphical modeling and simulation environment and a customizable set of component libraries. WSM offers a significant advantage in multi-domain modeling and system optimization through its visual tools, which facilitate a clear understanding of the model. This attribute makes it an good choice for conceptual modeling purposes.

Wolfram System Modeler's primary interface, Model Center, is an interactive graphical environment including a customizable set of component libraries. Models developed in Model Center can be simulated in the Simulation Center. The software also provides a tight integration with the Mathematica environment. Users can develop, simulate, document, and analyze their Wolfram System Modeler models within Mathematica notebooks [12].

3.1.1. Model based design

Model-based design (MBD) is a methodology that offers a mathematical and visual approach to tackle the challenges inherent in designing complex systems like control, signal processing, and communication systems. MBD is Widely used across industries such as motion control, industrial equipment, aerospace, and automotive applications, it serves as an efficient means of establishing a integrated framework for communication throughout the design process, while adhering to the development cycle (V-model).

Compared to traditional design methodologies, MBD offers numerous advantages. Instead of relying on complex structures and extensive software code, MBD enables designers to define plant models using intuitive building blocks representing continuous-time and discrete-time systems. Together with simulation tools, these models enable fast and easy prototyping, software testing, and validation.

One notable benefit of MBD is its ability to enhance the testing and verification processes. By leveraging simulation tools, designers can conduct comprehensive tests more efficiently compared to traditional methods. Moreover, MBD allows for the utilization of hardware-in-the-loop simulation, enabling dynamic effects on the system to be tested rapidly and effectively.

All in all, MBD revolutionizes the design process by providing a systematic approach that fosters efficient communication, rapid prototyping, and rigorous testing, ultimately leading to the development of robust and reliable systems.

3.2. Body of the quadrotor

To start with the modeling a mechanical system, the physical characteristics comprises of mass, inertia and the dimension of the system should be modeled.

"QuadrotorBody" component is the main physical component of the quadrotor that contains the mass and inertia of a quadrotor. It can move in six degrees of freedom, which include three attitude angular motions (roll, pitch, and yaw) and three positional changes (longitudinal, lateral, and vertical).

As can be seen in Figure 3.1, mass breakdown of the quadrotor comprises of the "bodyCenter", which represents the central mass of the quadrotor, and four other masses representing the mass of the propulsion systems, "body1", "body2", "body3", "body4". For each body component, the inertia and mass values is modifiable in property tab. Note that the mass and inertia properties of the four propulsion systems are equivalent.

The model also includes four arms connecting each propulsion system to the center of the mass of the quadrotor represents as "center" component which is a frame connector in the "Modelica" standard library. Note that, the arms have no mass.

Additionally, it includes a (CAD) model of the quadrotor for visualizing it in animation center. The connectors of the propulsion system frame are linked to the propulsion system in order to mount forces, moments, and rotational speed to the quadrotor body.

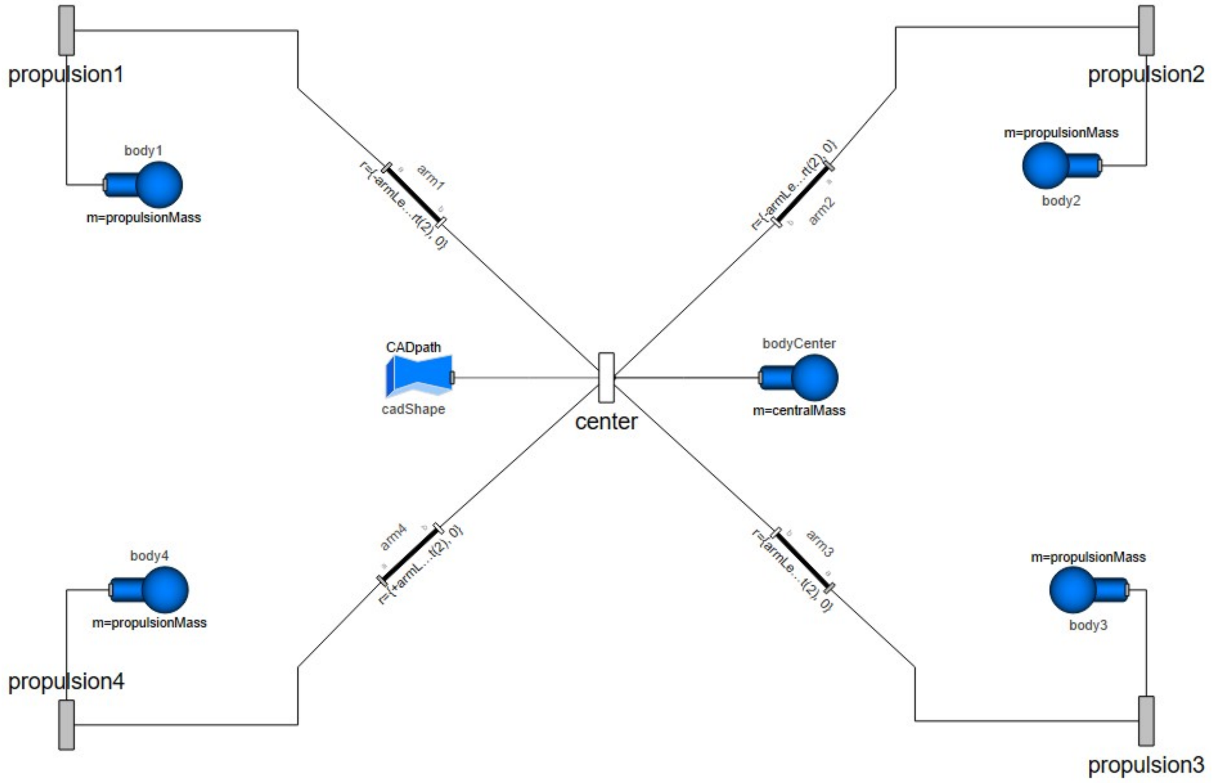


Figure 3.1: QuadrotorBody component containing mass and physical properties of the quadrotor

Table 3.1 summarizes the physical properties of a general quadrotor that has been modeled in WSM.

Variable	**Value**	**Unit**
bodyCenter mass	0.38	$[kg]$
propulsion system mass	0.01	$[kg]$
bodyCenter inertia	$diag[0.01, 0.01, 0.01]$	$[kg/m^2]$
propulsion system inertia	$diag[0.001, 0.001, 0.001]$	$[kg/m^2]$
arm length	0.192	$[m]$

Table 3.1: Physical information of the quadrotor body

3.3. Modeling the propulsion system

3.3.1. Electric motor

An electric motor is a system that converts electrical energy into mechanical energy. Electric motors typically operate through the interaction between the magnetic field and electric current of the motor to generate torque, which is then transmitted by the motor's shaft.

Electric motors can run on in two main configurations, direct current (DC), which usually powered from batteries or rectifiers, or on alternating current (AC), such as what you get from the power grid, inverters, or electrical generators. For the purpose of quadrotors, a variety of types of DC motors, especially brushless DC motors, have been used. Starting from the governing equation for DC motors, three main equations can be listed below. For the electrical part, using to Kirchhoff's voltage law (3.1):

$$v = Ri + L\frac{di}{dt} + e \tag{3.1}$$

Where v is the input voltage, R is the resistance, i is the current, and e is the voltage corresponds to electromotoric force (emf) which transforms the electrical energy into the mechanical energy through equation (3.2).

$$e = K_e \omega_m \tag{3.2}$$

Here K_e is the emf constant and ω_m is the rotation speed of the shaft. On the other hand for the mechanical part, the combination of the newton's second law leads to equation (3.3):

$$T_m - T_l = J\frac{d\omega_m}{dt} \tag{3.3}$$

Where T_l is the load torque to produce acceleration, J is the combination of the inertia of the motor and the inertia corresponds to the mechanical load, and electromagnetic torque T_m can be obtained from equation (3.4):

$$T_m = K_t i \tag{3.4}$$

Where K_t is the motor torque constant which is the same as K_e. For the first trial a simple DC motor has been designed on wolfram system modeler, the diagram view of the model can be seen in Figure 3.2. The First loop shows the electric circuit and the gray one represents the transmission of the electrical energy into mechanical energy.

Having the governing equations for the dc motor circuit, the model of the components of the DC electric motor has been implemented on wolfram system modeler. The benefit of using system modeler is that, the relevant components to model the dc motor are already included in the Electrical package of the Modelica standard library which with built-in mathematical representations. Figure 3.2 depicts the diagram view of the simplified electric motor for the quadrotor.

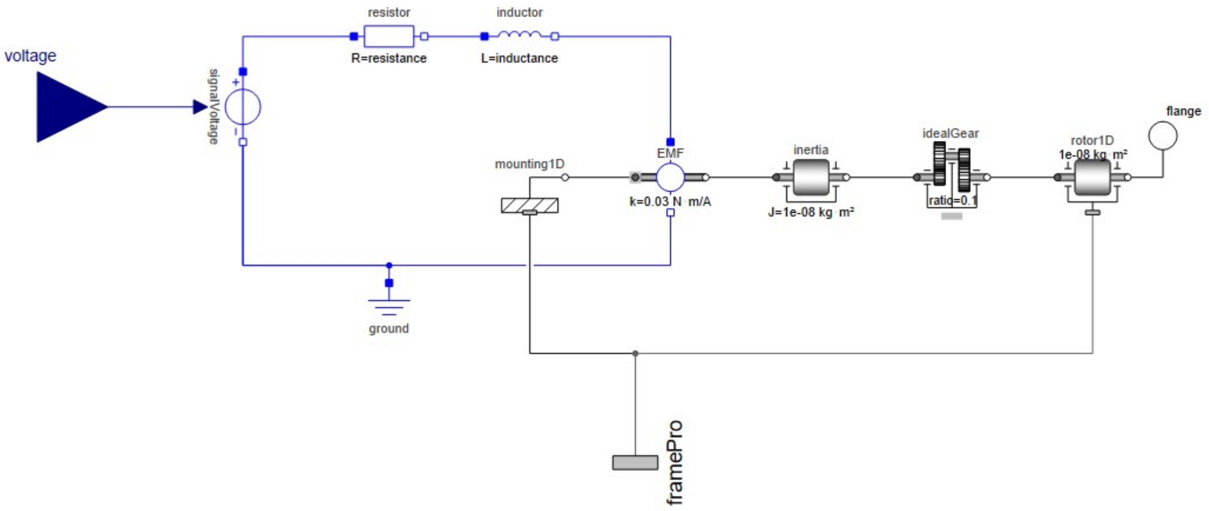


Figure 3.2: Simplified DC motor model developed on WSM

As can be interpreted from the Figure 3.2, the input of the electric motor is the commanded voltage while the output is the torque and rotational speed transmitted to the "framePro" connector and the "flange" components, respectively.

The electrical part comprises of the resistor, inductance and a voltage source with an emf as a connection component with the mechanical part.

The "mounting1D" component transmits one-dimensional support torque to the three-dimensional system, connecting to the "framePro" connector to link with the mass body of the propulsion system in the integrated model.

The mechanical part includes emf, the inertia of load inertia, an ideal gearbox which does not have inertia, elasticity, damping or backlash. In addition, a 1D-rotor is used to introduce the gyroscopic torques.

A more advanced model of the motor has been developed for a customer by knowing the datasheet of the motor [5], featuring an improved motor model integrated with an interface designed to connect it with the battery model.

3.3.2. Propeller

The propeller model receives angular rotation speed through the "flangeRotationSpeed" connector and transforms it into mechanical force. The propeller gain then translates this rotation speed into upward lift, which acts vertically in the z-axis, while it is assumed that there are no force components in the x and y directions.

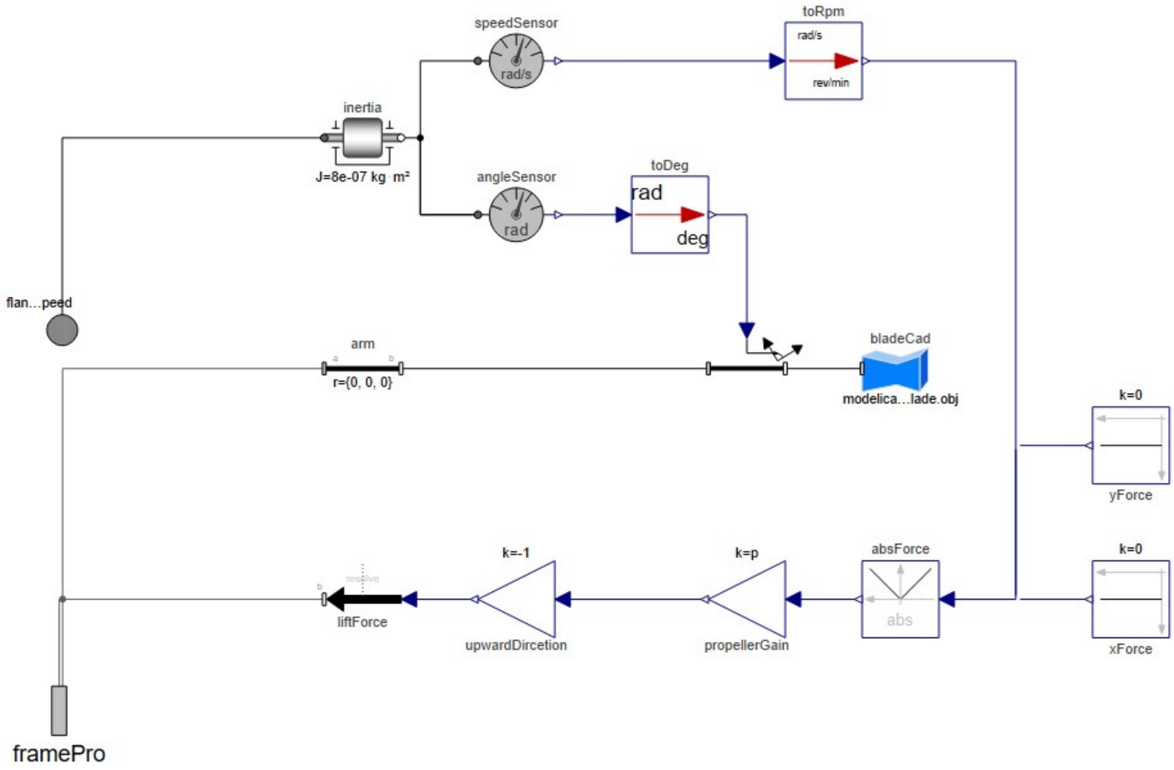


Figure 3.3: Simplified propeller model on WSM

Considering the Linear range of the operation, the simple equation has been considered for the force and motor RPM equation (3.5):

$$F_i = K\Omega_i^2 \quad for \quad i = 1, ..., 4 \tag{3.5}$$

Where, F_i is the thrust generated by each propeller, Ω_i is the angular rotation speed of the propeller. This simple mathematical model let us to define a propeller gain K_p for our operating range. While a more exact model of the propeller will be presented later.
Propeller model demonstrated in Figure 3.3 includes the following listed components:

- **speedSensor:** Measures the rotation speed of the propeller.
- **angleSensor:** Measure the angle corresponding to the rotation of the inertia component (rotor) to pass it to the variable rotation component for the rotation visualization in animation.
- **inertia:** Represents the rotational component (rotor) with an inertia.
- **arm:** Represents the distance between the propeller center to the propulsion system hub on the quadrotor body case which can be tuned by the user.
- **variableRotation:** Visualizes the rotation of the blades of the propeller in the animation center.
- **framePro:** This connector will be attached to the mass body to support the force, rotation, and CAD shape on the propulsion connector frame of the "Quadrotorbody".

- **liftForce:** Represents the lift force of the quadrotor
- **bladeCad:** A CAD model of a propeller is used for animation and one can replace them with another CAD file with the .obj extend. The geometric properties of the propeller can be set up by user. The rest of the blocks has been used to adjust the CAD model with the World and body reference frames.

In a recent project for a client, a more refined model of the propeller was developed by incorporating detailed specifications. This enhancement involved introducing an additional vector component for dynamic thrust into the existing model. The finalized equation is referenced as equation (3.6), and further elaboration on the modifications can be found in the work by Staples et al. (2013) [9].

$$Thrust = 4.392399 \times 10^{-8}.RPM.\frac{d^{3.5}}{\sqrt{p}}(4.23333 \times 10^{-4}.RPM.p - V_0) \tag{3.6}$$

Where, RPM is the rotation speed of the propeller (rotation per minute) propeller, d is the propeller diameter, p is the propeller pitch and V_0 is the propeller forward airspeed.

3.4. Quadrotor base model

QuadrotorBase model represents the complete drone model consists of the QuadrotorBody (representing mass and inertia of the drone), sensors, and propulsion system (including DC motor, gear and propeller). It takes input commands from controller and the outputs of the model are the translational and rotational position/angle and velocities.

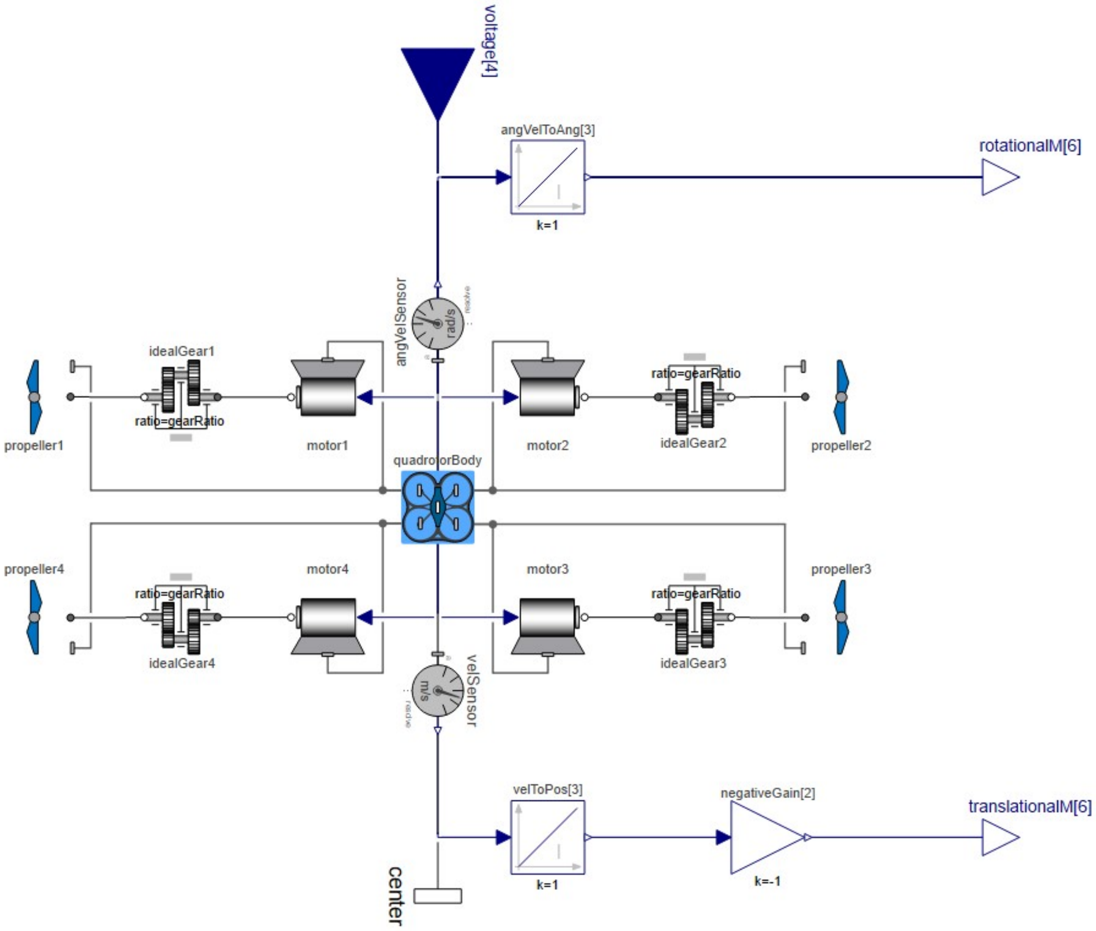


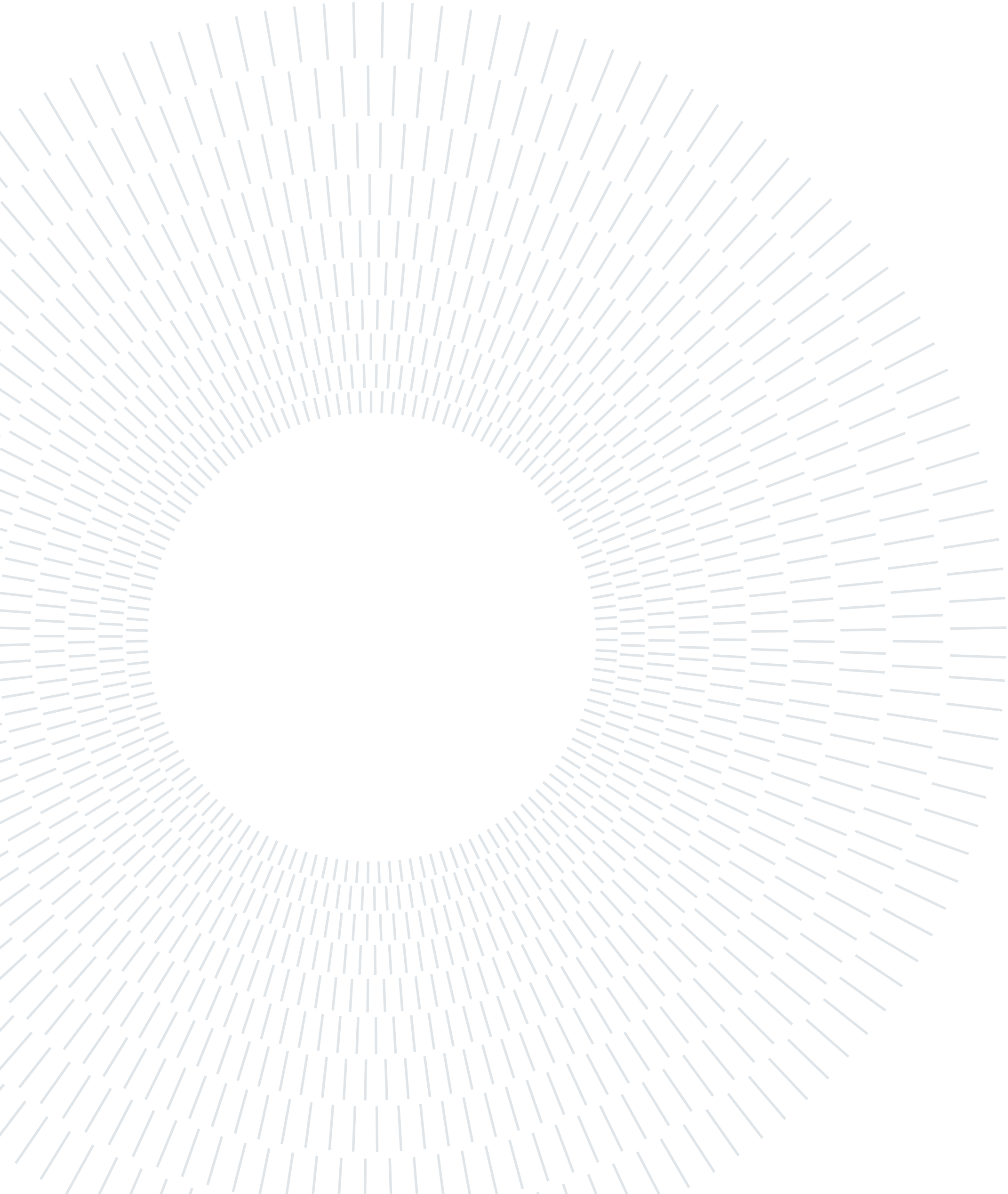
Figure 3.4: Quadrotor base model, including the mass body, propulsion system and sensors

As can be seen in the Figure 3.4, the propulsion system integration includes the electric motors connected to an ideal gearbox and followed by the propeller to transmit the forces and rotation speed. On the other hand, the forces, torques and rotation speed from DC motor and propeller connectors are mounted on the "quadrotorBody" component.

Regarding measurements, two sensors from Modelica standard library has been used for this model. The first one is the "AbsoluteVelocity" sensor, which is connected to the frame of the center of the "QuadrotorBody" and measures the linear velocity of the quadrotor in the Earth's reference frame, breaking it down into u, v, and w components corresponding to the world's x, y, and z directions. By integrating the data from this sensor, the absolute position vector in the world reference frame, consisting of the x, y, and z positions, can be determined. Note that negative gain is used to change the direction of the position and velocity in the z direction to be along with the world reference frame of the Aircraft library. All measurements pass to the output "TranslationalM" which is vector of [x, y, z, u, v, w].

The "absoluteAngularVelocity" sensor is connected to the frame of the center of the "QuadrotorBody" and records angular velocities in the quadrotor's body reference frame, denoted as p, q, and r corresponding to rotations around the x, y, and z axes. By applying an integrator, the quadrotor's rotational angles, roll, pitch and yaw can be derived. Taking into account the small angle assumption, these angles are equivalent to the Euler angles, ϕ, θ and ψ. All measurements pass to the output "RotatioalM" which is vector of [ϕ, θ, ψ, p, q, r].

To simplify the process of importing and testing quadrotor information, the key properties of the quadrotor component have been elevated to the higher level (for example the mass properties can be changed when user selects "QuadrotorBase" component). This makes it easier for users to identify the primary parameters of the model and facilitates the adjustment and tuning.



4 | Linear controller design on WSM

This chapter discusses a cascade position-attitude linear controller implementation for the developed quadrotor model on the wolfram system modeler. The initial part will outline the attitude controller algorithm and its implementation. Following that, the second section will delve into the examination and implementation of the position controller. At the end, the integration of both attitude and position controllers will be discussed.

Controlling a quadrotor is a challenging and intriguing task due to its six degrees of freedom, consisting of three rotational and three translational movements. What adds to the complexity is that quadcopters are under-actuated systems, with only four independent inputs available (rotor speeds). Consequently, achieving all six degrees of freedom involves coupling between rotational and translational motions.The quadrotor dynamics is highly nonlinear, and the complex aerodynamic effects further add to the challenge. Unlike ground vehicles, quadcopters have very little friction, making it difficult to prevent motion, and they require their own damping to remain stable. These factors make quadcopter control an interesting and complex problem. To tackle this problem, a simplified model of quadcopter dynamics is used, and controllers are designed to follow a designated trajectory.

4.1. Attitude controller

4.1.1. Methodology

Cascade PID control is often applied in controlling quadrotor attitude. It usually consists of two tiers of PID controllers: one to manage angular rates (like roll, pitch, and yaw rates), and another to handle desired angles (such as roll, pitch, and yaw angles). Assuming quadrotor operates in a regime where small angle approximations are valid, meaning the angular deviations from the desired attitudes are small, the transformation can be eliminated. [11]

The cascade control algorithm for roll channel control is as follows:

$$\begin{aligned} e_\phi &= \phi_m - \phi_{ref} \\ \dot{\phi}_{ref} &= k_{p_\phi} e_\phi \end{aligned} \tag{4.1}$$

Where e_ϕ is the roll angle error, ϕ_m is the measured phi angle, ϕ_{ref} is the reference roll angle and $\dot{\phi}_{ref}$ is the reference roll rate.

For angular velocity, the control law reads:

$$
\begin{gathered}
e_{\dot{\phi}} = \dot{\phi}_m - \dot{\phi}_{ref} \\
\tau_{\phi} = k_{p_{\dot{\phi}}} \left(e_{\dot{\phi}} + \frac{1}{T_i} \int_0^t e_{\dot{\phi}}(\tau) d\tau + T_d \frac{de(t)}{dt} \right)
\end{gathered}
\tag{4.2}
$$

Where $e_{\dot{\phi}}$ is the angular rate error, τ_{ϕ} is the roll control command equivalent to the roll moment increment. $\dot{\phi}_m$ is the measured angular rate, T_i is the integrator time constant and T_d is the derivative time constant The same applies to the pitch and yaw control channels. Figure 4.1 demonstrates the block diagram corresponds to the attitude controller.

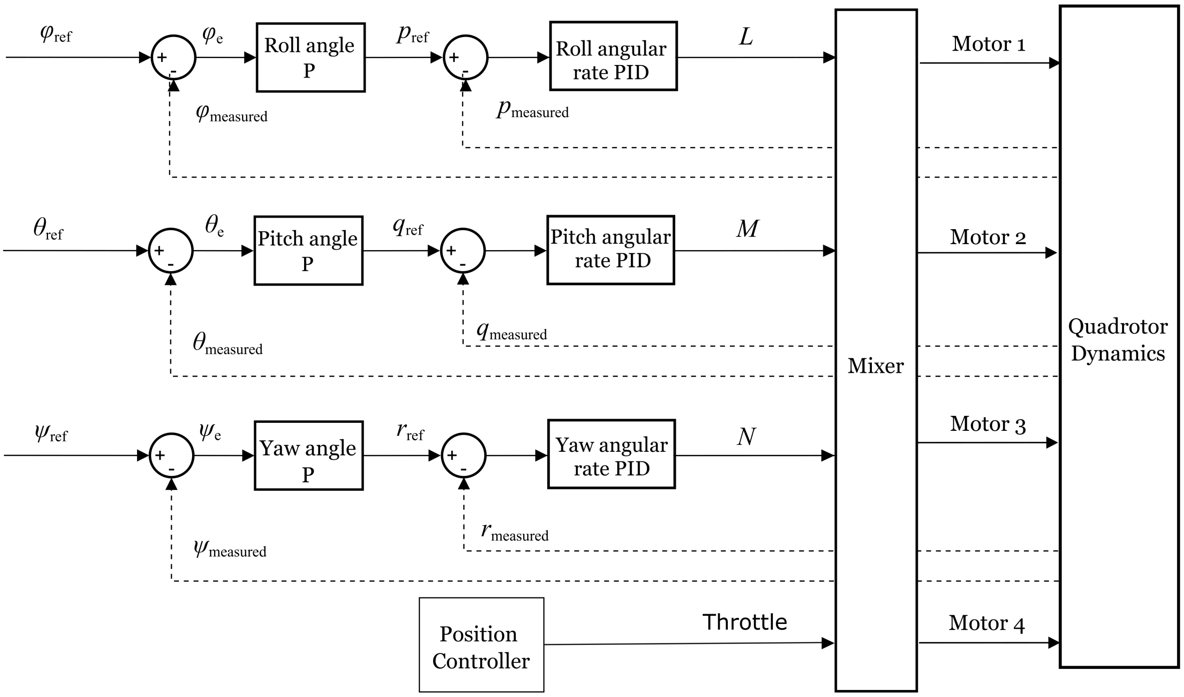


Figure 4.1: Linear attitude controller block diagram

4.1.2. Implementation

A cascade P-PID controller has been used for the linear attitude controller design. As can be interpreted from Figure 4.2, the input of the attitude controller block is the reference roll ("rollRef"), pitch ("pitchRef") and yaw ("yawRef") angles and the measured values of the angles and angular velocities "RotationM" input, coming from the rotational sensor measurement. And the output is the longitudinal, lateral and directional moment control command. The proportional (P) and PID controller blocks are also introduced from the Modelica standard library named "LimPID".

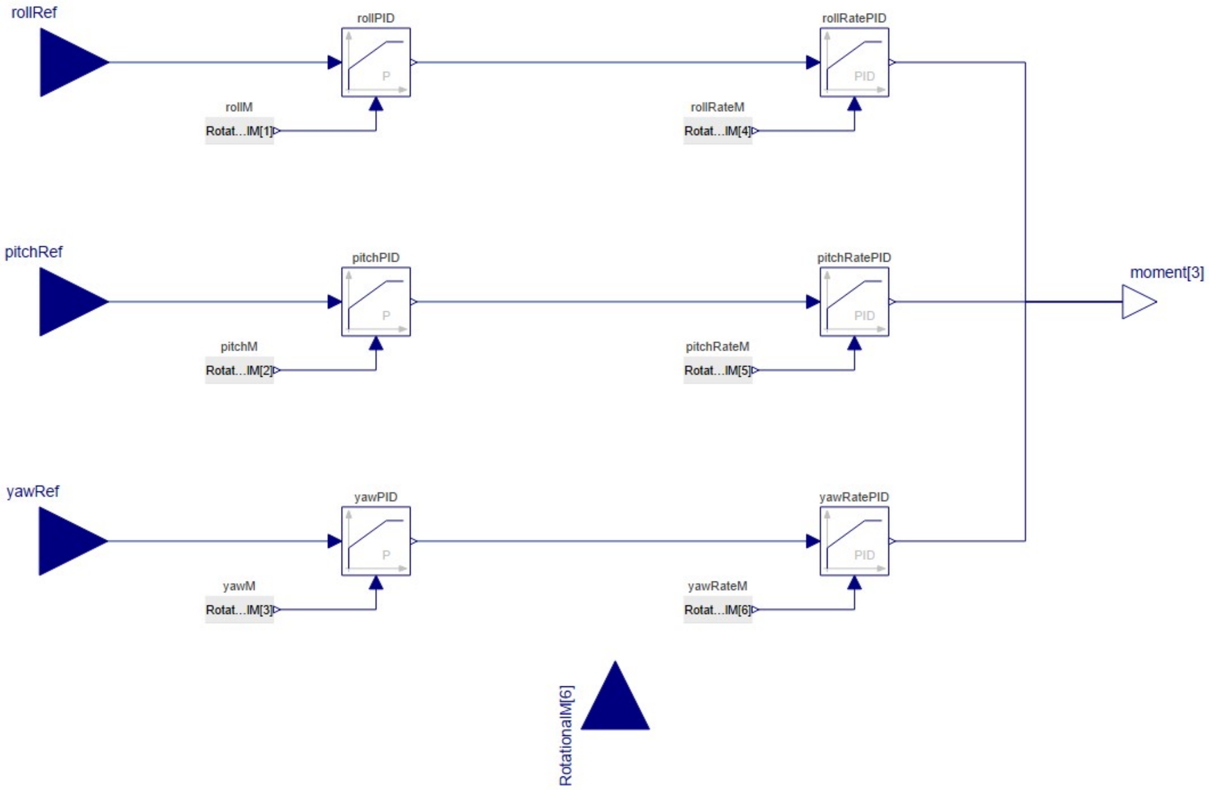


Figure 4.2: Linear attitude controller implementation on WSM

The "LimPID" component is presented in Figure 4.3 in which there are two inputs that are the set-point signal and measured signal and the output is the control command.

The "LimPID" controller can be adjusted by modifying its proportional gain, integrator time constant, derivative time constant, and saturation limit. The saturation limit can be configured within the constraints defined by "yMax" and "yMin", taking into account model limitations such as actuator deflection limits or motor RPM. Once the controller type in "LimPID" is selected, a combination of proportional, derivative, and integral controllers can be configured to meet the desired control objectives. Also, it gives user the opportunity to select the controller type from a drop-down menu, and also add the feed-forward action.

Therefore, the proportional LimPID has been used for the angle regulation and a PID version has been used for the angular velocity control. The control law in "LimPID" can be written as:

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right) \tag{4.3}$$

where K_p is the proportional gain of the PID, T_i is the integrator time constant, and T_d is the derivative time constant.

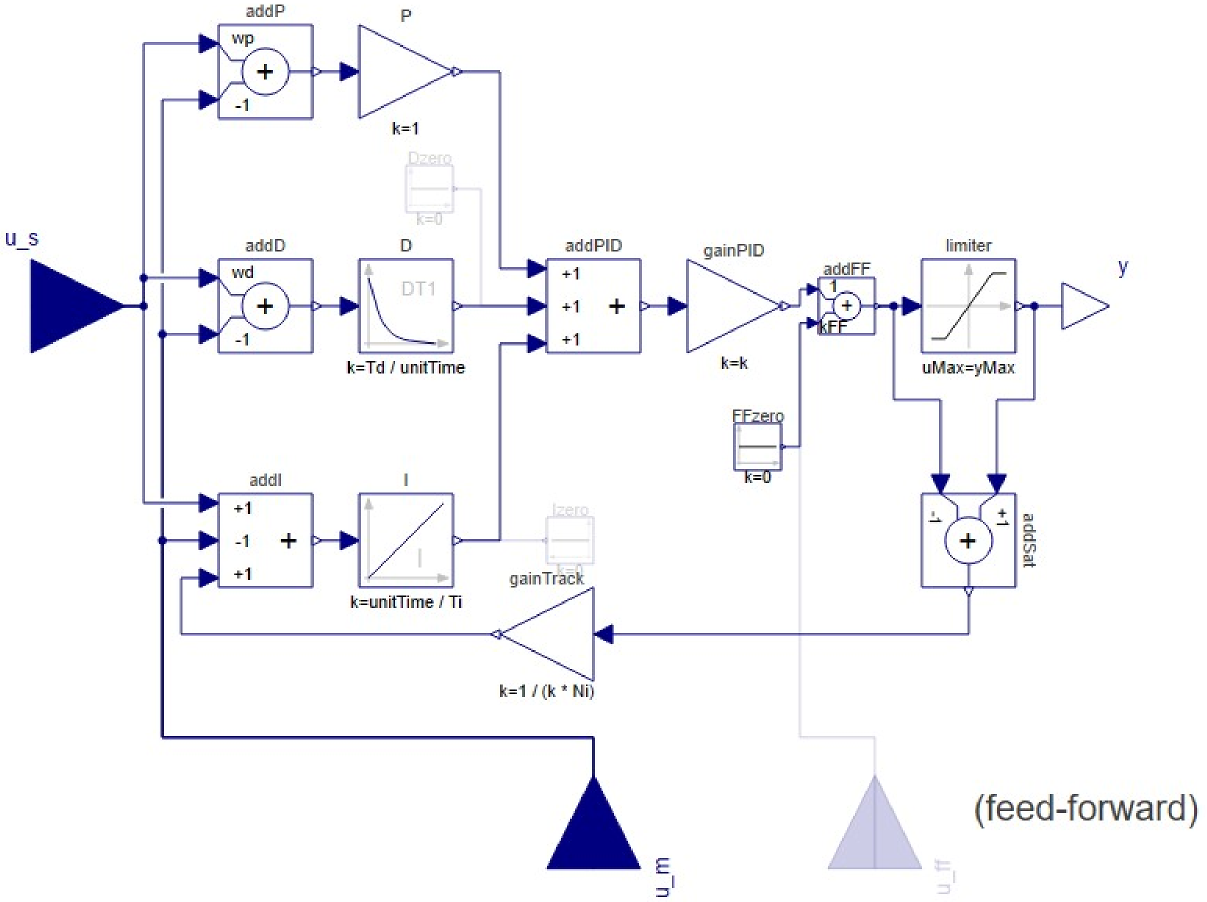


Figure 4.3: LimPID component of the Modelica standard library including: P, PI, PD, and PID controller with limited output, anti-windup compensation, setpoint weighting and optional feed-forward

The values of the attitude controller gains are summarized in the Table 4.1

Gain	Value	Gain	Value
k_{p_ϕ}	10	T_{i_p}	20
k_{p_θ}	10	T_{i_q}	20
k_{p_ψ}	10	T_{i_r}	10
k_{p_p}	10	T_{d_p}	0.01
k_{p_q}	10	T_{d_q}	0.01
k_{p_r}	10	T_{d_r}	0.01

Table 4.1: Linear attitude controller gains of the WSM model

4.2. Position controller

4.2.1. Methodology

For the vertical direction a PID controller is implemented to generate the proper throttle command. The control law can be obtained as follows:

$$\begin{aligned} e_z &= z_m - z_{ref} \\ throttle &= k_{p_z}\left(e_z + \frac{1}{T_i}\int_0^t e_z(\tau)d\tau + T_d\frac{de(t)}{dt}\right) + F_{base} \end{aligned} \tag{4.4}$$

Where F_{base} is the required minimum force to overcome the wight of the drone (F_{base} = wight of the drone).

The same methodology as the attitude controllers has been used for the north and east channel of the position controller. A cascade P-PID controller has been implemented for longitudinal and lateral control in which the inner loop (velocities) have been regulated with PID controllers and a proportional gain is used for the outer loop (position) control. Equation (4.5) presents proportional north position control law:

$$\begin{aligned} e_x &= x_m - x_{ref} \\ \dot{x} = u_{ref} &= K_{p_x} e_x \end{aligned} \tag{4.5}$$

Where e_x is the x position error, x_m is the measured x position and x_{ref} is the desired x position. The control law corresponds to the north velocity of the quadrotor reads:

$$\begin{aligned} e_u &= u_m - u_{ref} \\ \theta_{ref} &= k_{p_u}\left(e_u + \frac{1}{T_i}\int_0^t e_u(\tau)d\tau + T_d\frac{de(t)}{dt}\right) \end{aligned} \tag{4.6}$$

Where e_u is the north velocity error, u_m is the measured north velocity and u_{ref} is the reference north velocity. The same approach is implemented for the east position and velocity regulation and generate set-point roll angle. The position control structure has been shown in Figure 4.4:

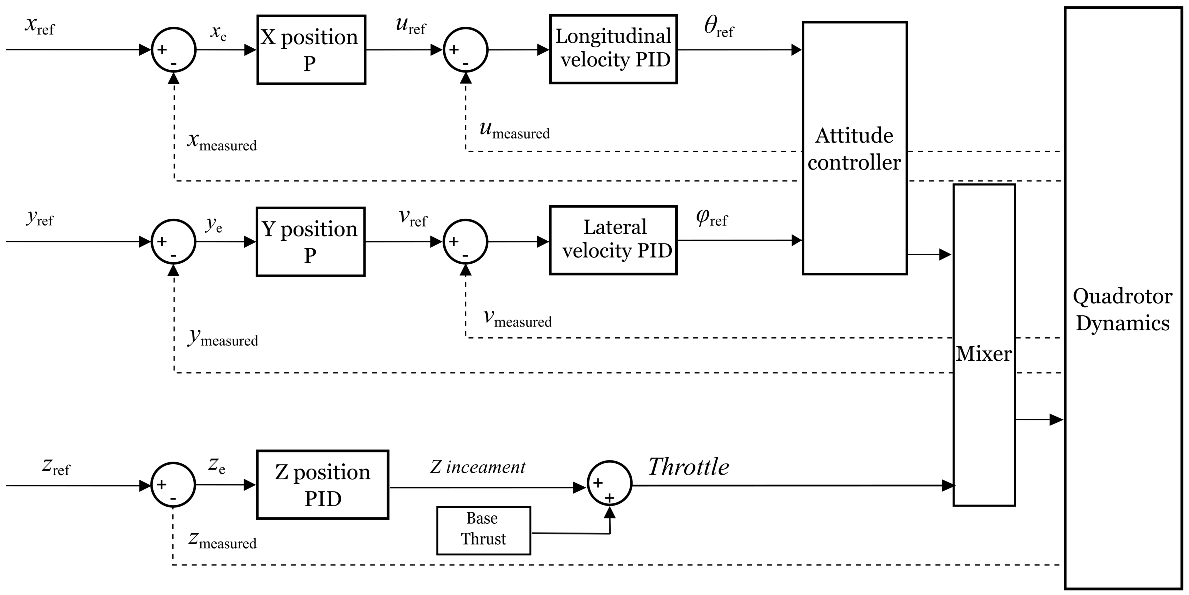


Figure 4.4: Linear position controller block diagram

4.2.2. Implementation

The translational controller receives user-defined reference positions as inputs and utilizes feedback of the measured translational position and velocity to control the translational dynamics of the quadrotor. Additionally, the position control generates reference rotation angles, roll (ϕ) and (θ), which are subsequently will be used in the attitude controller.

Note that the linear position is measured in the inertial frame, therefore the inputs from the sensors are relative to the inertia frame.

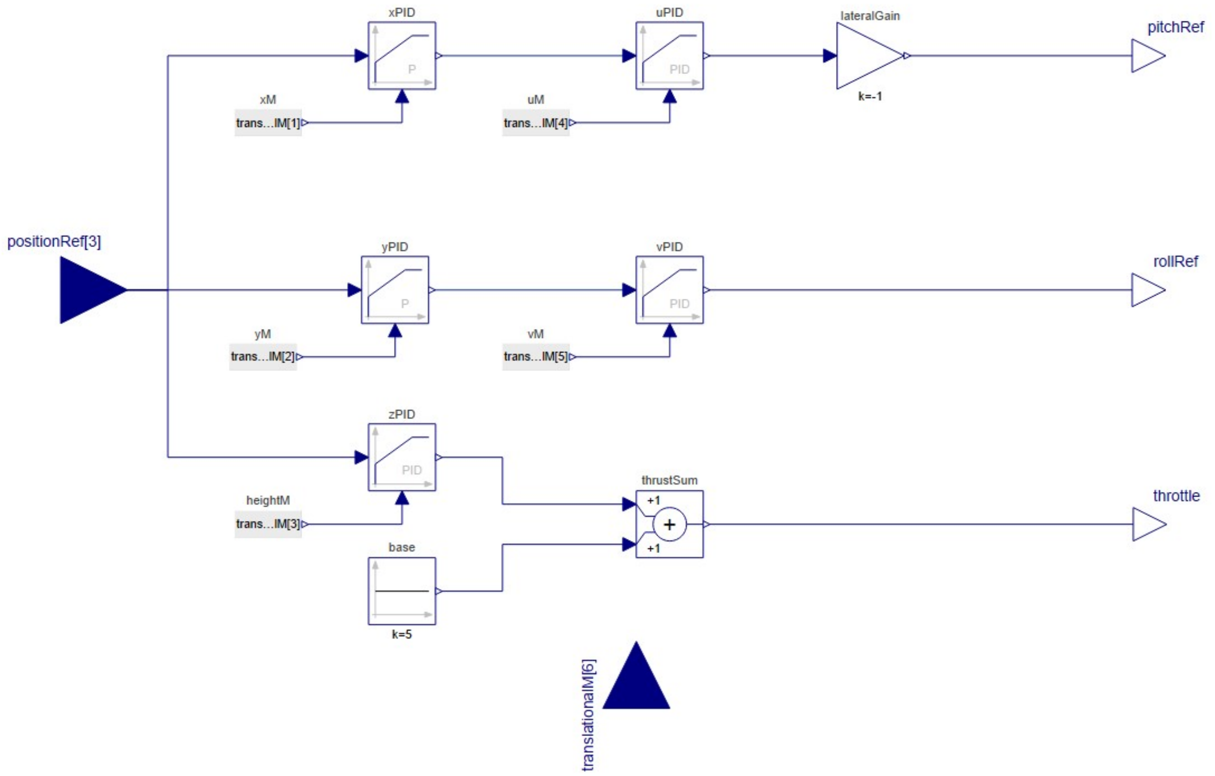


Figure 4.5: Linear position controller implementation on WSM

Figure 4.5 demonstrates the block diagram of the position controller. It employs a cascade P-PID controller for both longitudinal and lateral control channels, along with a PID controller for height control.

In-depth, in the longitudinal control, the reference x position is determined by user input, and it is subtracted from the quadrotor's measured x position "xMeasured" from the sensor to calculate the x position error (x_e). This error is then passed to a proportional (P) controller. The output of the x position controller is the reference longitudinal velocity, denoted as "uRef".
Utilizing the measured longitudinal velocity "uMeasured", the velocity error (v_e) is feed into the LimPID controller, generating the reference pitch angle (θ_{ref}) for input of the attitude controller.
The same approach is applied to the lateral channel control for controlling the quadrotor's lateral position.
The quadrotor's vertical position (z) is regulated by a LimPID controller, which produces the throttle increment command for altitude adjustment. The base thrust which represents the minimum required thrust to overcome the weight of the quadrotor should be added to the thrust increment.This net command is then sent to "Mixer", where the allocation of forces and moments takes place, enabling the generation of the appropriate motion.

Table 4.2: Linear attitude controller gains of the WSM model

Gain	Value	Gain	Value
k_{p_x}	3	k_{p_u}	1
k_{p_y}	3	k_{p_v}	1
k_{p_z}	10	T_{i_u}	200
T_{i_z}	2	T_{i_v}	200
T_{d_z}	0.1	T_{d_u}	0.001
		T_{d_v}	0.001

4.3. Controller integration

In the previous section the structure of the attitude and position controllers presented separately. In this chapter the full integrated controller will be presented. The force and moment allocation of the quadrotor controller should be determined at this point: Figure 4.6 depicts the mixer for force and moments allocation.

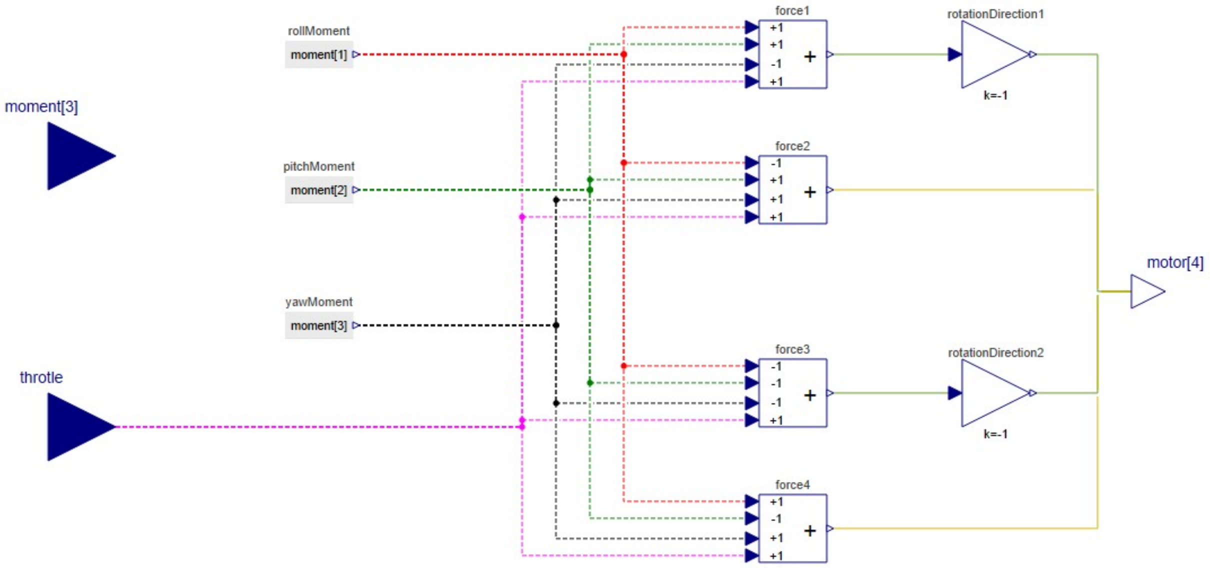
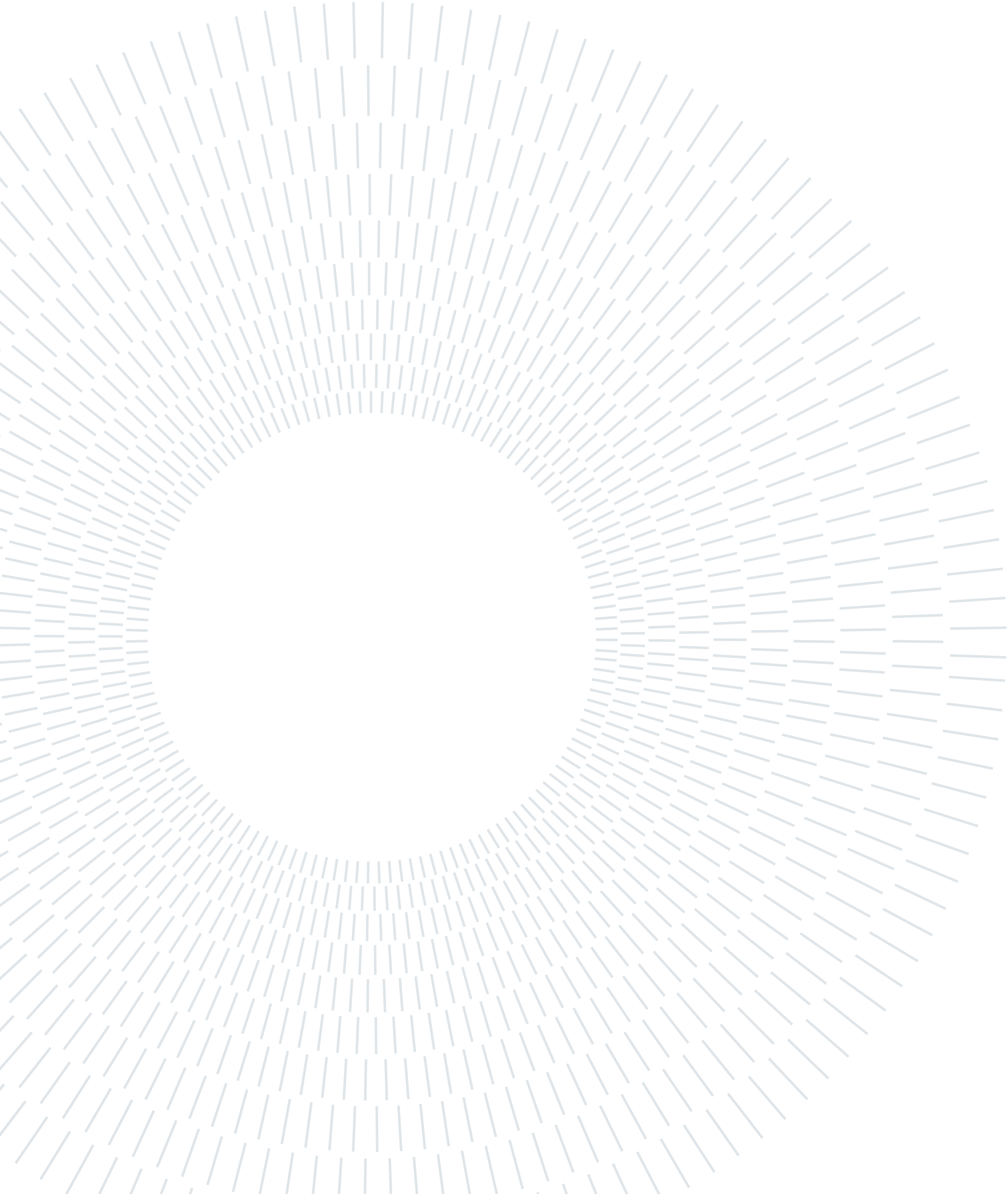


Figure 4.6: Mixer for force and moment allocation

The control allocation matrix can be summarized below.

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \\ f_{th} \end{bmatrix} \tag{4.7}$$

The conversion gains are responsible for altering the rotation direction of the diagonal pairs of the quadrotor, where propellers 1 and 3 (the green pair) rotate clockwise, while propellers 2 and 4 (the yellow pair) rotate counter-clockwise as demonstrated in Figure 2.2.



5 | Examples and results

In this section an integrated model of the quadrotor will be presented through two examples and the simulation result will be discussed.

5.1. Example 1: Custom trajectory tracking

The objective of this example is to ensure that the quadrotor follows a specified trajectory commanded by a position vector and a yaw angle. Figure 5.1 demonstrates the diagram view of the full quadrotor model and the reference inputs.

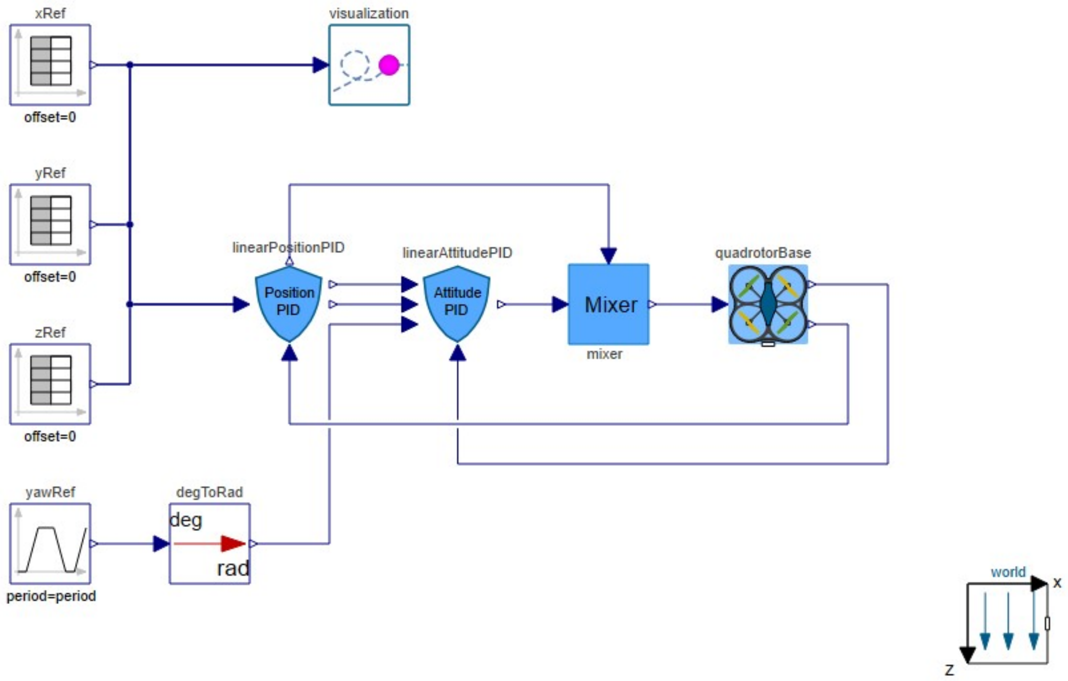


Figure 5.1: Full quadrotor example

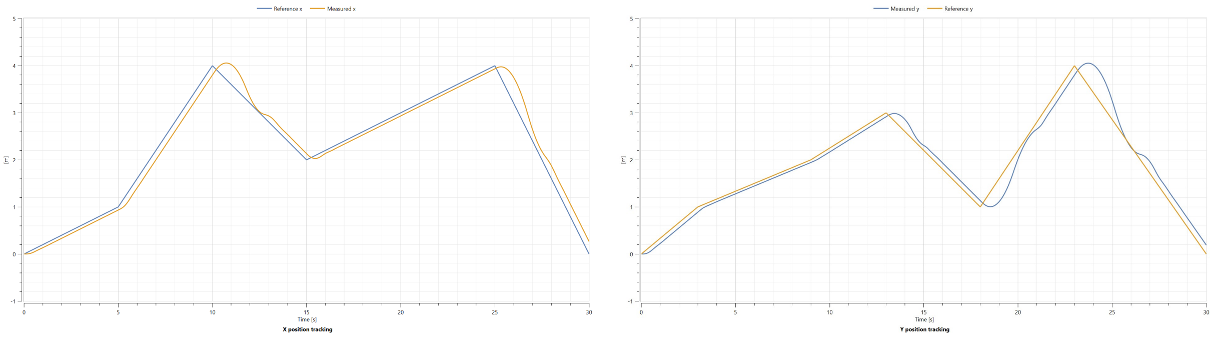
As can be interpreted in Figure 5.1, the quadrotor is connected to the mixer and then to an attitude and position controllers sequentially. The position controller block receives reference position inputs from the "timeTable" blocks (xRef, yRef, zRef) which is in the Modelica standard library, while The yaw angle, specified in degrees, is used to command the yaw signal of the attitude controller block and is

subsequently converted to radians to ensure consistency with the rest of the computations.
The model now includes a global reference frame using the north-east-down (NED) convention for aircraft library, along with the addition of a visualization block for animation purposes to illustrate the reference trajectory path.

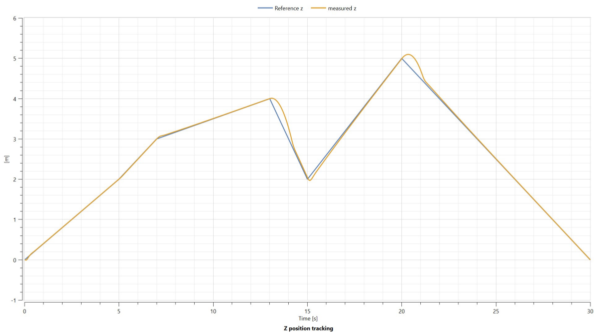
The example is designed to be easily understood and adjusted by users, even those who may not be fully familiar with quadrotor dynamics. To facilitate this, the most important parameters that may need tuning or modification are clearly listed in the parameter tab.

5.1.1. Result

The simulation results for positions are listed in Figure 5.2.



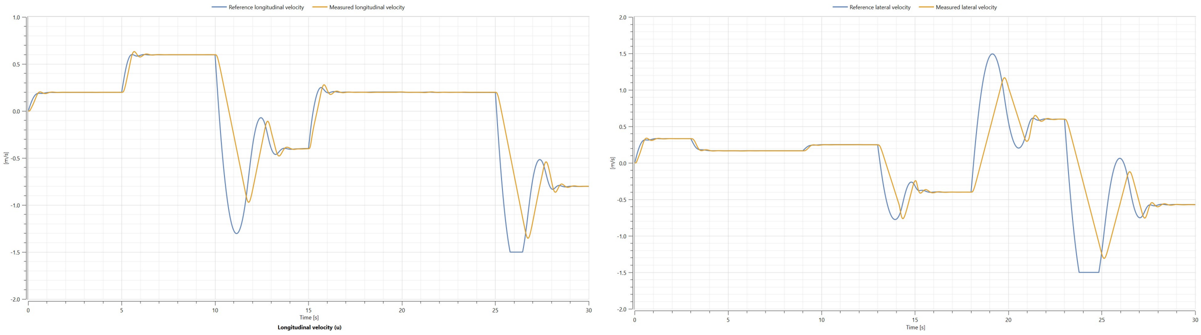
(a) X position tracking (b) Y position tracking



(c) Z position tracking

Figure 5.2: Result of position tracking of the quadrotor

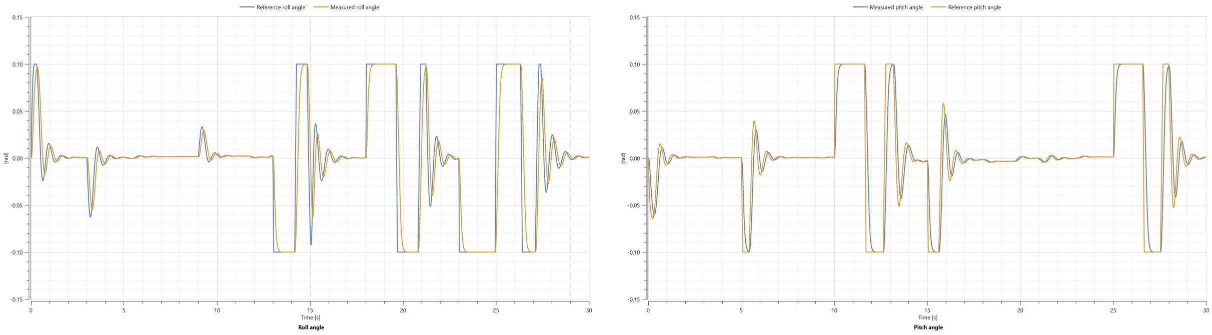
The result shows that the quadrotor is able to track the reference trajectory closely. As anticipated, the quadrotor follows the path more accurately when there are no sudden changes, but when faced with abrupt changes, it exhibits higher overshoot. However, it eventually realigns itself with the trajectory.
Below the longitudinal and lateral velocity tracking is demonstrated which shows the fairly good tracing of the velocity, yet the same behaviour as the position tracking in the abrupt changes 5.2.



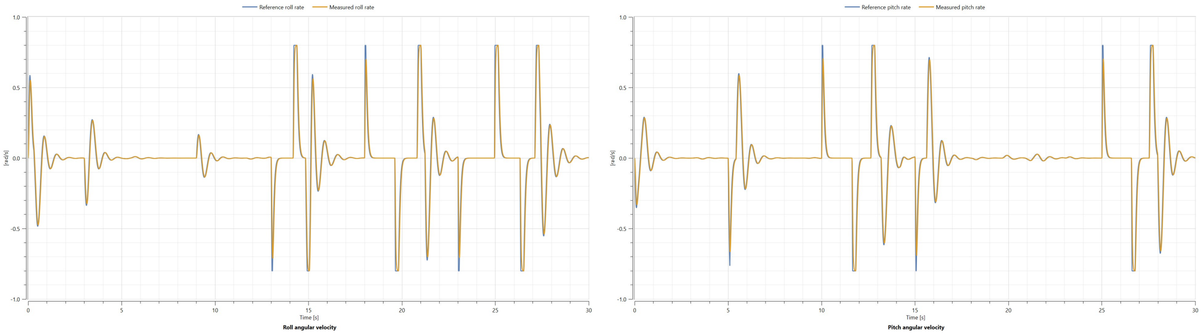
(a) u velocity tracking (b) v velocity tracking

Figure 5.3: Result of velocity tracking of quadrotor

The attitude angles and angular rates are summarized in Figure 5.4, which tracks the set-point precisely.



(a) Roll angle tracking (b) Pitch angle tracking



(c) Roll angular velocity tracking (d) Pitch angular velocity tracking

Figure 5.4: Result of angles and angular rates tracking of quadrotor

5.2. Example 2: Package delivery

In today's rapidly evolving society, the quadrotor's role in package delivery is more crucial than ever. For surveillance and medicine delivery to the online shopping, there is an increasing need for prompt and efficient delivery services. Quadrotors, equipped with the capability to navigate urban landscapes and deliver packages with precision, are essential in fulfilling this demand. They are instrumental in reaching inaccessible or remote areas where traditional delivery methods may struggle. Moreover, quadrotors contribute to reducing the environmental impact associated with conventional delivery systems, thus making them indispensable in modern logistics.

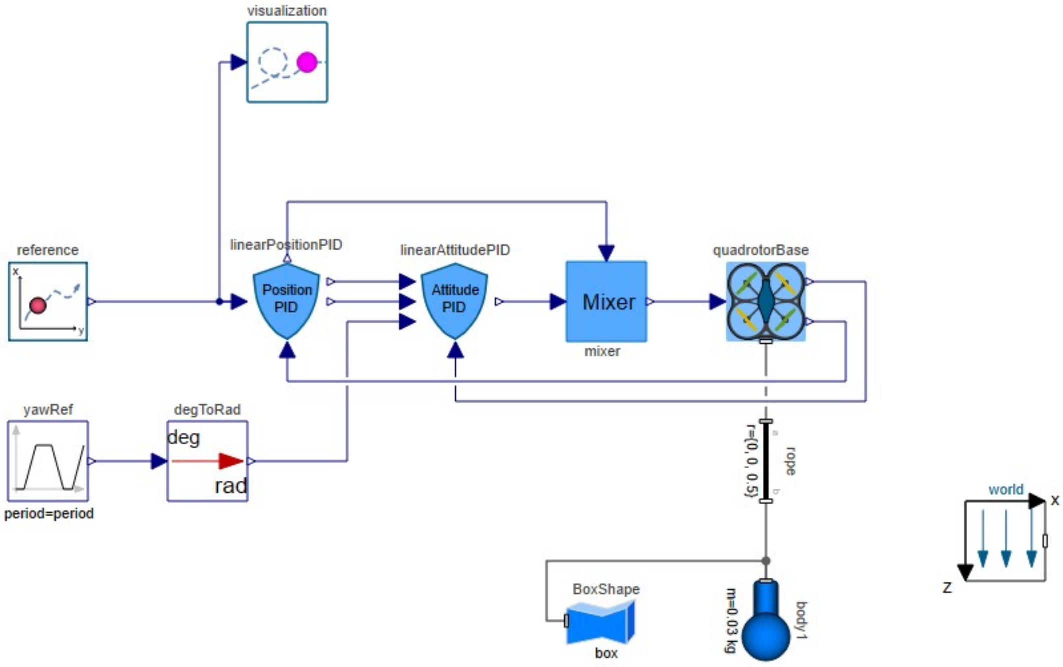


Figure 5.5: Quadrotor package delivery example

Figure 5.5 demonstrates the package delivery mission example in the context of a circular trajectory motion to also introduce the "ReferenceTrajectories" pre-planned path component for defining the reference position B.1. Two trajectory types can be selected and modified within the reference trajectory, where amplitude and period determine the maximum distance from the origin and the time required for a complete path (so higher period means slower path generation), respectively. The inputs are connected to the controller, and the controller command is then connected to the quadrotor body. While the default controller values work adequately for both (circular and infinity) trajectories, achieving more precise path tracking entails tuning the PID controllers for x and y positions for each trajectory type.
Obviously, when the frequency of the path generation is slower, the quadrotor can trace the path better. In relation to the package, there is a mass object attached to the center of the quadrotor via a fixed-translation component (referred as a "rope" in the example) within the Modelica standard library's Multibody package. Additionally, a "fixedShape" element has been introduced for visualization in the animation center.

5.2.1. Results

Figure 5.6 illustrates the position trajectory of the quadrotor plotted in the x-y plane. It's evident that there's a slight overshoot at the onset of the circular motion, but this discrepancy diminishes over time. The accuracy of the model is relies on the frequency of trajectory generation, which is influences the quadrotor's speed. As the desired speed passes a certain threshold, the quadrotor may deviate from the desired path. Moreover, this example serves to determine the maximum allowable weight for the package.

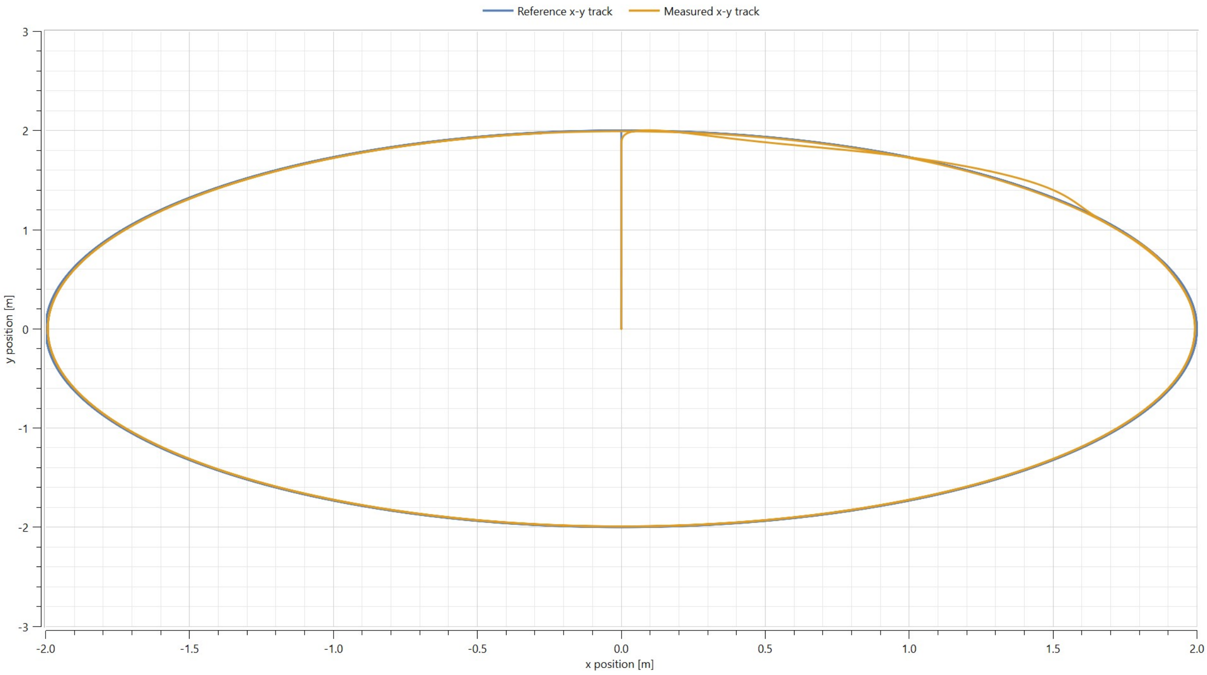
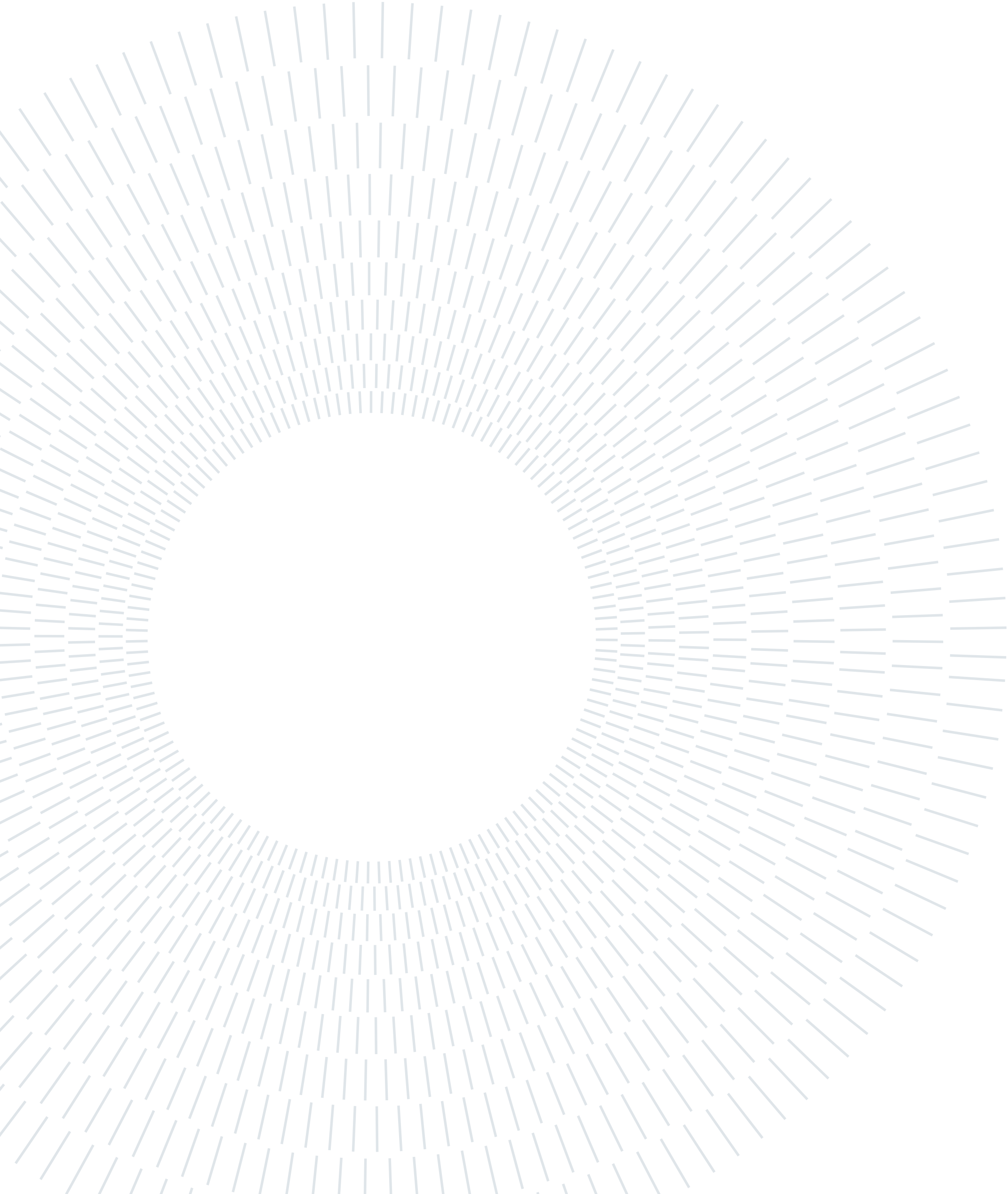


Figure 5.6: Quadrotor package delivery result: the x-z trace

5.3. Discussion

While the linear cascade P-PID controller effectively handles smooth paths and small angle changes, it restricts the quadrotor's operational capabilities. To achieve better performance and enable the quadrotor to navigate more challenging paths, a non-linear controller needs to be developed. Additionally, for a more realistic model of the quadrotor, simplifications in modeling should be minimized, requiring more precise data and specifications of the components involved.

The former problem will be addressed in next chapter by introducing the incremental non-linear dynamic inversion. The later will be discussed here. A more detailed model for a multirotor drone has been developed for a costumer based on the introduced simple mode. In the advanced model, hcomprehensive component data sheets are utilized, integrating a power source like a battery model alongside the electric motor. Additionally, a noise component is introduced into the model, and a more detailed model is employed for the brushless DC motor and propeller [5]. Which unfortunately can not be presented in detail. The short introduction and animation of the flight of the rotor can be presented can be found in the wolfram virtual conference [2].



6 | Incremental non-linear dynamic inversion

The first section of this chapter introduces the concept and mathematical representation of the of the classical nonlinear dynamic inversion and incremental nonlinear dynamic inversion. The subsequent section includes the implementation of the INDI method on MATLAB and Simulink and in the last section the results will be discussed.

6.1. Nonlinear dynamic inversion basis

Nonlinear Dynamic Inversion (NDI) is a control approach developed in the late 1970s to improve the limitations of standard linear methods. The basic idea of NDI is to eliminate the non-linearity in a system, making it behave more predictably like a linear system. This allows traditional linear control techniques to be used to achieve the desired behavior in the system. Therefore, conventional linear control techniques can be applied so that a desired dynamics is achieved for the closed loop system [6].
It's particularly useful when dealing with complex systems that don't adhere to linear relationships between inputs and outputs, which are common in real-world scenarios such as aerospace, robotics, and automotive systems.
To get the better understanding of the NDI concept lets delve into the mathematical representation of the NDI method. Consider a general nonlinear system of order n described by the following state space representation:

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ y &= h(x) \end{aligned} \tag{6.1}$$

Let's consider a single input single output system (SISO), the number of inputs is denoted by $m = 1$, and the order of the system is n. In this case, $f(x)$ and $g(x)$ represent vector fields within the spaces of $\mathbb{R}^n$ and $\mathbb{R}$ respectively and $h(x)$ is a nonlinear function from $\mathbb{R}^n$ and $\mathbb{R}$.
The objective is to determine a control law such that the closed-loop system follows to desired dynamics.

The NDI approach is based on sequentially deriving the output over time until an explicit relation of u, is achieved. With every derivative taken, a new state vector is created, and the final state vector's derivative is related to a nonlinear equation, known as the virtual control [7].
To begin with the derivation of the control law, the first step is to take the derivative of the output [1]:

$$\dot{y} = \frac{dy}{dt} = \frac{\partial h(x)}{\partial x}\frac{dx}{dt} = \nabla h(x)\dot{x} = \nabla h(x)[f(x) + g(x)u] \tag{6.2}$$

Where $\nabla h(x)$ represents the gradient operator of the scalar function $h(x)$ with respect to the state vector x:

$$\frac{\partial h(x)}{\partial x_i} \qquad x_i = 1, 2, ..., n \tag{6.3}$$

Before proceeding with further derivation of the control law, let's introduce the concept of the Lie derivative. For a scalar function $h(x)$ and a vector field $f(x)$, we define a new scalar function $L_f h$, known as the Lie derivative (or simply, the derivative) of h with respect to f. $L_f h$ represents the directional derivative of h along the vector f. Thus, in this context [7]:

$$L_f h(x) = \nabla h(x) f(x) \tag{6.4}$$

We can recursively define repeated Lie derivatives as follows:

$$\begin{aligned} L_f^0 h(x) &= h(x) \\ L_f^i h(x) = L_f[L_f^{i-1} h(x)] = \nabla(L_f^{i-1} h(x)) f(x) \qquad & i = 1, 2, ... \end{aligned} \tag{6.5}$$

Likewise, if g is another vector field:

$$L_g L_f h(x) = \nabla(L_f h(x)) g(x) \tag{6.6}$$

Now, we can continue with the rest of the derivation of the control law. Using equation (6.4), we can rewrite equation (6.2) as

$$\dot{y} = L_f h(x) + L_g h(x) u \tag{6.7}$$

If ($L_g h(x) \neq 0$), rearranging equation (6.7) and introducing the virtual control ($\nu = \dot{y}$), the control law result in:

$$u = L_g h(x)^{-1}[\nu - L_f h(x)] \tag{6.8}$$

Otherwise, If $L_g h(x) = 0$, indicating there is no explicit relation between u and y, another differentiation of y is required. To achieve this, utilizing equations (6.6) and (6.7), the second derivative of y is as follows:

$$\ddot{y} = L_f^2 h(x) L_g L_f h(x) u \tag{6.9}$$

Then, the control law will be:

$$u = L_g L_f h(x)^{-1}[\nu - L_f^2 h(x)] \tag{6.10}$$

The derivation continues until finding a number r such that $L_g L_f h(x) \neq 0$, indicating an explicit relation between u and y is obtained.
The general rth order differentiated output can be summarized as:

$$y^{(r)} = L_f^r h(x) + L_g L_f^{r-1} h(x) u \tag{6.11}$$

Following that, the generalized control input reads:

$$u = L_g L_f^{r-1} h(x)^{-1}[\nu - L_f^r h(x)] \tag{6.12}$$

Equation (6.12) demonstrate the relation between the virtual control and the output o the linearized system, corresponding to a cascade of r integrators (i.e., $\nu = y^{(r)}$) to achieve an explicit dependence between input and output.

To exemplify, let's consider a system in which the number of inputs is denoted by m, and a system of order n. In this case, $f(x)$ and $h(x)$ represent vector fields within the spaces of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. In this context G is a control effectiveness matrix of the order $n \times m$ [6].

$$\begin{aligned} \dot{x} &= f(x) + G(x)u \\ y &= h(x) \end{aligned} \tag{6.13}$$

Imagine $h(x) = x$, therefore, within the first time derivative of the y an explicit dependence of u is found.

$$\dot{y} = \dot{x} = f(x) + G(x)u \tag{6.14}$$

The linear relationship for the virtual input can be expressed as follows:

$$\nu = \dot{y} \tag{6.15}$$

By substituting equation (6.15) into equation (6.14) and rearranging based on the variable u, the objective is obtained and the control law is as follows:

$$u = G(x)^{-1}[\nu - f(x)] \tag{6.16}$$

Note that the $G(x)$ assumed to be non zero ($G(x) \neq 0$).

Nonlinear Dynamic Inversion (NDI) presents numerous benefits such as enhanced performance compared to linear control techniques in nonlinear systems and the capability to manage complex dynamics. Nevertheless, designing NDI controllers can be challenging due to the need for precise system dynamics ($f(x)$ and $G(x)$ in this case) and failing in providing the exact dynamics leads model to diverges significantly. To cope with this high dependency to the exact model of the system which is most of the time impossible to be known, the Incremental version of the NDI is proposed.

6.2. Incremental non-linear dynamic inversion basis

As mention in the last section, to over come the dependency of the NDI control method to the exact model of the system, and increase the robustness of the system, Incremental Non-linear Dynamic Inversion (INDI) method has been introduced. So that instead of computing the whole control input, the required increment of the control action in a specific moment, relative to the system conditions just before that instant in time is computed.

To begin with the mathematical representation of the INDI, we can expand the first order Taylor series of the equation (6.1) around the aforementioned instant of time (specified by 0 subscript).

$$x \approx \dot{x}_0 + \frac{d(f(x) + G(x)u)_{x_0,u_0}}{dx}(x - x_0) + G(x_0)(u - u_0) \tag{6.17}$$

If we assume a high frequency of the sample rate of the system and the input change be relatively much faster than the change of the state, the change in state can be equal to zero $(x - x_0 = 0)$ so the equation (6.17) can be reduced to:

$$x \approx \dot{x}_0 + G(x_0)(u - u_0) \tag{6.18}$$

Using the same virtual control input notation ν, and considering $(G(x) \neq 0)$, the INDI control law can be obtained as follows:

$$u = G^{-1}(x_0)(\nu - \dot{x}_0) + u_0 \tag{6.19}$$

As can be interpreted form the INDI control law and compare it to the NDI control law (6.16), the INDI control law does not include the $f(x)$ function. In other words, it eliminates the dependency of the controller on the system states, which stands as a significant advantage of the INDI method over the NDI method. To address the absence of state information, it becomes necessary to measure both the control action signal at a later time (u_0) and the derivative of the state at that later time $(\dot{x}_0)$. The limitation for this method is that sometimes the state derivative are not measurable so an effective way of the estimation should be considered. Even if the measurement is available, one must deal with factors such as sensor noise, delay, and bias, which required careful consideration. The accuracy of the estimation and measurements of the (u_0) and $(\dot{x}_0)$ variables relies on the accuracy of the sensors and the filtering method.

Following comprehension of the theoretical framework and mathematical depiction of the INDI, we can proceed to its application in modeling the dynamics of a quadrotor.

Applying the INDI Method to Quadrotor Dynamics

First let's go through with derivation of the control law for attitude controller the rotational dynamics of the quadrotor that can be summarized as [10]:

$$\dot{q} = \frac{1}{2} q \otimes \omega_b \tag{6.20}$$

$$\dot{\omega}_b = J^{-1}(M + M_{ext} - \omega_b \times J\omega_b) \tag{6.21}$$

$$y = \eta \tag{6.22}$$

Where y is the output vector, $\eta = [\phi, \theta, \psi]$ is the Euler angle vector, $\omega_b = [p, q, r]$, is the angular velocity vector in body reference frame and J is the moment of Inertia vector of the quadrotor. The moment vector generated by the rotors is denoted as M, and the external disturbance moment is denoted as M_{ext}. Force and moments of the rotors can be modeled as a quadratic relation of the rotor rotational speed.

$$T_i = k_\tau \Omega_i^2 \tag{6.23}$$

$$Q_i = k_q \Omega_i^2 \tag{6.24}$$

where Ω_i is the rotation speed of each rotor, k_τ the rotor thrust coefficient, k_q the rotor torque coefficient.

The moment generated by the rotors can be calculated through equation below:

$$\begin{aligned} M &= \begin{bmatrix} l_y(T_1 - T_2 - T_3 + T_4) \\ l_x(T_1 + T_2 - T_3 - T_4) \\ -Q_1 + Q_2 - Q_3 + Q_4 + I_r(-\dot{\Omega}_1 + \dot{\Omega}_2 - \dot{\Omega}_3 + \dot{\Omega}_4) \end{bmatrix} \\ &= \begin{bmatrix} l_y k_\tau & -l_y k_\tau & -l_y k_\tau & l_y k_\tau \\ l_x k_\tau & l_x k_\tau & -l_x k_\tau & -l_x k_\tau \\ -k_q & k_q & -k_q & k_q \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ I_r & I_r & I_r & I_r \end{bmatrix} \begin{bmatrix} \dot{\Omega}_1 \\ \dot{\Omega}_2 \\ \dot{\Omega}_3 \\ \dot{\Omega}_4 \end{bmatrix} \end{aligned} \tag{6.25}$$

$$M = G_1 \overset{\circ}{\Omega} + G_2 \dot{\Omega} \tag{6.26}$$

Where G_1 and G_2 are control effectiveness matrices, $\overset{\circ}{\Omega}$ is the squared of the angular speed vector. Usually the I_r which is the propeller rotor moment of inertia can be neglected due to being relatively small with respect to the inertia of the quadrotor. Therefore, the equation (6.26) can be reduced to the following equation:

$$M = G \overset{\circ}{\Omega} \tag{6.27}$$

Where $(G = G_1)$, the control effectiveness matrix which has been mentioned in the previous sections. Considering the control law derived in the last section, and having regard that now the state vector is the control output $(x = y)$ and $(\nu = \dot{x})$ the control inputs can be achieved following below equations:

$$\Delta u = G^{-1}(x_0)(\dot{y} - \dot{y}_0) \tag{6.28}$$

$$u = u_0 + \Delta u \tag{6.29}$$

Here, $\dot{y}$ represents the derivative of the output at the current instant, $\dot{y}_0$ denotes the output derivative just prior to this moment, and $G^{-1}(x_0)$ refers to the inverse of the current control effectiveness matrix.

6.3. Implementation of the INDI Approach

To start with the implementation, the obtained control law as mentioned in equation (6.19) should be followed in the generation of the block diagram. A combination of a P-P-INDI has been used for the attitude controller. Therefore, no integral action has been used in the model which expects ta faster response.

Figure 6.1 shows the block diagram of the attitude controller of the quadrotor. The attitude controller comprises of the three sections, the first section from left is the attitude controller to generate the angular velocity of the quadrotor from attitude (quaternions) and attitude set-points which has a proportional gain inside the "attitudeRegulator" block. Considering the quaternion kinematic equation and having regard that the quaternion errors is quaternion product:

$$q_e = q_0 \otimes q \tag{6.30}$$

Where q_0 represents the set-point quaternion, q denotes the measured quaternion, and q_e indicates the error between these two vectors. The attitude regulator can be written as follows [4]:

$$\omega_0 = 2K_q S(q)\dot{q} \tag{6.31}$$

Note that from now on, the angular velocity is in the body frame, we will drop the subscript b from ω_b and simply refer to it as ω. Therefore, ω_0 is the set-point angular velocity, K_q is the proportional gain vector, S is the skew-symmetric matrix derived from the quaternion q.
Regarding the angular velocity regulation, a proportional controller has been implemented for roll rate, pitch rate and yaw rate of the angular velocity vector.

$$\omega_c = K_\omega(\omega_m - \omega_0) \tag{6.32}$$

Where k_ω is the proportional gain of the angular velocity controller, ω_m is the measured angular velocity and the ω_0 is the set-point angular velocity and ω_c is the commanded angular speed.
Finally, the INDI approach has been used for the angular acceleration control to generate desired moment command for the quadrotor actuators. The control law follows equation (6.33):

$$M_c = M_f + J(\dot{\omega}_0 - \dot{\omega}_f) \tag{6.33}$$

Where M_c is the commanded moment to the actuator, M_f is the filtered moment, J is the Inertia tensor of the quadrotor:

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{6.34}$$

Where ω_0 is the set-point angular acceleration error and ω_f is the filtered angular acceleration of the quadrotor. Note that the angular acceleration and moments are in body reference frame.

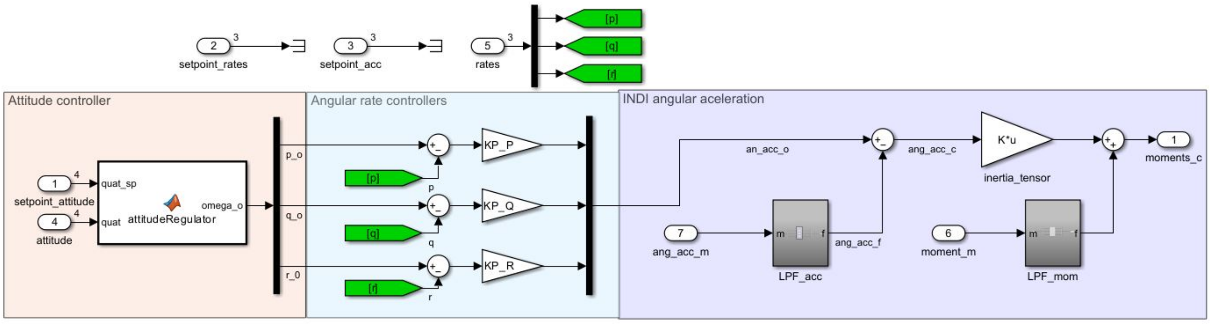


Figure 6.1: Simulink implementation of the attitude controller

As depicted in Figure 6.1, the two low pass filters are used to filter the high frequency measured acceleration and moment. The error in angular acceleration is then scaled by the quadrotor's inertia matrix to produce the desired moment which would subsequently sum up with the filtered moment increment.

The gains of the the two proportional controllers are listed in Table 6.1:

Gains	Value
K_q	$diag[2.34, 2.34, 2]$
K_ω	$diag[8, 8, 8]$

Table 6.1: Gains of the INDI attitude controller

6.3.1. Filter selection

As we have mentioned before, the efficacy of the INDI method lies behind the accuracy of the measured data and the filtering process. The angular acceleration which is computed by differentiating the angular rate of the quadrotors is highly noisy. The idea of adding low pass filter is to reduce the effect of the high-frequency noise or disturbance on the performance of the system while preserving the essential low frequency information of the system. Thus, the selection of the filter variables should be in such a way to keep the dynamic of the system in the computation.

After testing various types of first and second-order filters such as Butterworth, Chebyshev, and elliptic filters, a second-order low-pass filter has been chosen. Although the Butterworth filter proved effective for the attitude controller, it showed less efficacy for the vertical acceleration controller [8].

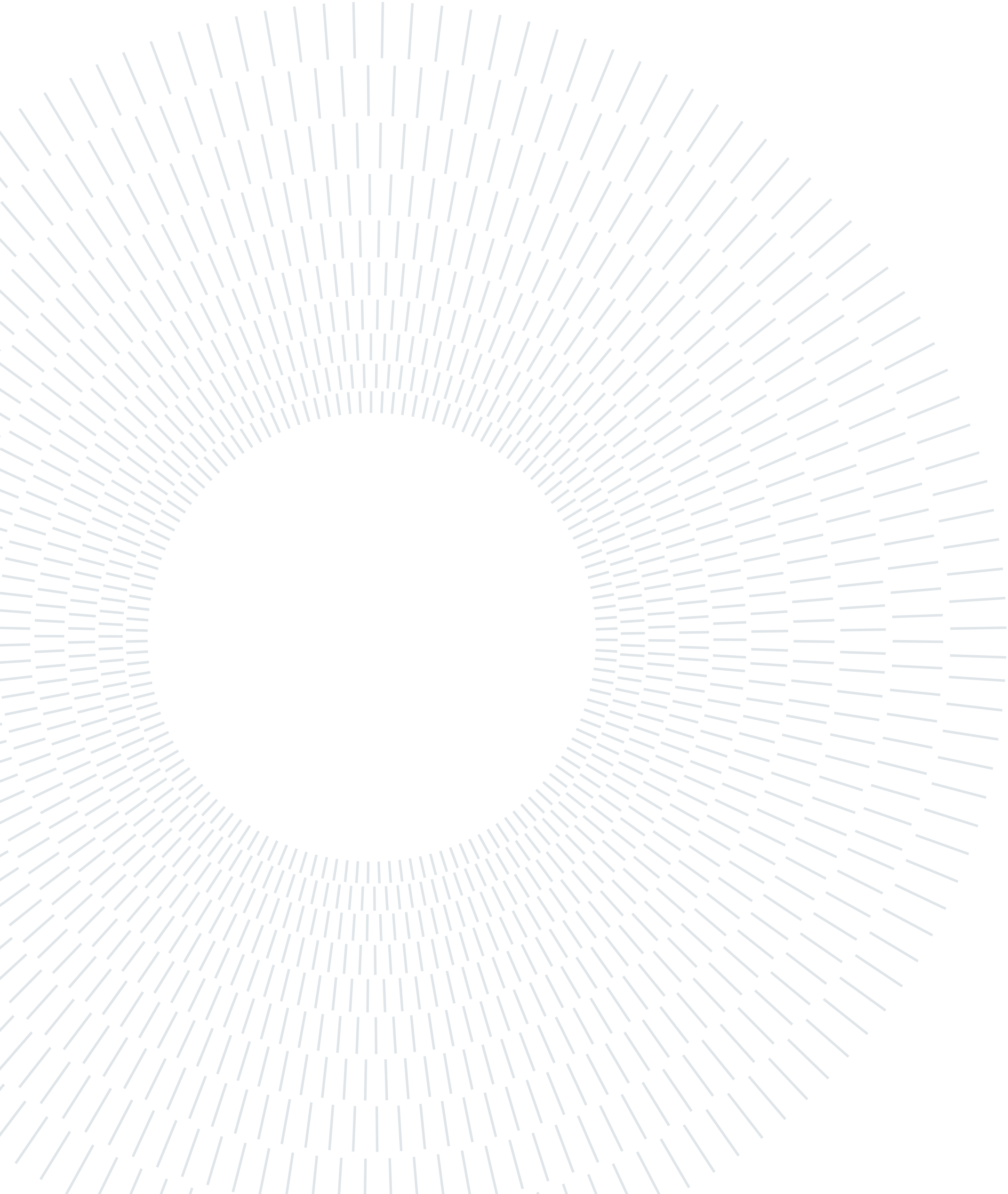
$$H(s) = \frac{\omega_n}{s^2 + 2\zeta\omega_n + \omega_n^2} \tag{6.35}$$

The information of the selected filter is presented in Table 6.2:

Gains	Value
ζ	0.6
ω_n	$1[rad/s]$

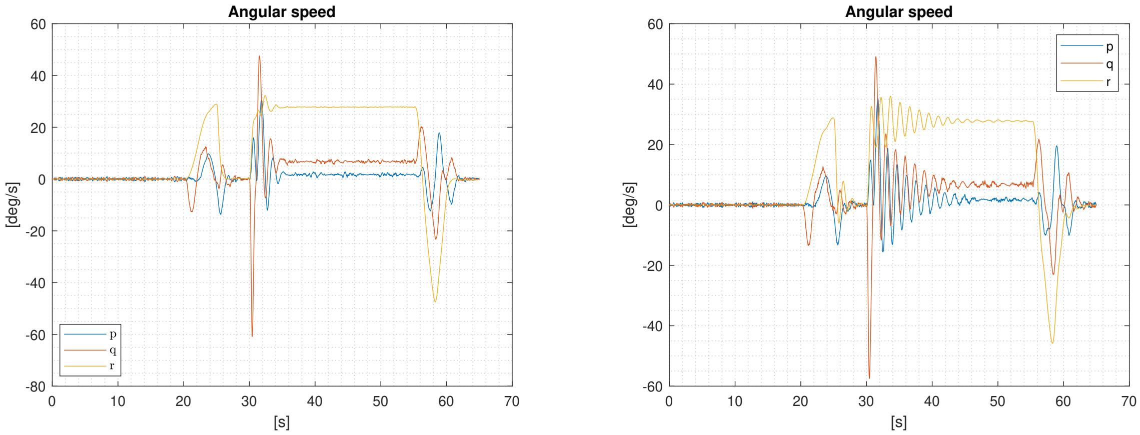
Table 6.2: Second order low pass filter parameters

The procedure to set the values for ω_n and ζ has mainly been done by trial and error around the cut-off frequency of the system. While we attempt to cancel out the measurement noise via the filter, we also need to consider that if the cutoff frequency of the filter is low, the delay may make the rejection of this disturbance too slow. Moreover, since adding a filter is followed by a phase delay, the same filter should be used across all applicable areas [8].

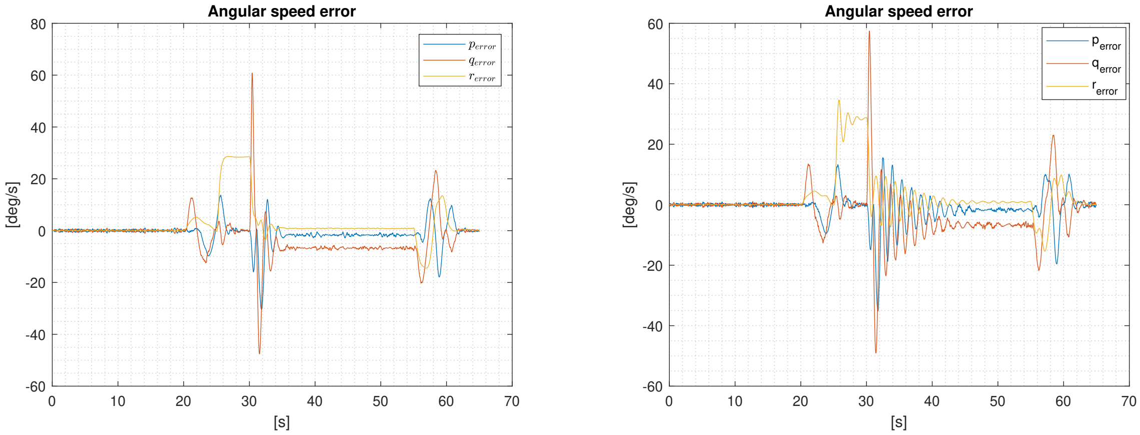


7 | Results and discussion for INDI attitude controller

The selection of a circular trajectory was made because of its inherent non-linearity, which was intended to highlight the discrepancies better and validate the reliability of the outcomes. The result for the attitude controller using INDI approach for a circular mode trajectory in comparison with the cascade P-PID method is highlighted in Figure 7.1. The cascade PID block diagram and information is presented in A.2.



(a) Angular velocity: INDI (b) Angular velocity: cascade P-PID

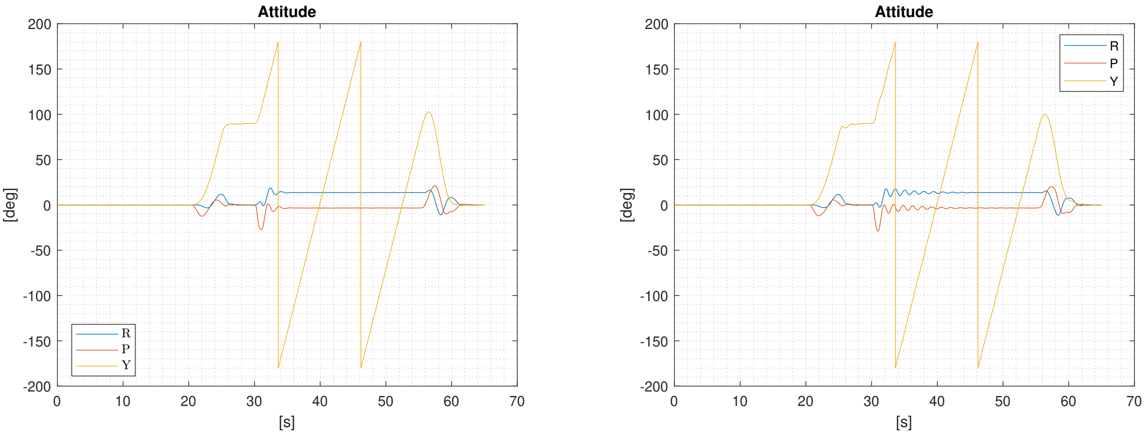


(c) Angular velocity error: INDI (d) Angular velocity error: cascade P-PID

Figure 7.1: Quadrotor angular velocity and angular velocity error comparison using INDI approach and cascade P-PID controller

Comparing Figures 7.1a and 7.1b, the oscillation in angular velocity notably diminishes with the implementation of the INDI controller approach. Employing just proportional gains for angular velocity and attitude regulation, outperforms the cascade P-PID controller method. Moreover, the absence of integral action leads to quicker responses and reduces transient time to achieve a steady-state condition. Figure 7.1c and 7.1d depict the error of the set-point and measured angular rate which again follows the same trend which the amplitude is decreased in the result of the INDI method.

Figure 7.2 demonstrates the difference in the result of the attitude of the circular mode trajectory in the mentioned two approaches.



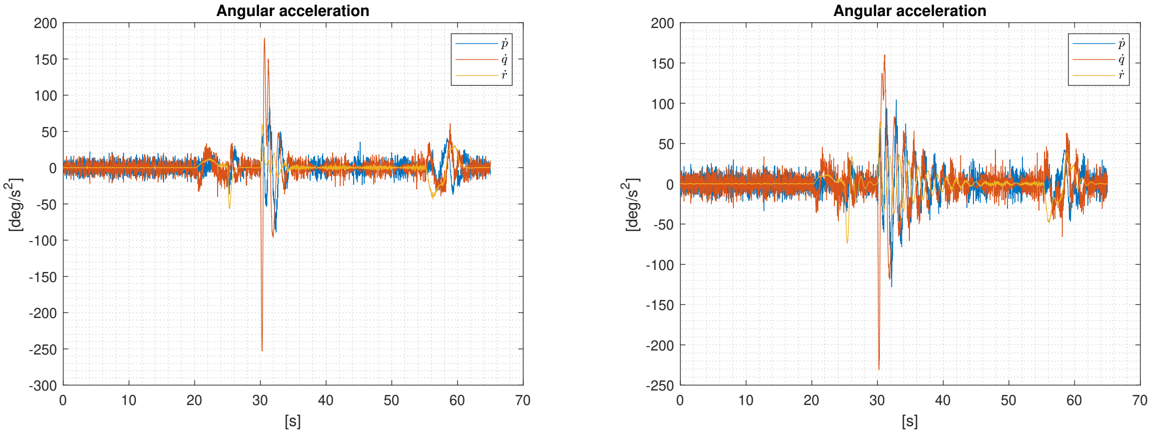
(a) Attitude: INDI

(b) Attitude: cascade P-PID

Figure 7.2: Quadrotor attitude comparison using INDI approach and cascade P-PID controller

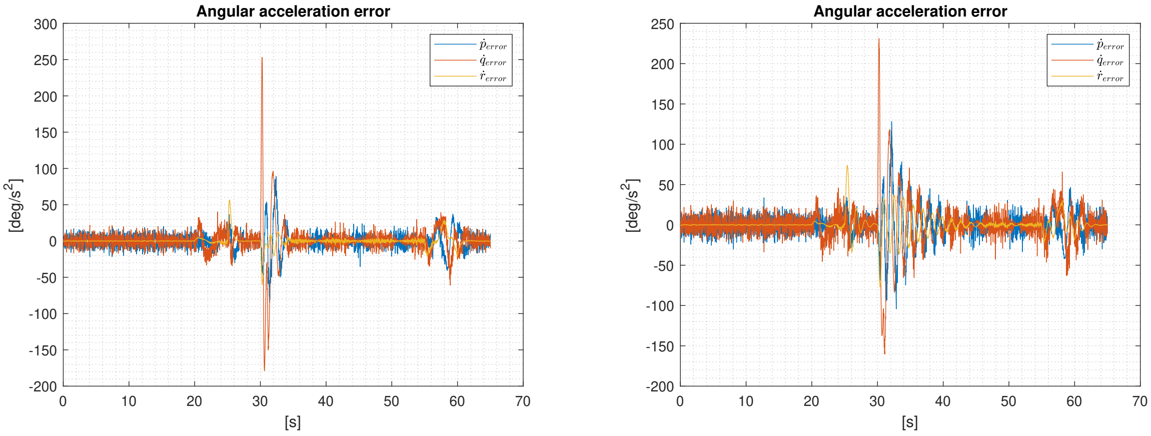
The plot shown in Figure 7.2a illustrates a slight enhancement in stability for both pitch and roll angles, characterized by a decrease in signal oscillations. Specifically, oscillations gradually diminish around the 35-second for the INDI algorithm, whereas for the cascade P-PID control method, oscillations decrease around the 46-second.

Figure 7.3 presents a comparison of the angular acceleration outcomes for a circular mode trajectory. By directly filtering the angular acceleration signal and incorporating measured angular acceleration into the control algorithm, noticeable reductions in signal amplitude compared to the cascade PID method are observed. Additionally, noise cancellation and oscillation reduction are clear.



(a) Angular acceleration: INDI

(b) Angular acceleration: cascade P-PID



(c) Angular acceleration error: INDI

(d) Angular acceleration error: cascade P-PID

Figure 7.3: Quadrotor angular acceleration and angular acceleration error comparison using INDI approach and cascade P-PID controller

The results indicate that the INDI implementation effectively mitigates noise and disturbances to a greater extent than the cascade P-PID algorithm. Moreover, despite the absence of integral action in the algorithm, the INDI approach demonstrates a faster response to high-amplitude changes and achieves shorter transient times. This suggests that the INDI method exhibits superior adaptability and efficiency in handling dynamic changes, thereby improving overall system performance.

It is worth mentioning that by substituting the INDI attitude controller into the simulator, the quadrotor still demonstrates stable position performance, as depicted in Figure 7.4.

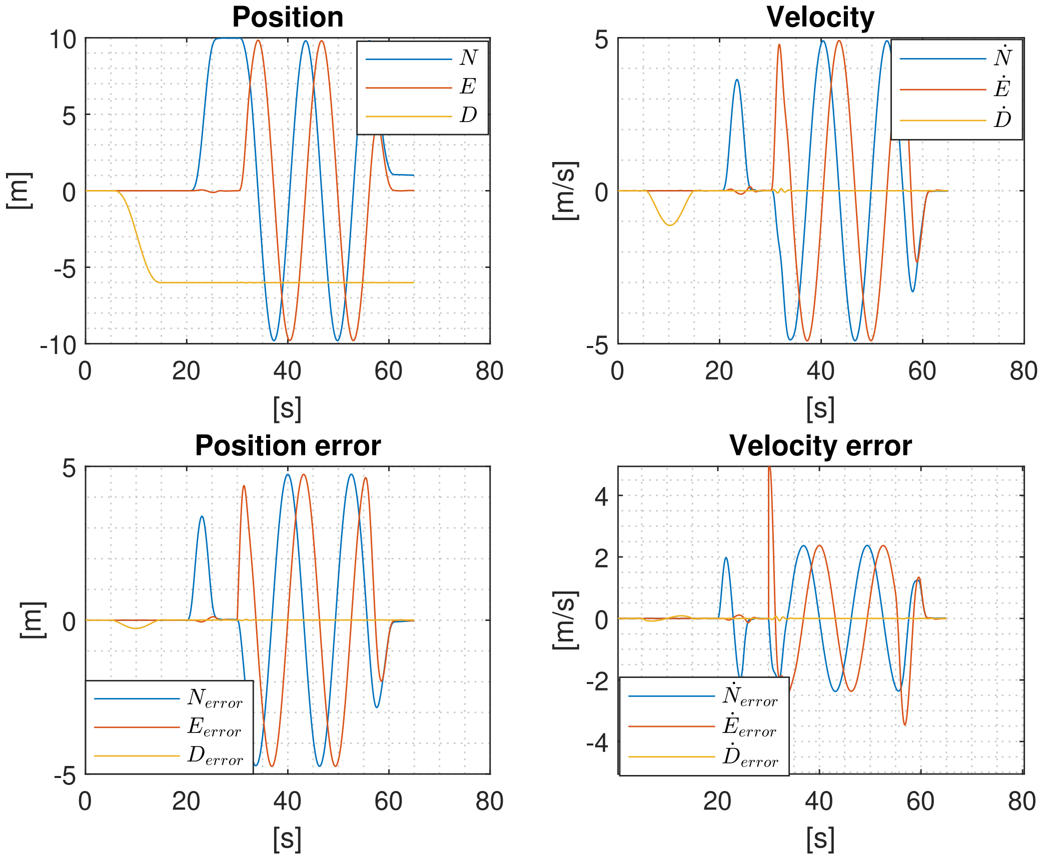
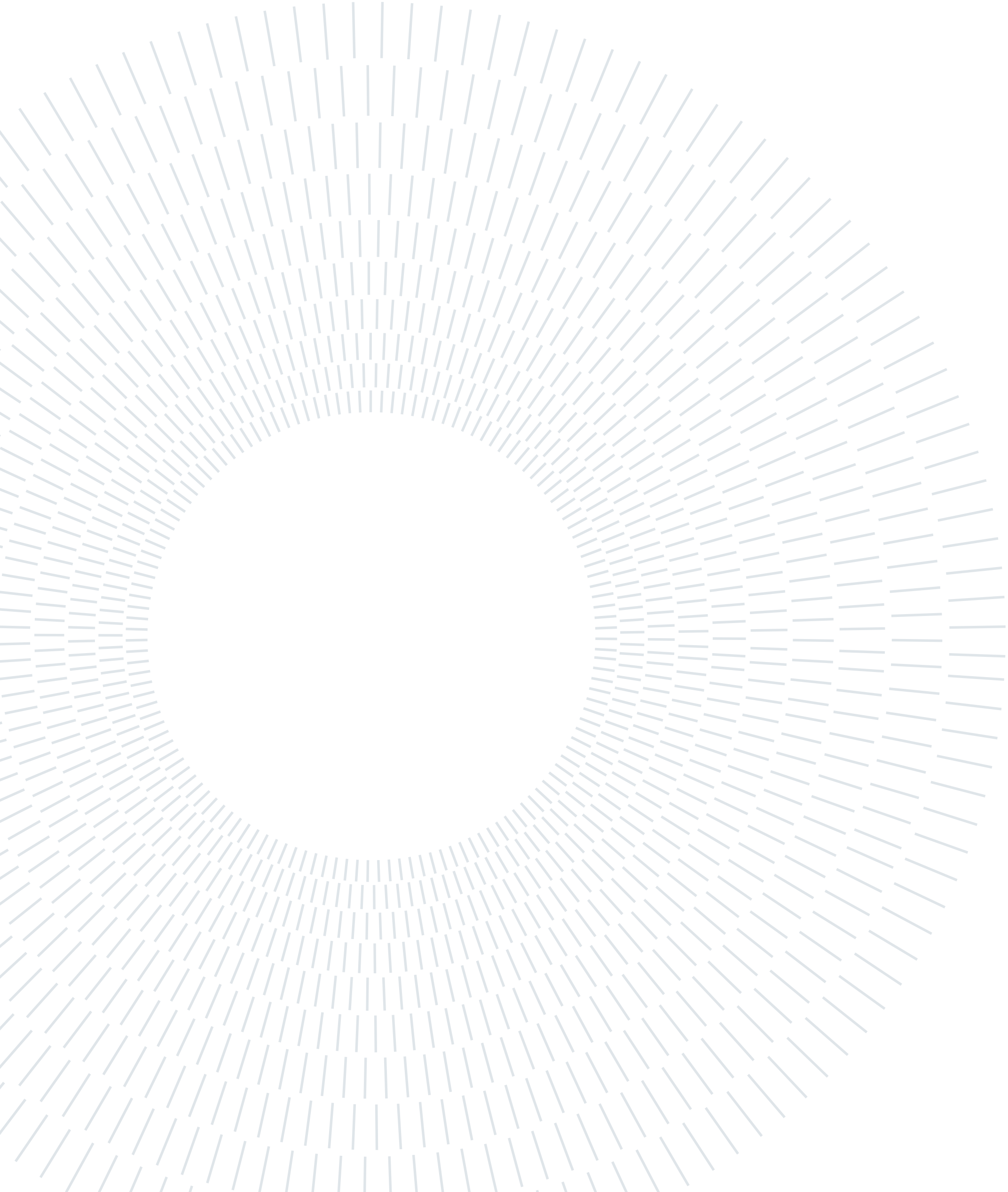


Figure 7.4: Result of the position and velocity performance



8 | Conclusions and future developments

The initial part of the thesis delved into the model-based design of quadrotor components up to the full quadrotor model, alongside the development of a linear position and attitude controller designed for the aircraft library within the Wolfram System Modeler. Additionally, several utility features were developed to enhance visualization and user-friendliness and facilitating the understanding of examples and components. These components are used as the foundation not only for quadrotors but also for hexa-copter and drones equipped with more propellers.

Furthermore, by leveraging the advanced Multibody library of the System Modeler, a more complex model of a multi-rotor vehicle can be designed. Implementing advanced control systems improves the stability of flight for the quadrotor and enables more precise state tracking, and improves its overall performance in wider operating range.

The second part of this thesis focused on the non-linear attitude controller of the quadrotor by applying INDI approach. INDI is a nonlinear controller, driven by a relatively straightforward design process and robust capabilities. Comparing INDI with NDI approach, NDI requires a detailed system model and it has dependency on states, while INDI only needs a control effectiveness matrix, which can deal with the uncertainties better. However, implementing INDI successfully requires estimating the state derivative and using a high sampling frequency.

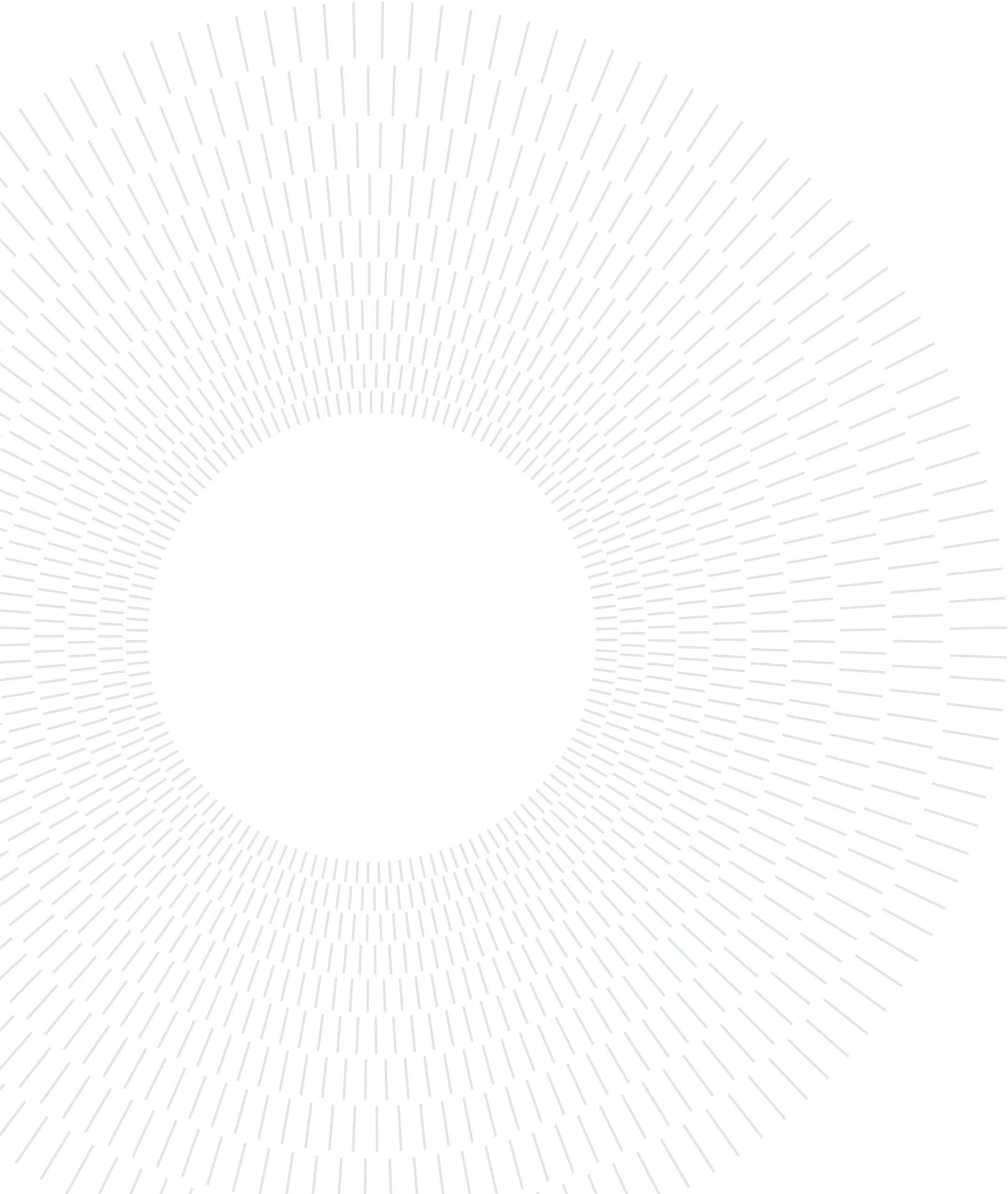
To address the requirement for estimating the state derivative in the INDI controller, an efficient solution was implemented using a second-order lowpass filter. This not only estimates the derivative but also effectively filters out high-frequency noise from sensor measurements.

For future works, an INDI algorithm can be applied for the vertical acceleration control. An attempted has been done for the vertical velocity control yet it didn't perform adequately and due to the time limit hasn't been done. A block diagram of the vertical position, velocity and acceleration controller using INDI approach is depicted in A.1.

Because of a slight discrepancy between the linear attitude model and the non-linear model, a precise analysis of the system's response in the frequency domain and uncertainty analysis did not occur. However, introducing an improved linearized attitude model would allow a more thorough analysis of the system's response.

An adaptive algorithm can be integrated to have a better estimate of the states values in real-time, enabling the system to adapt to stationary disturbances affecting the quadrotor.

Needless to say, while a detailed MATLAB and Simulink model provide adequate model of the quadrotor, real-time application with experimental tests is crucial for validating the robustness of the chosen control method against various disturbances that were not modeled. For instance, loss of performance due to battery consumption.



Bibliography

[1] R. F. E. Almeida. Incremental nonlinear dynamic inversion applied to quadrotor uav control, Nov 2017. URL `https://fenix.tecnico.ulisboa.pt/cursos/meaer/dissertacao/1972678479053224`.

[2] A. Esmaeilzadehrostam and E. E. Juarez. Wolfram Videos: Modeling Fighters and Drones Using System Modeler — wolfram.com. `https://www.wolfram.com/broadcast/video.php?v=3889`, 2023. [Accessed 04-03-2024].

[3] M. Giurato. Design, integration and control of multirotor uav platforms. 2020.

[4] F. L. Markley and J. L. Crassidis. *Fundamentals of spacecraft attitude determination and control*, volume 1286. Springer, 2014.

[5] A. I. Pte. About Aerial Industries Pte. Ltd. - Hardware company in Singapore | F6S — f6s.com. `https://www.f6s.com/company/aerial-industries-pte.-ltd#about`, 2023. [Accessed 04-03-2024].

[6] P. Simplício, M. Pavel, E. Van Kampen, and Q. Chu. An acceleration measurements-based approach for helicopter nonlinear flight control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, 21(8):1065–1077, 2013.

[7] J.-J. E. Slotine. Applied nonlinear control. *PRENTICE-HALL google schola*, 2:1123–1131, 1991.

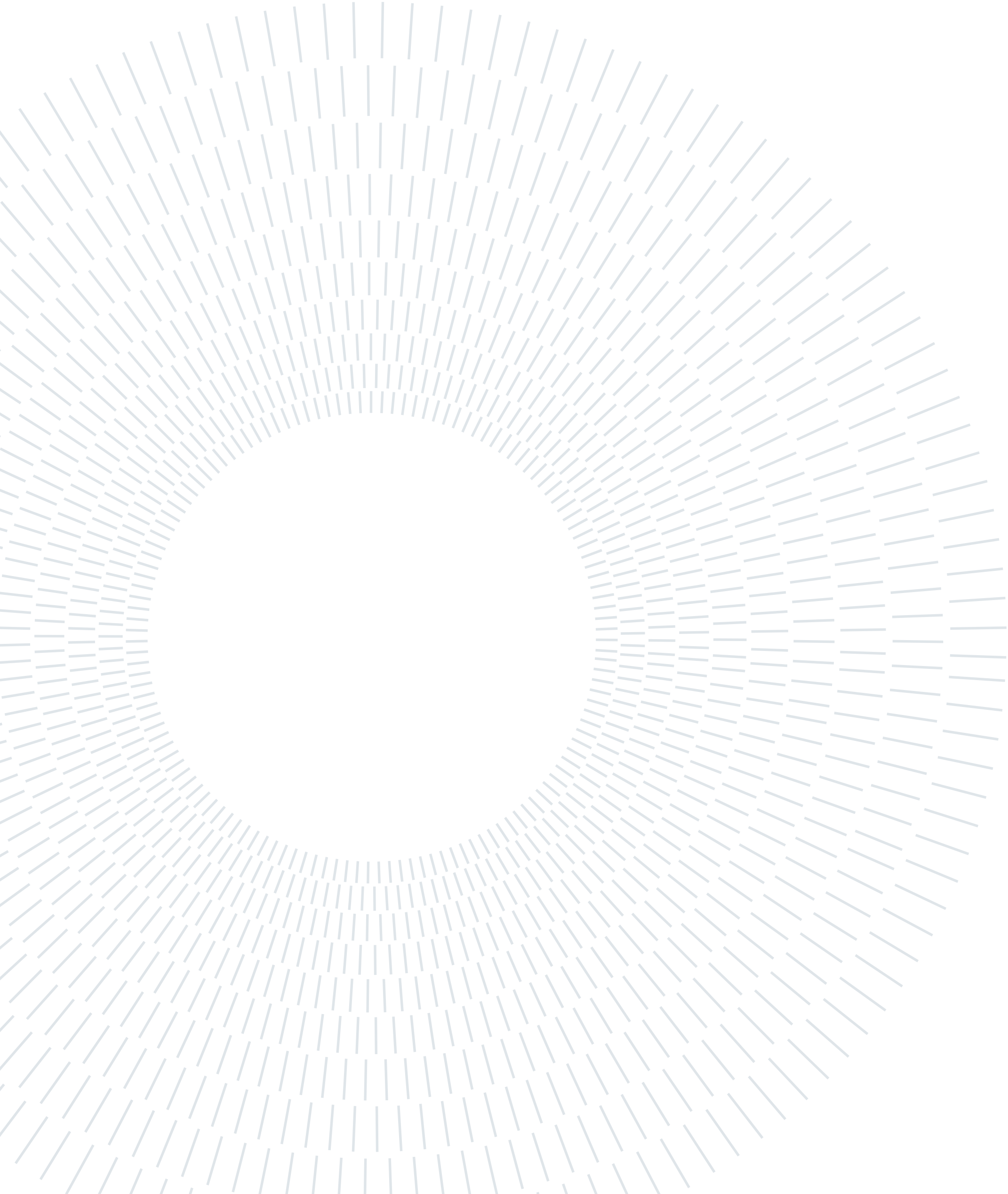
[8] E. J. Smeur, G. C. de Croon, and Q. Chu. Cascaded incremental nonlinear dynamic inversion for mav disturbance rejection. *Control Engineering Practice*, 73:79–90, 2018.

[9] G. Staples. Propeller static & dynamic thrust calculation. *Electric RC Aircraft Guy*, 2013.

[10] E. Tal and S. Karaman. Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness. *IEEE Transactions on Control Systems Technology*, 29(3):1203–1218, 2020.

[11] P. Wang, Z. Man, Z. Cao, J. Zheng, and Y. Zhao. Dynamics modelling and linear control of quadcopter. In *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pages 498–503. IEEE, 2016.

[12] Wikipedia. Wolfram SystemModeler — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Wolfram%20SystemModeler&oldid=1194652735`, 2024. [Online; accessed 27-February-2024].



A

Appendix A

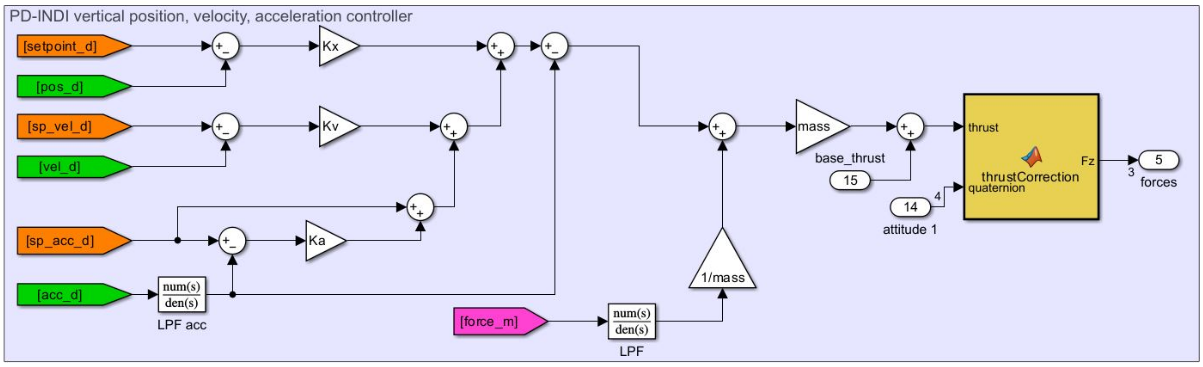


Figure A.1: INDI algorithm for vertical acceleration along with PD controllers for vertical position and velocity[10]

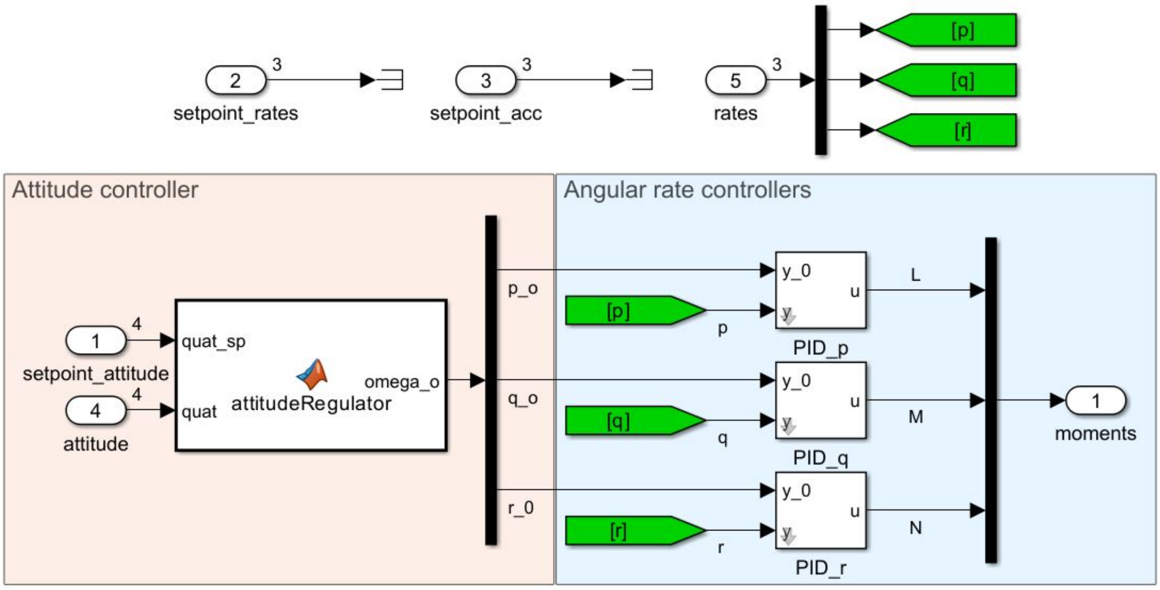
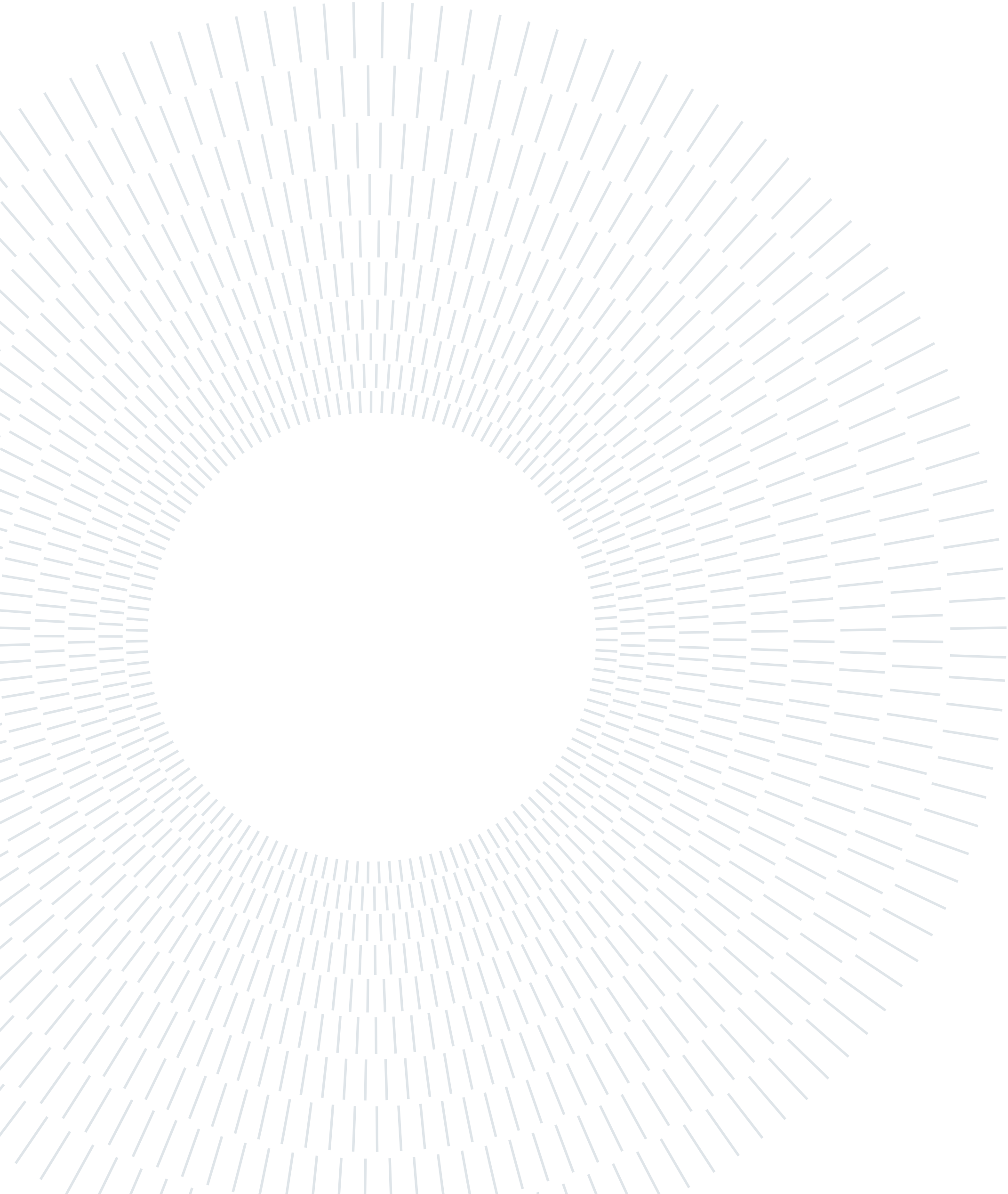
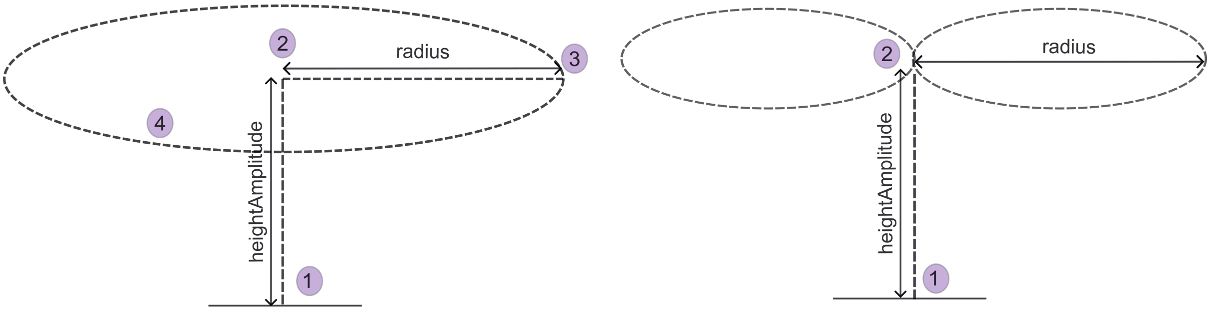


Figure A.2: Cascade P-PID attitude controller of the quadrotor[3].



B | Appendix B

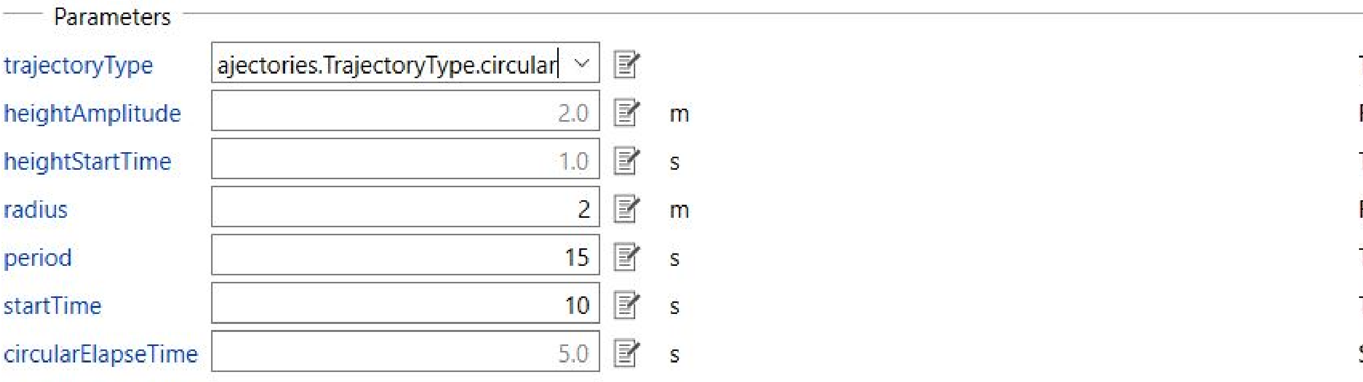
The mission profile of these two trajectories is depicted below:



(a) Schematic of the circular trajectory (b) Schematic of the infinity trajectory

Figure B.1: Reference trajectory

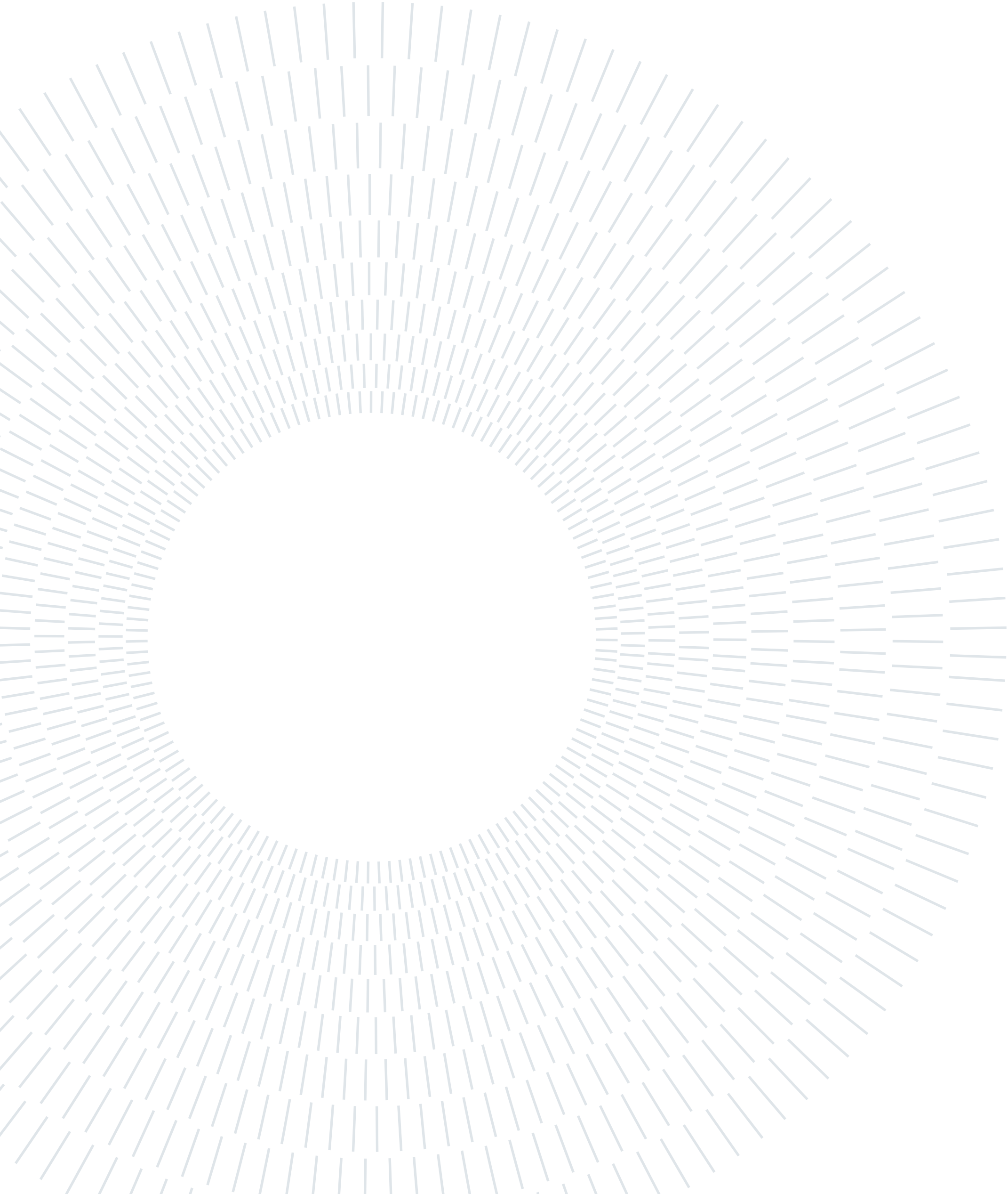
Figure B.2 shows the tenable parameters for the reference trajectory block.



Parameters

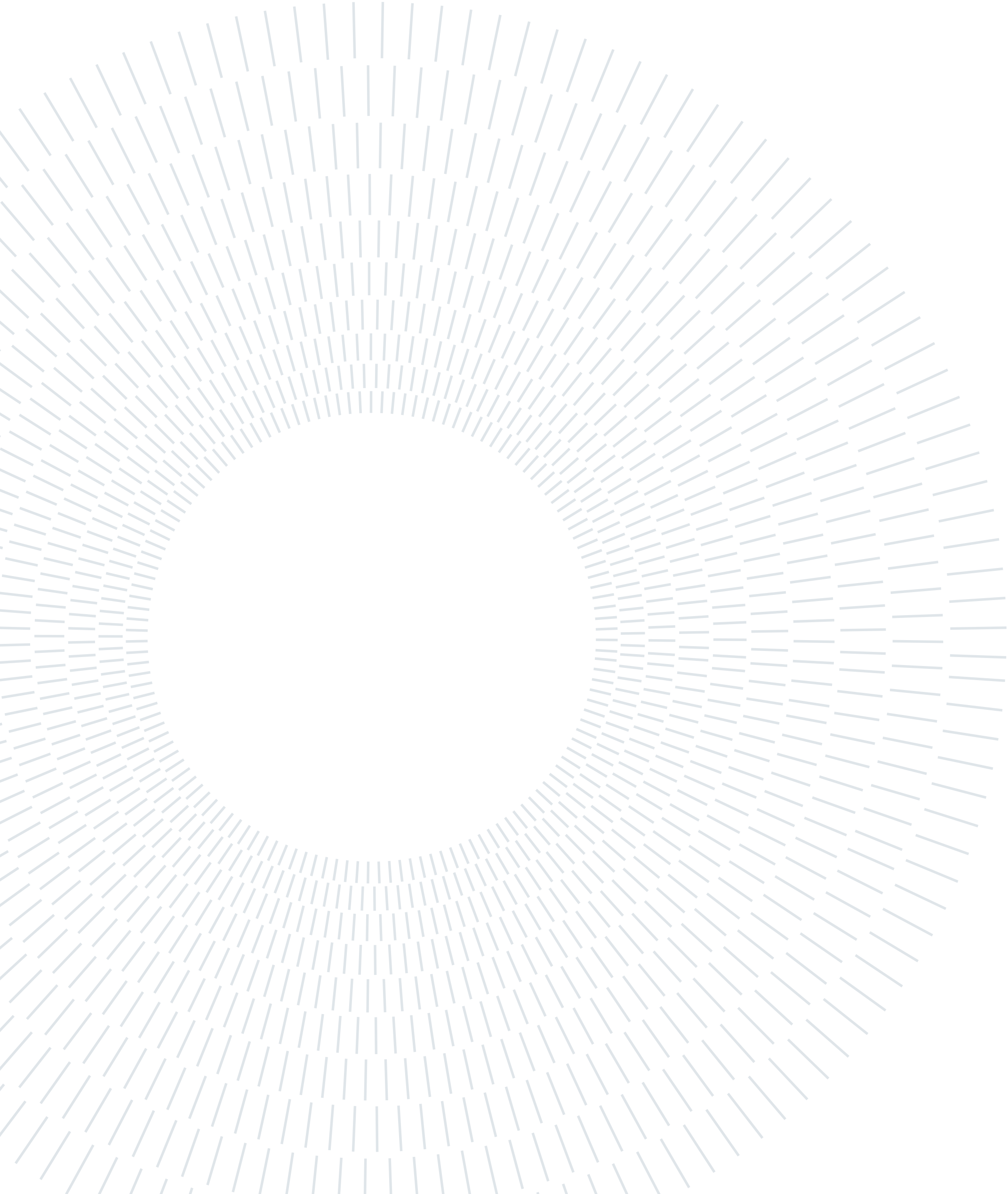
trajectoryType	ajectories.TrajectoryType.circular	
heightAmplitude	2.0	m
heightStartTime	1.0	s
radius	2	m
period	15	s
startTime	10	s
circularElapseTime	5.0	s

Figure B.2: Parameter tab for the reference trajectory



List of Figures

List of Tables



Acknowledgements

I am deeply grateful to Jan Brugard, CEO of Wolfram MathCore, for his invaluable support and mentorship during my internship. His guidance and insights have played a pivotal role in shaping my understanding of the modeling and development of the multirotor package of the aircraft library and have inspired me to strive for excellence in my work.

I would also like to thanks to Ankit Naik, Sergio Camilo Vargas Avila, and Vedat Şenol for their assistance, expertise, and encouragement throughout my internship. Their dedication to fostering a collaborative and supportive environment has greatly enhanced my learning experience and has contributed significantly to the success of my internship project.

Additionally, I am indebted to my sister, family and friends for their endless love, encouragement, and understanding. Their unwavering support and belief in me have been a constant source of strength and motivation, and I am truly grateful for their presence in my life.