Politecnico di Milano
School of Industrial and Information Engineering

Master of Science in Space Engineering

Model reference adaptive control strategies for tiltable propellers UAVs

Advisor:	Prof. Marco Lovera
Co-Advisor:	Dr. Davide Invernizzi
	Eng. Salvatore Meraglia

Thesis by:
Lorenzo Bendinelli   Matr. 925395

Academic Year 2020–2021

To my family . . .

Acknowledgments

I would like to express my deepest and special gratitude to my mother Marzia, father Stefano and brother Marco who helped and supported me in all these years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Even if with some regrets, they allowed me to study what I wanted where I always wanted, making sure that my only thought was to do what I liked most. They have always stayed by me, day after day, with their valuable advices, their guidance and their patience that cannot be underestimated.

I am also grateful to my girlfriend and my closest friends who, in their sincere way, have always eased my days. They unconsciously gave me the right spirit to face all those moments in which the things got a bit discouraging. They were able not to make me feel the distance as a so big obstacle. When I came back home they were always there like old times. Our weekends, our escapades, our chats have been essential and very helpful in my everyday life. Thanks guys for always being there for me.

I would like also to extend my sincere thanks to my thesis Co-Advisors: Dr. Invernizzi and Eng. Meraglia that were always open whenever I ran into a trouble spot or had a question about my research or writing. They consistently allowed this paper to be my own work, but steered me in the right direction whenever they thought I needed it.

Finally, a special thanks goes also to all those people, which are too many to be listed, that I met during my course of study. With them, I shared all the university adventures, exams, challenges and, especially, the deep passion for what we have studied together. It has been a pleasure to work with you as a team, facing our failures and collecting our successes. I will always carry you in my heart.

Abstract

Nowadays, unnamed aerial vehicles (UAV) are more and more exploited in many field of applications. The term UAV, generally called drone, usually refers to a category of multi-rotor Vertical Take-Off and Landing (VTOL) platforms equipped with four or more motors characterized by the absence of the pilot aboard the aircraft. Currently, the most used and famous UAVs are the well known co-planar drones, which were the first ones to be developed. Therefore, there is an extensive documentation regarding methods and control systems that can improve their performance; however, they are physically limited because of their thrust production along a single direction only. This work will focus on a new kind of drone and in particular on quadrotors UAVs with tiltable propellers. These platforms have an over-actuated structure that ideally allows to independently control the six Degrees Of Freedom (DOFs) of a rigid body in space. This is possible thanks to the use of eight actuators: four brush-less motors, at which the propellers are fixed and that generate the thrust; four servo-actuators, which tilt the motors and that change the direction of the thrust vector in the aircraft frame. Today, the control algorithms for this type of applications are still under study and they present some important limitations mainly related to the complex and uncertain dynamics of tiltable propellers UAVs.

For this reason, the purpose of this thesis is to design, analyse and apply model reference adaptive control algorithms to a mathematical model of tiltable propellers UAVs. This work will try to place emphasis on the quadrotor tracking capabilities together with the control system ease of tuning and implementation. In order to comprehend the possible improvements of these new architectures, the system response will be also compared to the one of a tiltable propellers UAV in which the last state of the art controller runs.

More in detail, after a short presentation of the object under study, first of all the present thesis will illustrate the main limitations of the current control strategies and it will present the most advanced tilt-arm simulator adopted. Secondly, it will follow a recap of the most recent model reference adaptive control architectures, focusing on their mathematical background, their limitations and their possible strategy of implementation for the problem of controlling a tiltable propellers UAV. More specifically, the Model Reference Adaptive Controller (MRAC) and Closed-loop Model Reference Adaptive Controller (CMRAC) will be analysed.

We will consider a decoupled control design for the attitude and the position dynamic, since it is easier, more intuitive, and we will demonstrate to be sufficient to obtain satisfactory results. In this context, the UAV uncertain model will be derived, given that it is required to correctly implement an adaptive control architecture. Focusing on the attitude, a first simplified tiltrotor simulator will be developed to test the above cited control strategies, in order to understand which one is more suitable and performing for this precise application. Afterwards, the controller chosen will be tuned and proven separately on the attitude and on the position trough the high fidelity simulator, where many uncertain effects and nonlinear dynamics will be taken into account. The results will be compared with the ones computed with the same simulator, but in which the last state of the art PID controller will be implemented. Then, the adaptation will be activated on both the UAV dynamics and the simulator outcomes obtained will be analysed. Finally, once the improvements will be verified with respect to an existing PID-based controller, a Monte Carlo simulation will be conducted to figure out the robustness of the proposed control algorithm to some uncertain model variables not included in the uncertain model used in the adaptive controller.

Sommario

Al giorno d'oggi, gli aeromobili a pilotaggio remoto (APR) sono sempre più sfruttati in molti campi di applicazione. Il termine APR, generalmente chiamato drone, di solito si riferisce ad una categoria di piattaforme multi-rotore a decollo ed atterraggio verticale dotate di quattro o più motori caratterizzati dall'assenza del pilota a bordo dell'aeromobile. Attualmente, i più usati e famosi APR sono i ben noti droni co-planari, che sono stati i primi ad essere sviluppati. Pertanto, esiste una vasta documentazione riguardante metodi e sistemi di controllo che hanno l'obiettivo di migliorare le loro prestazioni; tuttavia, sono fisicamente limitati a causa della loro produzione di spinta lungo una sola direzione. Questo lavoro si concentrerà su un nuovo tipo di drone ed in particolare sui quadrirotori con eliche inclinabili. Queste piattaforme hanno una struttura sovra-attuata che idealmente permette di controllare indipendentemente i sei gradi di libertà di un corpo rigido nello spazio. Questo è possibile grazie all'utilizzo di otto attuatori: quattro motori brush-less, a cui sono fissate le eliche e che generano la spinta; e quattro servo-attuatori, che inclinano i motori e che cambiano la direzione del vettore di spinta nel sistema di riferimento dell'aeromobile. Oggi, gli algoritmi di controllo per questo tipo di applicazioni sono ancora in fase di studio e presentano alcune importanti limitazioni legate principalmente alla complessa e incerta dinamica che caratterizza gli APR ad eliche inclinabili.

Per questo motivo, lo scopo di questa tesi è di progettare, analizzare e applicare algoritmi di controllo adattivo a modello di riferimento ad un modello matematico del drone ad eliche inclinabili. Questo lavoro cercherà di porre l'accento sulle capacità di tracking del quadrirotore insieme alla facilità di taratura del sistema di controllo e alla sua implementazione. Per comprendere i possibili miglioramenti di queste nuove architetture, la risposta del sistema sarà anche comparata a quella di un APR ad eliche inclinabili in cui gira l'ultimo controllore all'attuale stato dell'arte.

Più in dettaglio, dopo una breve presentazione dell'oggetto in esame, prima di tutto la presente tesi illustrerà i principali limiti delle attuali strategie di controllo e presenterà il simulatore adottato più avanzato. In secondo luogo, seguirà un riepilogo delle più recenti architetture di controllo adattivo a modello di riferimento, concentrandosi sul loro background matematico, sui loro limiti e sulla loro possibile strategia di implementazione per il problema di controllo di un drone ad eliche

inclinabili. Più specificamente, saranno analizzati il Model Reference Adaptive Controller (MRAC) e il Closed-loop Model Reference Adaptive Controller (CMRAC). Prenderemo in considerazione un design di controllo disaccoppiato per la dinamica di assetto e di posizione, dal momento che è più facile, più intuitivo, e dimostreremo di essere sufficiente per ottenere risultati soddisfacenti. In questo contesto, il modello incerto dell'APR verrà derivato, dato che è necessario per implementare correttamente un'architettura di controllo adattivo. Concentrandosi sull'assetto, verrà sviluppato un primo simulatore semplificato per testare le strategie di controllo sopra citate, al fine di capire quale sia la più adatta e performante per questa precisa applicazione. Successivamente, il controller scelto verrà tarato e collaudato separatamente sull'assetto e sulla posizione attraverso il simulatore ad alta fedeltà, dove verranno presi in considerazione molti effetti incerti e dinamiche non lineari. I risultati saranno confrontati con quelli calcolati con lo stesso simulatore, ma in cui sarà implementato l'ultimo controller PID all'attuale stato dell'arte. Quindi, l'adattamento sarà attivato sia sulla dinamica di assetto che di posizione e i risultati ottenuti dal simulatore verranno analizzati. Infine, una volta verificati i miglioramenti rispetto ad un controller PID, verrà condotta una simulazione di Monte Carlo per capire la robustezza dell'algoritmo di controllo proposto ad alcune variabili di modello incerto non incluse nel modello incerto utilizzato per il controllore adattivo.

Contents

List of Figures

List of Tables

Introduction

In recent years, the interest for the multirotor Unmanned Aerial Vehicles (UAVs) has grown due to their flexibility, the wide range cost, which can span from few to some thousands euros, and the fact that they are environmental friendly. Thanks to their dimensions and their autonomous capabilities, they can be used in many fields of applications such as: surveillance, inspections, scientific research, agriculture, military and much more. Today, fixed co-planar multirotors are the most used ones, mainly because of their construction simplicity and the easiness of the control algorithms that allow them to fly. However, like any other machine, they present some limitations. One of the most constrictive is that they are under-actuated platforms: in other words, they cannot independently control the attitude with respect to the position. In this scenario, the UAVs with thrust vectoring capabilities are really interesting because they offer the possibility of increasing the vehicle performances.

This work will focus on a particular platform, named tilt-arm UAV, in which the UAV propellers are installed on their respective brackets that are, in turn, attached to servo actuators. These latter have the ability to rotate the brackets making the propellers tiltable, hence the name "tilt-arm" for this kind of systems. From a physical point of view, this leads to a fully actuated aerial vehicle with, theoretically, the capability to reach a certain position in a completely independent way with respect to its attitude. Moreover, they are also efficient in terms of power consumption and they do not lead to the generation of internal forces in hovering conditions. Although, these kinds of multirotors are mechanically and electronically more complex than the co-planar UAVs. Since the tiltrotor is an unstable system, it needs a control algorithm capable to stabilize and control it. The state of the art of such controllers refers to different attempts such as the approach of Ryll et al. (FDBL), a quaternion based P/PID controller (CPID), the approach of Kamel et al. and the approach of Invernizzi et al. (IQTO), which are typically more suitable for co-planar multirotors. They have been compared in the literature, for more details one should see [2], where it has been shown that they offer similar results, since they are based on simplified dynamic models that limit the achievable performances. Therefore, the purpose of this thesis is to find a control algorithm with good tracking capabilities and simplicity of tuning, in order to improve the actual research in the field of control systems for tiltable propellers

UAVs. A possible solution has been found in adaptive controllers. Differently from the previous cited architectures, an adaptive control system is characterized by the ability to "learn" from the state measurements, and to autonomously "adapt" itself against uncertainties in the system or non-predictable events. More specifically, in this work two different adaptive approaches will be analysed: the Model Reference Adaptive Controller (MRAC) and the Closed-loop Model Reference Adaptive Controller (CMRAC). The words "Model Reference" refer to the working principle of these particular architectures. In fact, the idea is to build up a dynamic mathematical model, the so called reference model, which describes the tiltrotor ideal behaviour generating the desired state. Then, this information will be used by the controller to estimate some uncertain model parameters, to automatically produce the correct control input capable to bring the UAV dynamic as close as possible to the ideal one. The two cited adaptive approaches are very similar and differ for the reference model form only. In fact, the MRAC is made by a reference model that simulates exactly the UAV dynamic, while the CMRAC has an additional term that takes into account for the difference between the plant and the reference model state (closed-loop). All the results will be verified trough the use of different numerical simulations with increasing complexity. Since the tiltable propellers UAV is able to independently control the position and the attitude, a decoupled approach will be chosen to design the control system. More specifically, the first simulations and the tuning process will be carried out separately on the attitude and on the position, then the complete adaptive system will be proven on the full model.

This thesis will be structured in:

- Chapter 1. Mathematical modelling and control of tiltable propellers UAVs. This chapter will firstly present the tiltable propellers UAV mathematical model and an its simplified version more suitable for the control problem. Then, an overview of the current state of the art control systems will be illustrated highlighting their limitations. Finally, the most complex numerical simulator used in this work will be described.

- Chapter 2. Adaptive controllers introduction. In this chapter, a brief overview of the MRAC and CMRAC approaches will be shown, focusing on their peculiarities. Then, more specifically, we will describe the control architecture chosen and the instability phenomena that usually affect these kinds of control systems. Possible solutions to these problems will be analysed considering different adaptive laws.

- Chapter 3. Uncertain Model. This chapter will begin with the derivation of the UAV linear plant. After that, considering a decoupled approach for the attitude and the position dynamic, the uncertain model parameters will be identified and the reasons of their choice will be analysed. Finally, the complete uncertain model will be computed.

- Chapter 4. Attitude adaptive control system. This chapter will focus on the design of the control system for the UAV attitude dynamic. Firstly, the details regarding the baseline controller, the reference model and the adaptive laws will be showed. Secondly, a simplified numerical simulator will be presented, and it will be used to understand the sensitivity of the gains characterizing the MRAC and CMRAC architectures. After the most suitable adaptive architecture has been chosen, the chapter will end up with the presentation of the results obtained with the complete tilt-arm simulator.
- Chapter 5. Position adaptive control system. This chapter will follow the same structure of chapter 4, but referring on the position dynamic. Also in this case, the details regarding the baseline controller, the reference model and the adaptive laws will be illustrated. Finally, the results computed with the tilt-arm simulator, considering the adaptation active on the position only, will be showed.
- Chapter 6. Attitude and position adaptive controllers. This last chapter will analyse the outcomes obtained by the tilt-arm simulator with the adaptation working on both the attitude and the position dynamic. The discussion will end up with the implementation of a Monte Carlo simulation, with the purpose of figuring out the system robustness to uncertainties in the mixer matrix.

Chapter 1

Mathematical modelling and control of tiltable propellers UAVs

In this chapter, after a short presentation of the platform, the complete mathematical model of UAVs with tiltable propellers will be presented. Then, under appropriate assumptions a simplified model will be derived in order to develop a sufficiently accurate simulator of the UAV dynamics. Moreover, the most recent control designs that have been proposed in the literature will be shown together with their advantages and disadvantages. Afterwards, considering an existing control law, its current limitations will be highlighted. Finally, the architecture of the most complex simulator developed will be illustrated with the help of a block diagram for an easier comprehension.

1.1 Mathematical model

The tiltable propellers quadrotor shown in 1.1 is a new type of Unmanned Aerial Vehicle that differs from the common fixed coplanar propellers drones by the capability to rotate its rotors. There exist two different configurations: *tilt-arm*, in which the arms and consequently the motors rotate, or *tilt-rotor*, in which the propellers only tilt. In both of them each rotor can rotate independently. In this work, we will always refer to these two configurations as tiltable propellers UAVs. The increased mechanical complexity of such UAV is repaid by the fact that the tiltable propellers structure is, in theory, able to completely decouple its attitude from its position. In other words, these kinds of UAV should be able to maintain a certain attitude independently from the position that they have to reach. On the contrary, it is common to see a coplanar propeller UAV that changes its attitude, in order to reach a certain position. Instead, the tiltable propeller UAV is a fully actuated platform that allows to perform complex full-pose manoeuvres, which

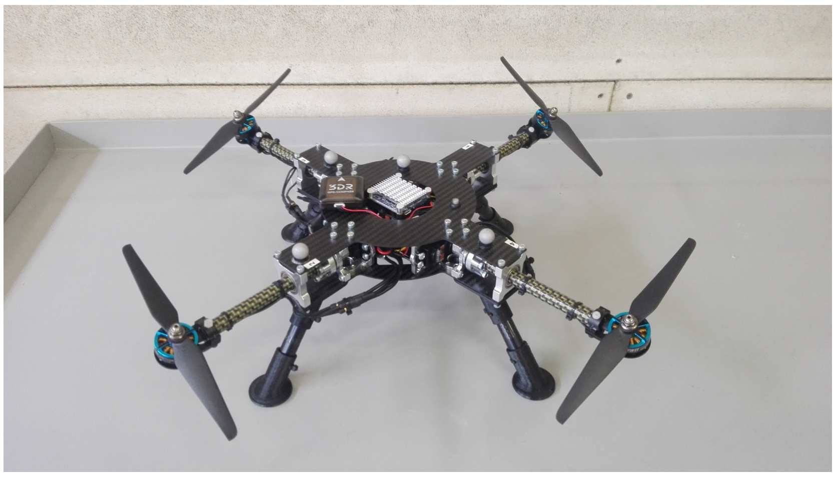

Figure 1.1: Tiltable propellers quadrotor UAV prototype

can be very helpful, for example, in inspection-like applications or in constrained environments. Assuming that the servo-actuators can perform a full rotation, these platforms have the ability to deliver both force and torque in any direction enhancing the aircraft interaction capabilities with the environment. On the other hand, they are characterized by an increased overall complexity of the drone. Since the focus of this thesis is on the design of suitable controls algorithm for tiltable propellers quadrotors, the set of equations that describe the UAV complete dynamic will not be derived: they will be simply taken from previous works on the same kind of system [1].

Before presenting the mathematical model, the following notation will be used throughout this work. $\mathbb{R}^n$ denotes the n-dimensional Euclidean space and consequently $\mathbb{R}^{m\times n}$ is the set of $m \times n$ real matrices. Given $x \in \mathbb{R}^n$, $\|x\| := (x_1^2 + \cdots + x_n^2)^{1/2}$ is the Euclidean norm. Given $A \in \mathbb{R}^{m\times n}$, the compact notation $A \in \mathbb{R}^{n\times n}_{>0}$ is used to represent a positive-definite matrix. The ith vector of the canonical basis of $\mathbb{R}^n$ is denoted as e_i (1 in the ith position and zeros elsewhere), therefore the identity matrix is defined as $I_n \in \mathbb{R}^{n\times n} := [e_1 \cdots e_i \cdots e_n]$. For compactness, 0_n and 0_{nm} is used to identify a matrix of zeroes with dimension $\mathbb{R}^{n\times n}$ and $\mathbb{R}^{n\times m}$ respectively. The set $SO(3) := \{R \in \mathbb{R}^{3\times 3} : R^T R = I_3, \det(R) = 1\}$ denotes the third-order special orthogonal group and $\mathbb{S}^1 := \mathbb{R} \bmod 2\pi$ denotes the unit circle manifold. Then, the map $S(\cdot) : \mathbb{R}^3 \to SO(3)$ given by:

$$\omega \to \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{1.1}$$

defines an isomorphism between $\mathbb{R}^3$ and the vector space of third order skew-symmetric matrices, i.e. $SO(3) := \{W \in \mathbb{R}^{3\times 3} : W = -W^T\}$, such that one has $S(\omega)y = \omega \times y$, $\forall \omega,\ y \in \mathbb{R}^3$, where $\times$ is the cross product. Note that the corresponding inverse is $S^{-1}(\cdot) : SO(3) \to \mathbb{R}^3$.

As reported in [2], the dynamics of this kind of platforms evolve on the product manifold $SO(3) \times \mathbb{R}^3 \times (\mathbb{S}^1 \times \mathbb{S}^1)^n$, for which one refers to $SO(3) \times \mathbb{R}^3$ as the base space whereas to $(\mathbb{S}^1 \times \mathbb{S}^1)^n$ as the shape space. As shown in figure 1.2, the base space is described by the configuration of a body-fixed reference frame $F_B = (O_B, \{b_1, b_2, b_3\})$ with respect to an inertial frame $F_I = (O_I, \{i_1, i_2, i_3\})$. In this context, $R \in SO(3) := [b_1, b_2, b_3]$ is the attitude matrix that describes the orientation of F_B with respect to F_I. b_1, b_2 and $b_3 \in \mathbb{S}^3$ are the body reference frame unit vectors written with respect to F_I. i_1, i_2 and $i_3 \in \mathbb{S}^3$ are the inertial reference frame unit vectors, and O_I, O_B are respectively the inertial and body reference frame origins.

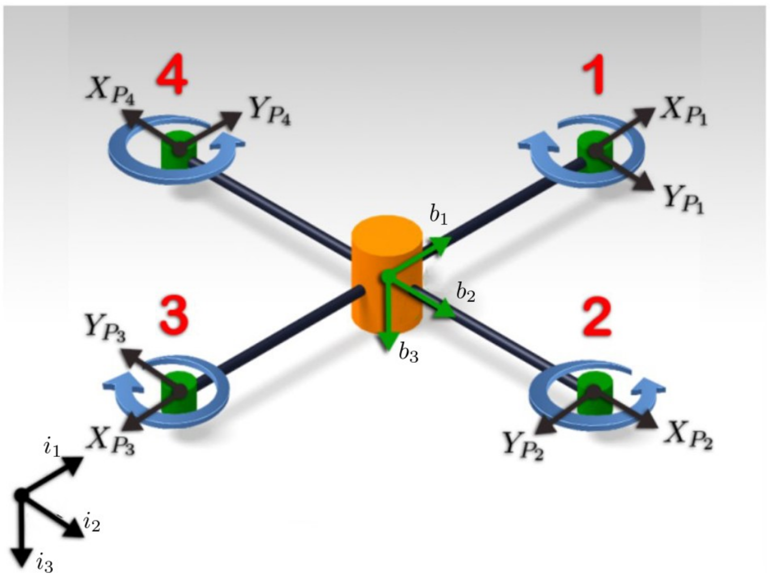

Figure 1.2: UAV reference frames and axis [1]

Then, referring to figure 1.3, the relative configuration of the ith tilt arm with respect to the main body belongs to the shape space. It is parametrized by the angles $(\beta_{ai}, \theta_{ri}) \in \mathbb{S}^1 \times \mathbb{S}^1$, where β_{ai} and θ_{ri} describe respectively the inclination of the ith arm axis and the angular position of the propellers, both with respect to a fixed direction in the body frame. Moreover, introducing the relative angular velocity of the tiltrotor ith arm as: $\dot{\beta}_{ai} := \omega_{ai} \in \mathbb{R}$ and of the rotor as: $\dot{\theta}_{ri} := \omega_{ri} \in \mathbb{R}_{\geqslant 0}$, and assuming linear dynamics for the servo actuators and the motors' closed loop system, the actuator dynamics is described by:

$$\beta_{ai} = G_s(s)\beta_{ai}^d \tag{1.2}$$

$$\omega_{ri} = G_m(s)\omega_{ri}^d \tag{1.3}$$

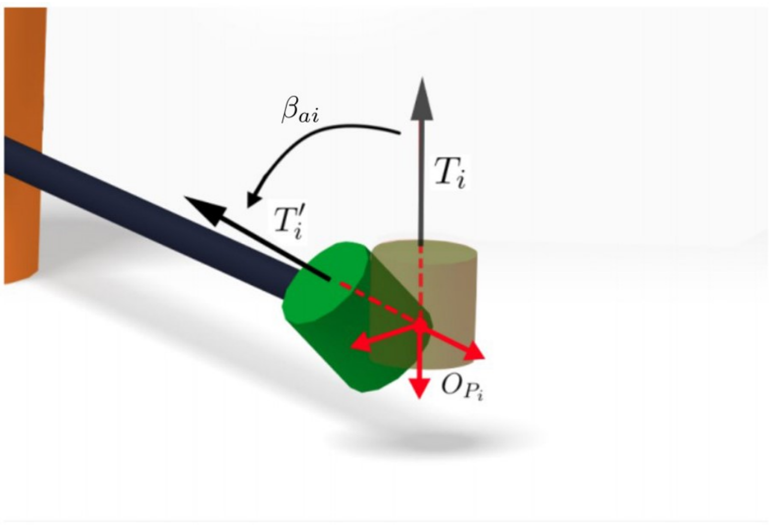

Figure 1.3: Propeller frame [1]

where β_{ai}^d and ω_{ri}^d are the desired tilt angle and rotor velocity, which are the control inputs of the system. Collecting the introduced angles and angular velocities in two different single vectors as: $\theta_s := (\beta_{a1}, \cdots, \beta_{an}, \theta_{r1}, \cdots, \theta_{rn})$ and $\omega_s := (\omega_{ai}, \cdots, \omega_{an}, \omega_{ri}, \cdots, \omega_{rn})$, it is possible to write the tiltrotor complete mathematical model [2], which derivation can be found in [1]:

$$\begin{cases} \dot{x} = v \\ \dot{R} = RS(\omega) \\ \dot{\theta}_r = \omega_r \\ m\dot{v} - mRS(x_c)\dot{\omega} = -mge_3 - mRS(\omega)^2 x_c + R(f_c + f_e) + d_F \\ J(\theta_s)\dot{\omega} + mS(x_c)R^T\dot{v} = -S(\omega)J(\theta_s)\omega - mgS(x_c)R^T e_3 + d_T + \tau_c + \tau_e \end{cases} \tag{1.4}$$

where $x \in \mathbb{R}^3$ is the position of O_b with respect to O_I, $v \in \mathbb{R}^3$ is the inertial translational velocity, $\omega \in \mathbb{R}^3$ is the body angular velocity, $J(\theta_s) = J(\theta_s)^T \in \mathbb{R}^{3\times3}_{>0}$ is the UAV inertia tensor as a function of the propellers position, $m \in \mathbb{R}_{>0}$ is the tiltrotor overall mass, $g = 9.81\frac{m}{s^2}$ is the gravity acceleration, $f_e := f_e(R, x, \omega, v, \dot{\omega}, \dot{v}, \theta_s, \omega_s, \dot{\omega}_s, t)$ and $\tau_e := \tau_e(R, x, \omega, v, \dot{\omega}, \dot{v}, \theta_s, \omega_s, \dot{\omega}_s, t)$ are respectively the external body force and torque, $f_c \in \mathbb{R}^3$ is the control force, $\tau_c \in \mathbb{R}^3$ is the control torque, $d_F \in \mathbb{R}^3$ and $d_T \in \mathbb{R}^3$ are the disturbance terms, $x_c \in \mathbb{R}^3$ is the center of mass position with respect to O_b.

1.1.1 Allocation algorithm

Referring to the dynamic system showed in the previous section, it is possible to see that the UAV plant can be controlled by generating a certain force and torque through f_c and τ_c. However, the real physical inputs to the system are the rotor angular velocities and the servo actuators angles, namely β_a and ω_r. For this reason, the standard approach splits the control design in two parts. A first one, usually put into action by specific controllers that compute the necessary force and torque, then a second one that is performed by an allocation algorithm, which transforms the control force and torque into physical inputs. As shown below, the allocation is based on the inversion of a non-linear static map which depends on some basic information about the UAV and has to deal with kinematic singularities, limited actuators bandwidth and a controller fast enough to track any desired set point, see [2]. In this thesis, we only concentrate on the first part of the control design; therefore the allocation algorithm is briefly recalled from previously works such as the one reported in [1].

Referring to a tiltable propellers quadrotor, the idea is to build up a mixer matrix M that relates the forces and torques coming from the controllers θ_a and ω_r as:

$$\begin{bmatrix} f_c \\ \tau_c \end{bmatrix} = M_{6\times 8}\; f_{u\,8\times 1}(\beta_a, \omega_r) \tag{1.5}$$

where f_u is a vector where the first four elements correspond to the horizontal components of the thrust generated by the four propellers and the last four to the vertical components, for example as the one showed in equation (1.6). While $M = M(\sigma, b)$ is a function of the arm length $b \in \mathbb{R}_{>0}$, and of the actuators properties $\sigma = \frac{K_q}{K_t}$ where $K_q \in \mathbb{R}_{>0}$ and $K_t \in \mathbb{R}_{>0}$ are respectively the torque and thrust coefficients that can be computed after a proper identification of the propellers.

$$f_u = \begin{bmatrix} f_{p1}\cos(\beta_{a1}) \\ f_{p2}\cos(\beta_{a2}) \\ f_{p3}\cos(\beta_{a3}) \\ f_{p4}\cos(\beta_{a4}) \\ f_{p1}\sin(\beta_{a1}) \\ f_{p2}\sin(\beta_{a2}) \\ f_{p3}\sin(\beta_{a3}) \\ f_{p4}\sin(\beta_{a4}) \end{bmatrix} \tag{1.6}$$

In the above expression, $f_{pi} = f_{pi}(\omega_{ri})$ represents the thrust of the i-th propeller, which is typically written using a quadratic law for multi-rotor UAVs control, namely $f_{pi} = K_t \omega_{ri}^2$. By inverting equation (1.5) it is possible to obtain the map that gives the physical inputs from the control force and torque:

$$f_u = M^+ \begin{bmatrix} f_c \\ \tau_c \end{bmatrix} \tag{1.7}$$

where M^+ is the Moore-Penrose pseudo-inverse of matrix M. Once the f_u vector is recovered from equation (1.7), one can compute ω_r and β_a from the following relations:

$$f_{pi} = \sqrt{f_u^2(i) + f_u^2(i+4)} \tag{1.8}$$

$$\omega_{ri} = \frac{f_{pi}}{K_t} \tag{1.9}$$

$$\beta_{ai} = \arctan 2(f_u(i+4), f_u(i)). \tag{1.10}$$

It is important to note that the mixer matrix M is constant and only depends on physical quantities and coefficients of the UAV: thanks to this property, its pseudo-inverse needs to be computed just once and it can be stored on-board as part of the controller implementation.

1.2 Model for control

In order to make the model suitable for the derivation of the control algorithms, some assumptions have to be made to derive a simplified version of the above dynamic system. In particular, we can consider the following assumptions:

1. The UAV is rigid: material flexibility is not considered.
2. The UAV inertia tensor is constant.
3. The control wrench $w_c := (f_c, \tau_c)$ delivered by the actuators spans $\mathbb{R}^3 \times \mathbb{R}^3$, the system is fully actuated.
4. The actuator dynamics are sufficiently faster than the expected system dynamics.
5. Full-state measurements are available.

The first assumption lies on the idea that the UAV and the rotors are sufficiently rigid so that flexibility does not affect the dynamics in the frequency range of interest. Instead, the second assumption is the result of several minor ones. In fact, also considering the system as rigid, the inertial tensor is a function of the angular position β_{ai} of each ith propeller: $J(\theta_s)$. However, the shape changes due to the actuation mechanism do not significantly affect the geometrical and dynamical properties of the system, so: $J(\theta_s) = J = \text{constant}$. The third and the forth allow to apply the allocation algorithm since we are considering manoeuvres that the actuators can easily perform taking into account their dynamic response and physical limitations. Finally, the last assumption is to say that, on board, there are some sensors and an estimation algorithm that provide real-time full-state measurements. Therefore, the set of equations describing the simplified tilt-arm dynamic is shown in equation (1.11):

$$\begin{cases} \dot{x} = v \\ \dot{R} = RS(\omega) \\ \dot{\theta}_r = \omega_r \\ m\dot{v} - mRS(x_c)\dot{\omega} = -mge_3 - mRS(\omega)^2 x_c + Rf_c + d_F \\ J\dot{\omega} + mS(x_c)R^T\dot{v} = -S(\omega)J\omega - mgS(x_c)R^Te_3 + d_T + \tau_c. \end{cases} \tag{1.11}$$

1.3 Overview of existing control designs

Regarding the control systems for tiltable propellers quadrotor, a brief recap of the current existing control strategies will be done.

The first that it is considered is a modification of the *Approach of Ryll et al. (FDBL)* showed in [2]. The idea is to consider a simplified feedback linearization controller that exploits f_c and τ_c as physical inputs and the invertibility of the allocation map. The main problem of this approach is that the angular accelerations and velocities of the tilting propellers are used as variables for control design. Indeed, these latter do not appear in the dynamic system and consequently a dynamic extension procedure has to be used. Moreover, the control law requires high computational power due to the presence of many integrators. Optimization strategies have to be employed for reducing the power loss and for avoiding the UAV to reach the hovering condition with inclined propellers, which produces internal forces and wastes power. Any suitable tuning method can be applied to obtain the desired performances.

The second control strategy is the *Approach of Kamel et al.* [7] that was applied to an hexacopter tilt-rotor UAV. This approach is based on a proportional integral stabilizer with feed-forward action for position tracking and on a cascade architecture for attitude tracking. The main limitations come from the fact that the controller needs the known of the center of mass position and there are not clear methods to estimate it. Furthermore, there are no integral actions in the attitude controller, and this can lead to problems when the propellers are not perfectly balanced.

The third is the *Approach of Invernizzi et al. (IQTO)* [2], which is based on conditional integrators and non linear stabilizers for position tracking and a geometric PID for attitude tracking. The main limitation in this case was found in the tuning phase. In particular, the large number of parameters and their non straightforward contribution to the system response makes the tuning very complex. In position tracking the controller provides worse results with respect to the other ones but it was showed to be quite good in attitude.

Finally, the last approach is the *Quaternion-Based P/PID Controller (CPID)*

[2] that relies on a proportional action in the outer loop and PID controller in the inner loop for both attitude and position as illustrated in figure 1.4. Referring

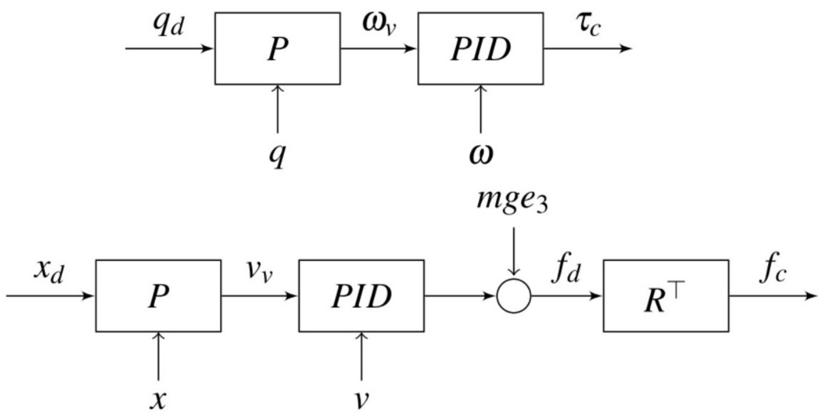

Figure 1.4: Nested attitude and position control loops [2]

to equation (1.12), the idea is the one used in many multirotor platforms such as the PX4 autopilot [8]. The outer attitude loop, which is based on the quaternion error and a switching strategy to avoid the unwinding phenomenon, computes the desired angular velocity, and in turn this is sent as a reference to the inner loop.

$$\begin{cases} \omega_v = -K_{po}^A sign(q_e)\mathbf{q_e} \\ \tau_c = K_{ff}^A \omega_v + (K_{pi}^A + K_i^A \frac{1}{s})(\omega_V - \omega) - K_d^A \frac{s}{1+\frac{s}{N}} \omega \\ v_v = -K_{po}^P e_x \\ f_c = R^T (K_{ff}^P v_v - (K_{pi}^P + K_i^P \frac{1}{s})(v - v_V) - K_d^P \frac{s}{1+\frac{s}{N}} v + mge_3) \end{cases} \tag{1.12}$$

Where $e_x = x - x_d$ is the position tracking error in which $x_d \in \mathbb{R}^3$ is the desired position, $\mathbf{q_e} \in \mathbb{R}^3$ and $q_e \in \mathbb{R}$ are respectively the vectorial and the scalar part of the quaternion error $\hat{q}_e \in \mathbb{S}^3$. This latter is computed as the Hamiltonian product between the desired quaternion $q_d \in \mathbb{S}^3$ and the conjugate of the measured quaternion $q \in \mathbb{S}^3$, for example $\hat{q}_e := q_d \otimes q*$ [9]. Furthermore, $K_{po}^A \in \mathbb{R}^{3\times3}$ is the proportional gain of the outer loop, and $K_{ff}^A \in \mathbb{R}^{3\times3}$, $K_{pi}^A \in \mathbb{R}^{3\times3}$, $K_i^A \in \mathbb{R}^{3\times3}$ and $K_d^A \in \mathbb{R}^{3\times3}$ are respectively the feedforward, the proportional, the integral, and the derivative gains of the inner loop. Since in the next sections a linear model will be derived and it will not work with quaternions but instead with roll, pitch and yaw angles, as showed in figure 1.5, it should be noted that the first equation in (1.12) can be also linearised to retrieve the Euler angles:

$$\omega_v = -K_{po}^A sign(q_e)\mathbf{q_e} \approx -\frac{1}{2} K_{po}^A \alpha_x \tag{1.13}$$

where $\alpha_x = \alpha - \alpha_d$ is the attitude tracking error in which $\alpha_d \in \mathbb{R}^3$ is the vector of the desired Euler angles.

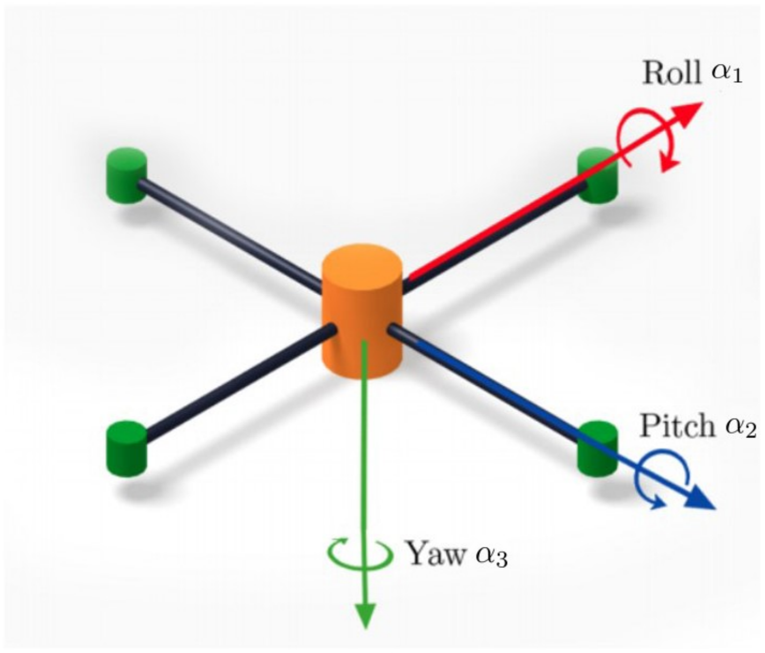

Figure 1.5: UAV Euler angles [1]

1.4 Tilt-arm quadrotor simulator

Starting from the dynamic model described in equation (1.11), the numerical implementation for simulation purposes of the tilt-arm model is presented. This model is an update version of the one presented in [1]. The simulation environment used is *Simulink* and Matlab [10] by MathWorks. All the physical data and the modelling of the actuators dynamic have been taken with the goal to reconstruct in a numerical environment an existent tiltrotor prototype built in the ASCL laboratory of the Politecnico of Milan.

- Tiltrotor: this block contains all the dynamic equations presented in (1.11). It receives as input the thrusts requested to the four motors in %, and the tilting angles of the four arms β_{ai} for $i = 1, 2, 3, 4$. The outputs of this block are the position, the velocity, the attitude and the angular rates vectors.

- Controller: it is the key block of this thesis. Different non linear control strategies have been implemented, all receive as inputs the measurements and the set points, and produce as outputs the eight control variables. As it will be showed in the next sections, this block contains other two blocks representing respectively the baseline and the adaptive controller.

- State Filter: this block reads the output of the tilt-rotor block and transforms the signals by discretizing them. This is a way to take into account

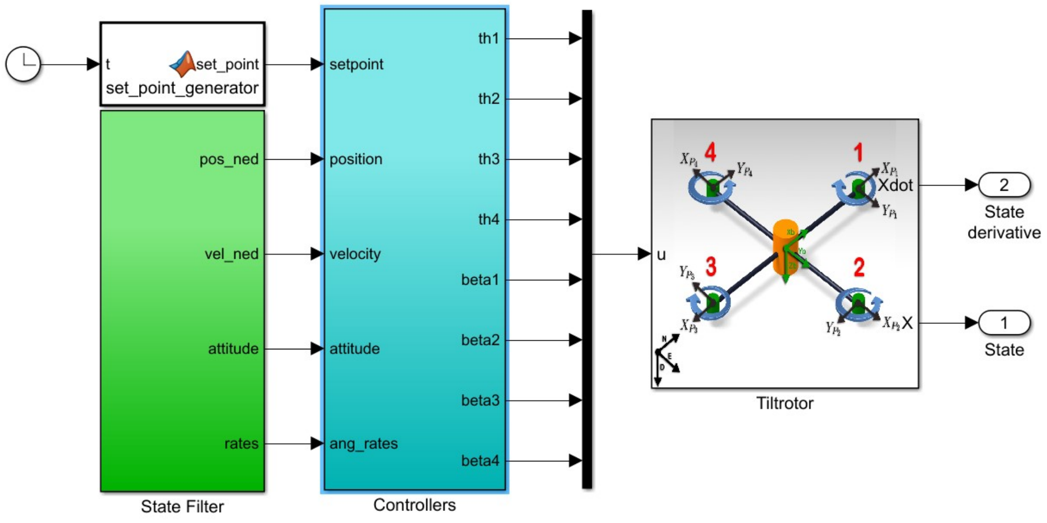

Figure 1.6: Main blocks composing the numerical model

the sampling time of an hypothetical hardware on which the control system should be implemented. The considered working frequency is 250Hz.

- Setpoint Generator: this block generates the trajectory that the tilt-rotor is supposed to follow. The set-point specifies not only the required position and the attitude in function of time, but also velocity, acceleration, angular rates and angular accelerations. Fortunately, the control strategies adopted require only the first twos. The trajectories are approximated with a fifth order polynomial in the position as variable.

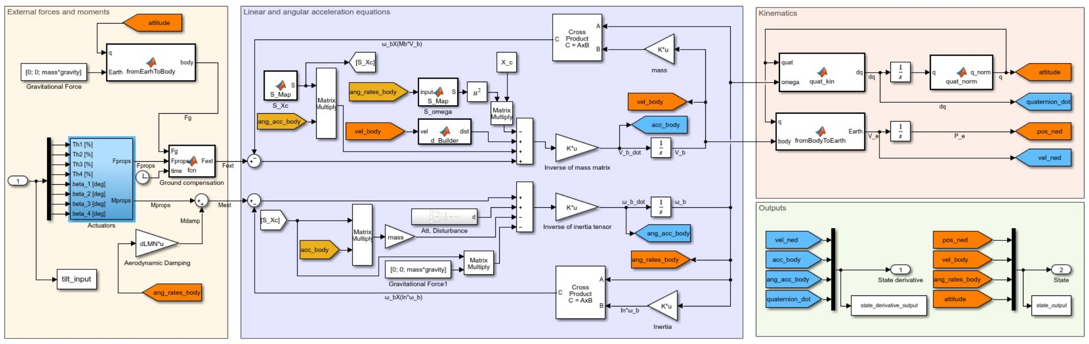

Figure 1.7: Tilt-arm Simulink dynamic model

Figure 1.6 shows the implementation in Matlab Simulink of the dynamic model described in equation (1.11). The actuators block contains the dynamic equations of motors and servo-motors whose parameters have been identified in laboratory.

It computes the inverse of the allocation procedure (mixing) showed in section 1.1.1. The left rectangle contains forces and moments acting on the system in body frame: gravity force, propellers forces, propellers moments and aerodynamic damping. The central section contains the integrators of the non linear dynamic equations counting for gyroscope effect, UAV inertia, center of mass position and disturbances. In the right upper rectangle the frames transformations are executed and then the kinematic equations are integrated. Finally, in the right lower rectangle, the output of the model simulation is produced.

This numerical simulator has been used in this work to validate the designed control architecture under study. Other simplified simulators, which will be described later, have been developed in order to make a first selection of the most suitable control strategy for this particular system.

1.5 Limitations of existing control designs

In this section, the CPID controller will be analysed more in detail. In fact, since it has shown results comparable with others currently solutions [2] and it is demonstrated a simple and intuitive approach, the CPID has been chosen as the architecture with respect to which the adaptive systems implemented in this thesis will be compared. Moreover, as can be seen in the next chapters, an its slight modification will be chosen as baseline controller for the adaptive structure.

Concerning the controller limitations, an important aspect is that [2] and [1] do not consider the fact that the center of mass position is unknown and not coincident with the body reference frame origin. In fact, it is almost impossible to place all the UAV components in such a way that $x_c = [0, 0, 0]$. To better understand this concept, in this section some numerical results will be showed. More specifically, the outcomes computed with the Simulink plant showed in figure 1.7 will be illustrated with the CPID approach and with neglected d_T and d_F. As illustrated in figure 1.8, the desired set points for the simulations have been chosen as a simple step on the first and third position component while maintaining null Euler angles for all the manoeuvre.

Figure 1.9, 1.10, 1.11, 1.12, 1.13 and 1.14 show respectively the attitude, the angular velocity, the position and the translational velocity error, along with the control input forces and torques. The results obtained can be compared with $x_c = [0, 0, 0]^T$, reported on top of each figure, and $x_c = [0.01, 0.01, 0.01]^T$, revealed at the bottom.

It is possible to notice that in general the error given from the simulations with $x_c = [0.01, 0.01, 0.01]$ m is bigger than the ones with $x_c = [0, 0, 0]$, especially at the beginning and during the step changes. This effect is particularly relevant in the attitude where the angular velocity and the Euler angles show the biggest errors. Instead, the position and the translational velocity present some differences only in the first 5 seconds. A similar analysis can be carried out for the control

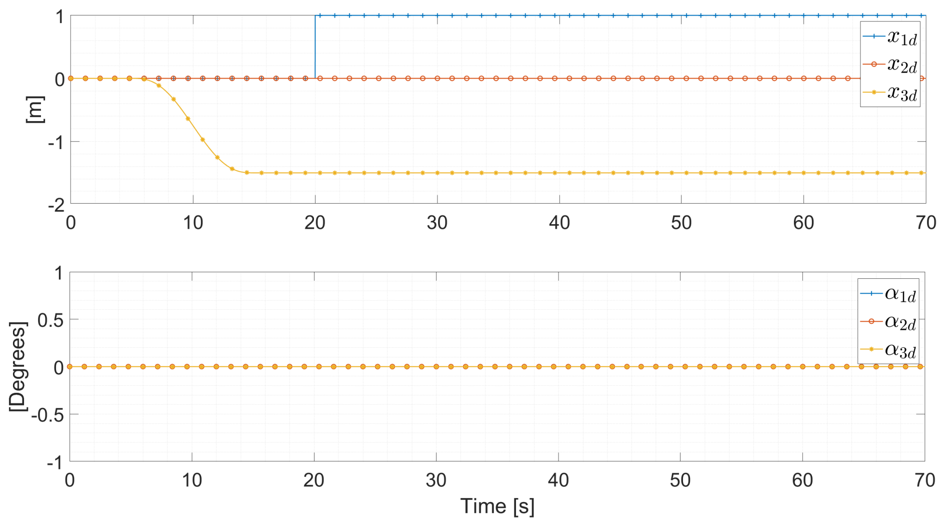

Figure 1.8: Desired position (on top) and attitude angles (bottom)

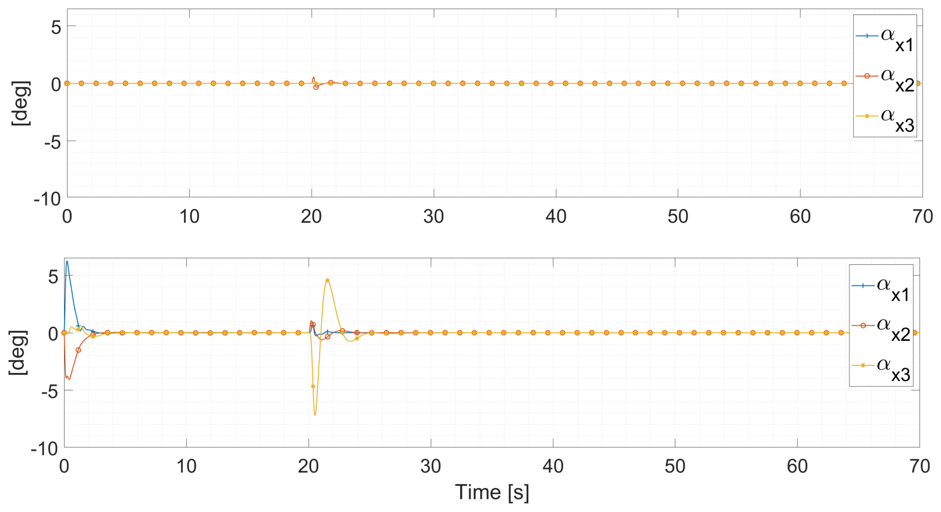

Figure 1.9: Attitude error with respect to the desired Euler angles for the simulator with $x_c = [0,0,0]^T$ (on top) and $x_c = [0.01, 0.01, 0.01]^T$ (bottom)

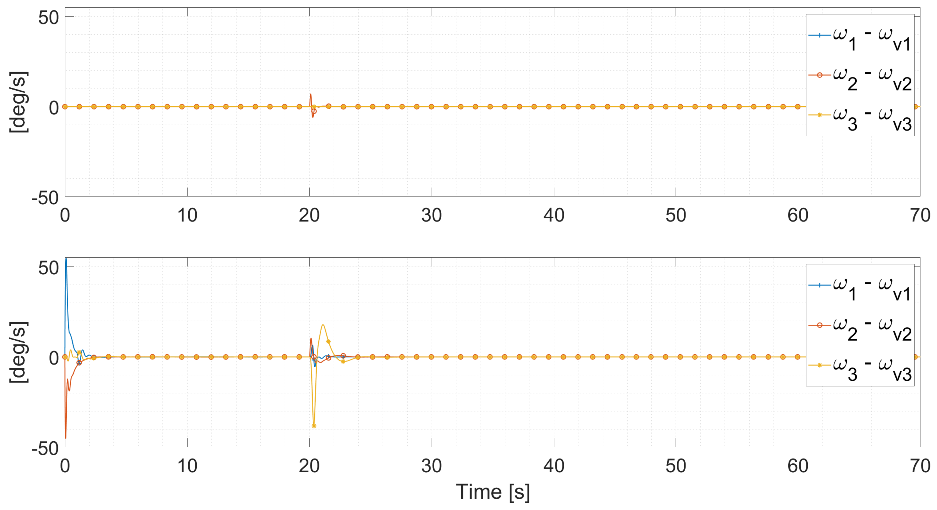

Figure 1.10: Angular velocity error with respect to the outer loop ω_v for the simulator with $x_c = [0, 0, 0]^T$ (on top) and $x_c = [0.01, 0.01, 0.01]^T$ (bottom)

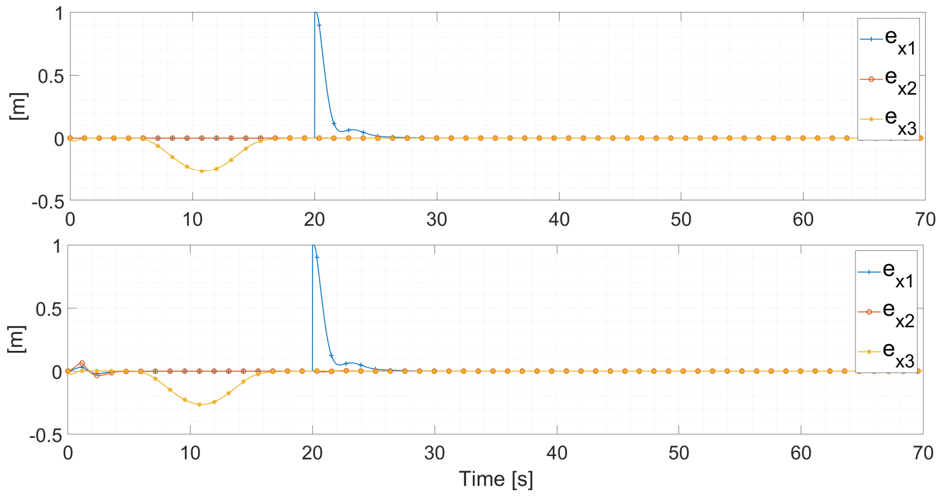

Figure 1.11: Position error with respect to the desired set points for the simulator with $x_c = [0, 0, 0]^T$ (on top) and $x_c = [0.01, 0.01, 0.01]^T$ (bottom)

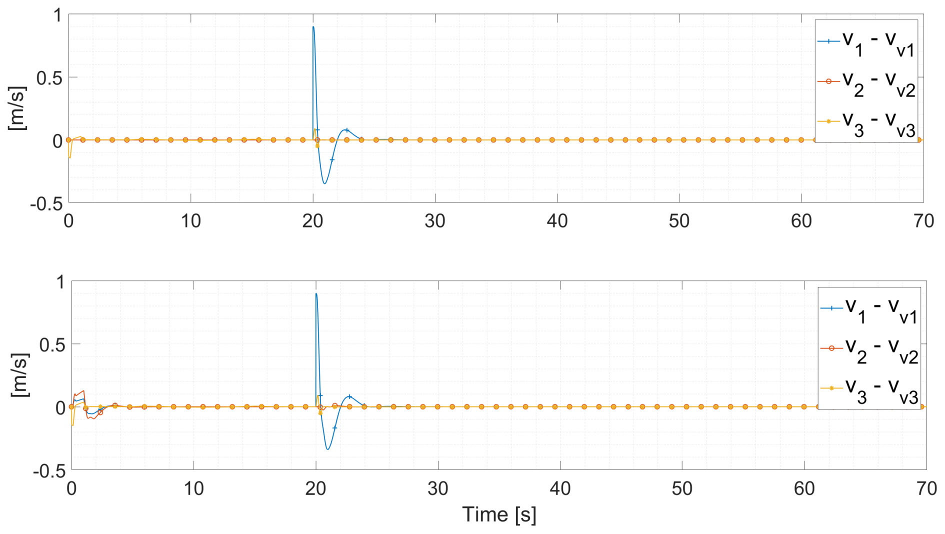

Figure 1.12: Velocity error with respect to the outer loop v_v for the simulator with $x_c = [0, 0, 0]^T$ (on top) and $x_c = [0.01, 0.01, 0.01]^T$ (bottom)

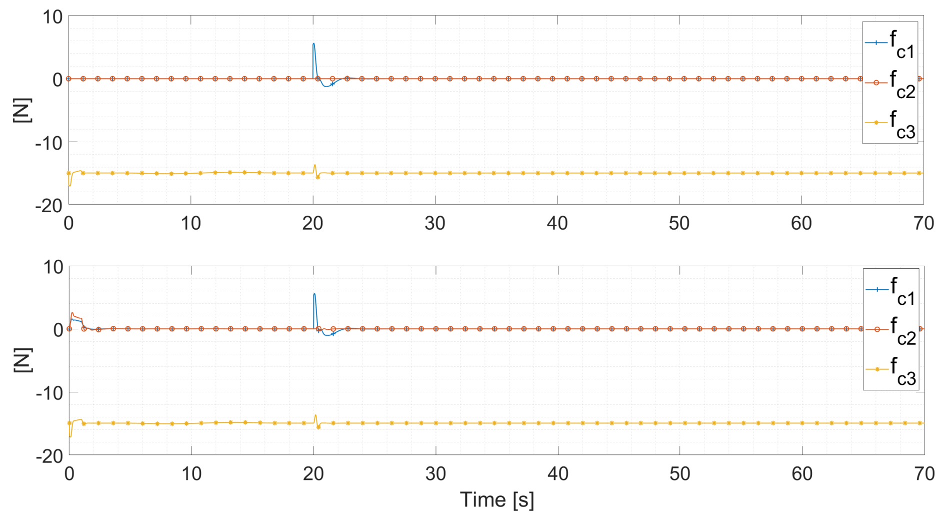

Figure 1.13: Input control force f_c for the simulator with $x_c = [0, 0, 0]^T$ (on top) and $x_c = [0.01, 0.01, 0.01]^T$ (bottom)

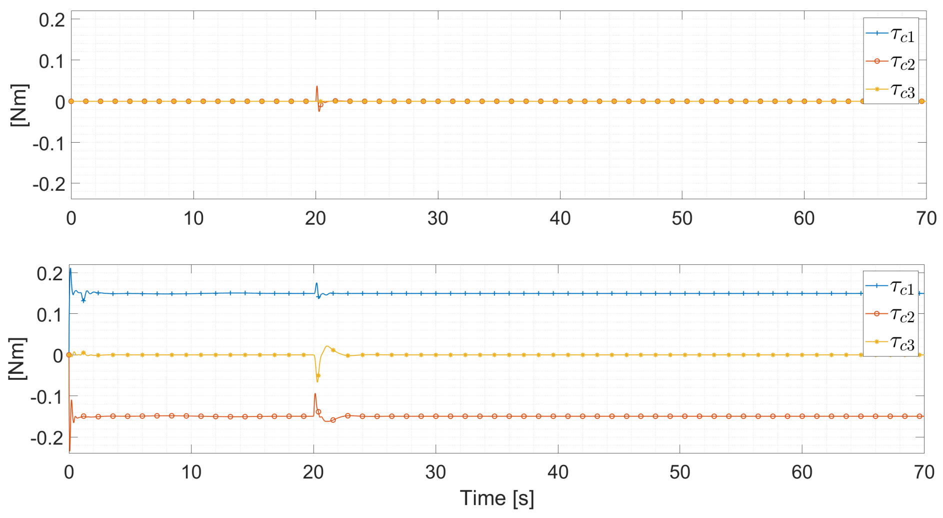

Figure 1.14: Input control torque τ_c

inputs. The control force, which affects the position dynamic, does not show large discrepancies, while the control toque, which influences the attitude dynamic, presents very different behaviours. Moreover, one should keep in mind that the center of mass position is only one of the uncertainties that are present on a tiltable propellers quadrotor. In addition, if we also consider the fact that, for example, it is very difficult to have an exact estimation of the inertia tensor and the UAV can be subjected to unknown external disturbances or not predictable events, it is evident that a CPID controller should not be satisfactory. Therefore, it should be clear that an adaptive controller is needed in order to counteract all these uncertain actions.

Continuing with the CPID limitations, by looking at equation (1.12) one can notice that the use of the sign function to avoid unwinding may result in chattering due to noise. In the paper [2], the controller is mainly designed for stabilization rather than a tracking task, so the performances for time varying set points cannot be satisfactory. As the previously seen control architectures, also the CPID depends heavily on the tuning. Techniques such as the structured H_∞ can be used but they are complex due to the large amount of parameters. Moreover, its design relies on the assumption of negligible gyroscopic effect and other stringent assumptions which seriously affect the performance especially in aggressive tasks.

1.6 Conclusion

In this chapter, the complete non linear mathematical model of the tiltable propeller UAV has been presented. A full non linear simulator, accounting for actuator dynamics and disturbances, has been implemented in Simulink and tested using a quaternion-based PID controller, which is a good compromise between simplicity of implementation and performance. A simplified dynamical model capturing all the relevant complexities of the considered plant has been developed for control design purposes and it will be exploited in the forthcoming chapters.

As shown in a numerical example, one of the major limitation of current state-of-the-art control designs lies in the fact that cross-couplings between attitude and position are neglected. In the next chapters, a solution to the main problems associated with existing designs will be provided by exploiting the tools of adaptive control.

Chapter 2

Adaptive Controllers Introduction

In this chapter, after a general background on adaptive control, different strategies will be discussed together with their advantages and limitations. In particular, this chapter is focused on model reference adaptive control (MRAC) and on closed-loop model reference adaptive control (CMRAC). A more broad review about adaptive control theory can be found in [11]. Each of them will be analysed and discussed trying to give an overall idea of their working principle and presenting the respective strategies used in this thesis.

2.1 Brief introduction to adaptive control

The tiltable propeller UAV dynamic is affected by both parametric uncertainties and unmodelled dynamics. Classical controllers similar to the ones previously analysed cannot be very suitable for this particular problem. Due to the high non linearity of the system, the lack of knowledge of certain variables makes the problem of controlling the tiltrotor difficult. According to Cambridge Dictionary, the verb "to adapt" means "to change something to fit a different use or situation". The first researches on adaptive control started in the early 1950s focusing on the design of autopilots for high-performance aircraft. In general, the idea behind the adaptive controller lies on the intuition that the output response $y(t)$ of the system carries information about the model state and parameters. In theory, one can process the input $u(t)$ and the output $y(t)$ in order to learn about parameter changes that can be exploited for tuning certain controller gains. As an example, a robust controller such as the classical Proportional Integral Derivative (PID) is generally designed to operate under the worst-case conditions, consequently it may use excessive actions to regulate the process. On the contrary, an adaptive controller primarily tries to estimate the process uncertainties on-line and then to produce a control input to anticipate, overcome or minimize the undesirable deviations from the prescribed closed-loop plant behaviour. Needless to say nothing is free: due to their more complex structure, an adaptive controller needs a more powerful and faster hardware to overcome the increased computational effort.

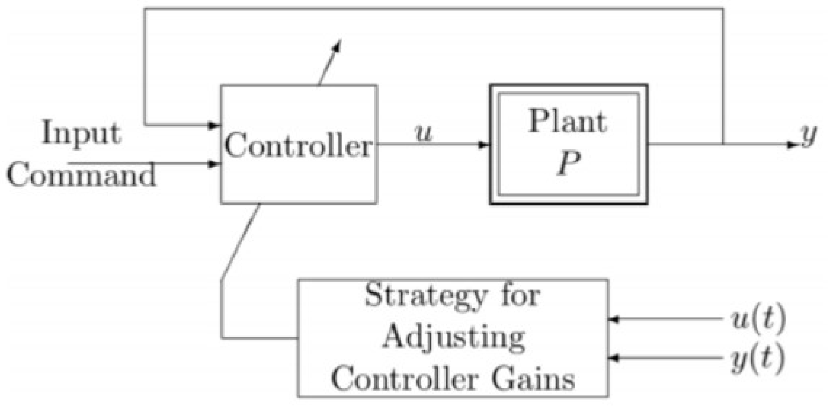

Figure 2.1: General structure of an adaptive controller [3]

To be more precise, referring to figure 2.1, as an example one can think to take a plant model in the form of equation (2.1):

$$\begin{cases} \dot{x} = A_i x + B_i u \\ y = C_i x + D_i u \end{cases} \tag{2.1}$$

where in this case A_i, B_i, C_i and D_i are functions of some operating conditions i. Therefore, while the system is operating, it can end up to different settings that change the above matrices. An adaptive controller should be able, in theory, to adjust its gains by simply extrapolating the necessary information from the input and output of the plant. In this way, it can accommodate itself even in the case of a non perfectly modelled dynamic system. In reality, recent results in adaptive control have shown that good performances can be achieved by combining a robust controller with an adaptive one maintaining closed-loop stability, enforcing robustness to uncertainties, and delivering the desired performance in the presence of unanticipated events.

2.1.1 Model Reference Adaptive Control (MRAC)

Referring to figure 2.2, it is possible to see that the main difference with figure 2.1 is the addition of an another block named reference model. This latter is a new dynamic system, which takes the same reference inputs of the plant that describes the desired behaviour that we want the plant to follow. In other words, it is simply a set of differential equations that are integrated on-line, in order to recover the ideal response with respect to which the output of the actual plant has to be compared. The resulted error is finally used to adjust the controller gains through an adaptive law. In general, the adaptive controller also takes the set point for a better estimation of the uncertain parameters.

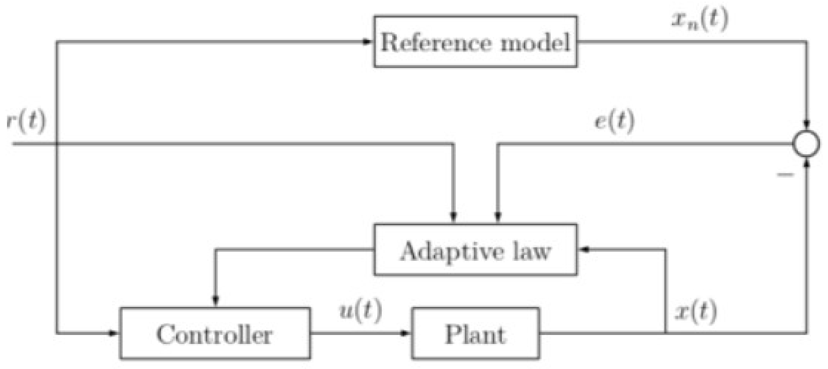

Figure 2.2: Structure of an MRAC control system [3]

More in detail, as showed in [4], it is possible to consider for example the following dynamical model representing the mathematical translation of a real physical system:

$$\dot{x} = Ax + B\Lambda(u(t) + f(x)) \tag{2.2}$$

where $x \in \mathbb{R}^n$ is the system state vector, $A \in \mathbb{R}^{n\times n}$ is a known Hurwitz matrix specifying the desired closed-loop dynamic, $u \in \mathbb{R}^{n_u}$ is the control input, $B \in \mathbb{R}^{n\times n_u}$ is the known control matrix and $\Lambda \in \mathbb{R}^{n_u\times n_u}$ is an unknown invertible matrix representing the uncertainty in the system. The uncertainty is introduced to model control failures or modelling errors, in the sense that there may exist uncertain control gains or the designer may have incorrectly estimated the system control effectiveness. The result is that, in contrast to the usual state space models, also the modelling of the uncertain parameters have been introduced directly in the dynamic system. Finally, $f(x) : \mathbb{R}^n \to \mathbb{R}^{n_u}$ represents the system matched uncertainty. It is assumed that each individual component $f_i(x)$ can be written as a linear combination of N known locally Lipschitz-continuous basis functions $\psi_i(x)$, $i = 1, \cdots, N$ with unknown constant coefficients, as showed in the following equation:

$$f(x) = \theta^T\psi(x), \quad \theta \in \mathbb{R}^{n\times N} \quad \text{where} \quad \psi(x) = \begin{bmatrix} \psi_1(x) \\ \vdots \\ \psi_N(x) \end{bmatrix}. \tag{2.3}$$

Therefore, assuming that the pair $(A, B\Lambda)$ is controllable, equation (2.2) together with (2.3) represent the common form of many dynamic systems in which an adaptive algorithm is implemented. The control objective is to design u such that the output of the system $y = Cx \in \mathbb{R}^{n_y}$ tracks any bounded possibly time-varying command $r(t) \in \mathbb{R}_{n_y}$ with the presence of system uncertainties. In particular, the state should converge to the state generated by the reference model that represents the desired system behaviour. A MRAC generic reference model is usually described as equation (2.4):

$$\dot{x}_m = A_m x_m + B K_r r(t) \tag{2.4}$$

where $A_m \in \mathbb{R}^{n \times n}$ represents the desired closed-loop behaviour and $K_r \in \mathbb{R}^{n_u \times n}$ is a constant feed-forward gain. In order to have the plant following the reference model, the control input has to be chosen such that it ensures the convergence of $e = \|x(t) - x_m(t)\|_\infty$ and that this latter is sufficiently small in order to satisfy the tracking problem.

The MRAC architecture has the advantage to be easy to implement and, with a suitable choice of the control input, it ensures closed-loop stability. At the same time, in the literature there is a lack of guidelines to select an adequate tuning that represents the MRAC major challenge. For this reason, the tuning phase is usually done by simply trial and error procedures or expensive Monte Carlo simulations. Moreover, this approach does not give the possibility to the designer to adjust the controller in order to have a desired transient behaviour which, in turn, typically shows an oscillatory behaviour.

2.1.2 Closed-Loop Model Reference Adaptive Control (CMRAC)

As previously mentioned, one of the major problems of the MRAC architecture is the tuning phase. Also in the literature [12], it is clear that the designer has to find a trade-off between the speed of convergence and the presence of very fast oscillations in the transients. In general, we want that the system tracks the reference model fast so that the transient dynamics extinguish as soon as possible. This involves high adaptation gains that in turn generate high frequency oscillations during the transients. Therefore, one should find the right compromise because a too fast controller can lead to dangerous chattering phenomena.

Historically, the Model Reference Adaptive Controller has been the one that was developed first and has been open-loop in nature. More precisely, the reference model is not influenced by the plant itself but it simply takes as input only the set point, as shown in figure 2.2. Therefore, an idea to solve the oscillations problem can be the one of finding a way to tell to the reference model how much it is deviating from the plant. The concept is very similar to a classical state observer, where a term equivalent to the output innovation feedback in a state observer is added to the reference model, as shown in equation (2.5) and illustrated in 2.3:

$$\dot{x}_m = A_m x_m + B K_r r(t) + K_e e. \tag{2.5}$$

In the above equation, $K_e \in \mathbb{R}^{n \times n}$ is the reference model feedback gain and $e = x - x_m$. This improved architecture, as showed in [12], has demonstrated to

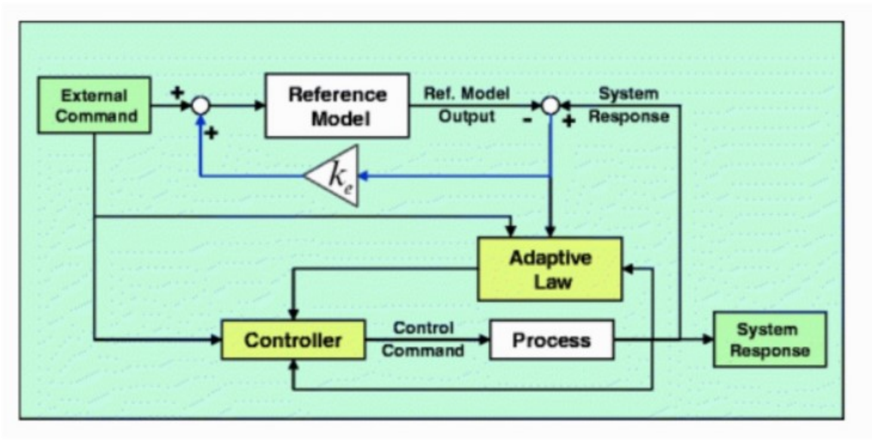

Figure 2.3: Structure of an CMRAC control system [4]

offer a decreased tracking error L_2 norm that is strictly related to the frequency characteristics of the signal and consequently to the number of oscillations. By tuning K_e, the MRAC transient dynamic can be forced to decay as fast as needed. It is an additional degree of freedom that, together with the adaptation gain, can be used to ensure fast tracking convergence and good transient performances.

Finally, as can be seen, the overall CMRAC architecture is equal to the one of the MRAC but with the exception of the new innovation term entering in the reference model. For this reason, as the MRAC, the limitation of how tuning the gains is still present, even if some guidelines exist [12]. The performance of this adaptive scheme is strongly dependent on the adaptation gain that cannot be too large for robustness reasons and furthermore, it is not so easy to understand what too large means.

2.2 Control architecture chosen

The previous twos architectures are two general approaches largely used in the adaptive control theory that have been presented only to show the specific background and their main peculiarities. In fact, it was mentioned in section 2.1 that the best performances can be achieved by integrating a baseline controller together with the adaptive one. In figure 2.4, the MRAC structure of the control system used in this work is reported. The same structure can be also used for the implementation of the CMRAC by simply adding the error term as input to the reference model as showed in figure 2.5. In particular, it is possible to notice that there are a total of three controller blocks. Two baselines and one adaptive. By definition, the reference model is made by a baseline controller plus the reference plant that generates the ideal output. The two baseline controllers take as input the reference signal and the plant state, and compute the baseline control input u_b, respectively for the reference dynamic system and for the plant. It should be

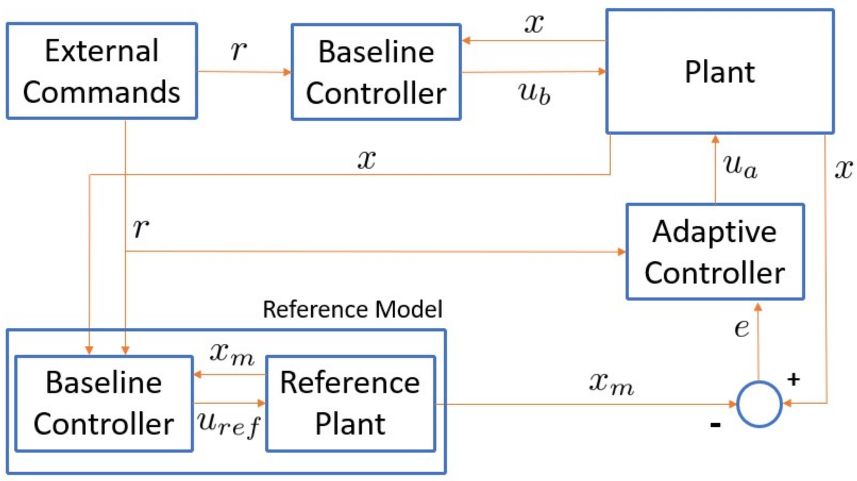

Figure 2.4: MRAC Control system architecture used in this work

noted that these two controllers are identical. Then, the error between the simulated reference model and the plant, along with the set points, are used by the adaptive controller to compute the adaptive control input u_a. In this architecture, u_b brings the system to the correct desired point and in the meanwhile, u_a tries to anticipate, overcome or minimize all the non predictable actions. For this reason, considering the plant presented in (1.11), the control input force and torque f_c and τ_c have been split in two different contributions to highlight the decoupling between the baseline controller and the adaptive one. More precisely:

$$\begin{cases} \tau_c = \tau_c^b + \tau_c^a \\ f_c = f_c^b + f_c^a \end{cases} \tag{2.6}$$

where $\tau_c^b \in \mathbb{R}^3$ and $\tau_c^a \in \mathbb{R}^3$ are respectively the baseline and the adaptive control torque inputs, while $f_c^b \in \mathbb{R}^3$ and $f_c^a \in \mathbb{R}^3$ are respectively the baseline and adaptive control force inputs. In fact, in general in a PID architecture the integral term accelerates the movement of the process towards set point and it eliminates the residual steady-state error that occurs with a pure proportional controller. Instead, the derivative action predicts system behaviour and thus improves settling time and stability of the system. In few words, both of them ensure that the system tracks the desired set point also in presence of disturbances. Therefore, the idea is to use an adaptive controller in place of the integral and derivative terms and to maintain the proportional action only for getting the system close to the desired signal. In fact, as previously discussed a proportional controller is capable to converge to zero steady error only in the ideal case with a perfectly modelled system and without disturbances, but in reality it needs additional terms able to reject all the uncertainties and helping the proportional term to annul the tracking error.

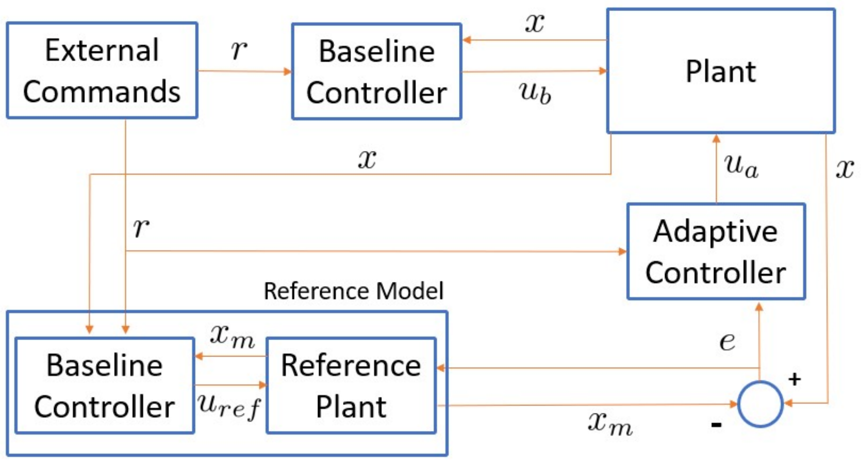

Figure 2.5: CMRAC Control system architecture used in this work

As we will see more in detail in the next sections, the adaptive control input has to be designed in such a way that it annuls the plant uncertain terms. For example, referring to the plant in equation (2.2), one can choose $u(t) = -\hat{\theta}^T\psi(x)$ where $\hat{\theta}$ is the estimated value of θ which has to be computed on-line through an adaptive law. This latter is simply an estimation algorithm that exploits the information contained in the error between the reference model and the plant together with the tracking set point, to compute an estimation of the uncertain parameters. Therefore, it should be clear that the choice of the adaptive law is very important. In fact, as additional remark, since the adaptive controllers suffer of some instability phenomena that can be potentially dangerous for the system, a suitable choice of the adaptive law can solve or, at least, minimize them.

2.3 Instability phenomena and robustness modifications

In general, the adaptive controllers are characterized by different robustness problems that are well known in the literature and that they can be solved by a suitable choice of the adaptive law which, in turn, has to ensures the correct estimation of the uncertain parameters and robustness of the system. The main instability phenomena describing this kind of controllers are five, as well explained in [3]:

1. Parameters drift
2. High-Gain instability
3. Instability resulting from fast adaptation
4. High-Frequency instability

5. Effect of parameter variations

The first twos are in some way correlated, in the sense that they both depend on the fact that, for a wrong choice of the adaptive law, the uncertain parameters can drift to infinite in order to completely counteract the disturbances and bringing the tracking error to zero. Furthermore, this drift can create a very high feedback gain which excites the unmodelled dynamics and leads to instability and unbounded solutions.

These phenomena are usually present when the only objective is the regulation of the system. Instead, if we want also to track a certain reference trajectory, it is possible to incur in fast adaptation and high frequency instability. More precisely, in theory some adaptive laws can contrast the disturbances without drifting but with very slow transients. This naturally leads to large adaptation gains for increasing the speed of convergence which, however, excites unmodelled dynamics causing instability.

Instead, the forth phenomena usually happens when the reference input signal has frequencies in the parasitic range that can excite the not modelled dynamics, cause the signal-to-noise ratio to be small and, therefore, lead to the wrong adjustment of the parameters and eventually to instability.

Finally, the last point concerns the fact that, historically, many adaptive controllers were designed for LTI systems since they demonstrated to perform well also for plants with parameters varying slowing with time. Conversely, for complete linear time varying plants this could not be completely true and therefore, the adaptive law has to be modified to meet the control objective in the case of parameter variations, typically encountered in high non linear plants.

2.3.1 Dead-zone modification

In order to enforce robustness one possible strategy could be the one of considering an adaptive law with the so called dead-zone modification showed in equation (2.7) and reported in [13]:

$$\dot{\hat{\theta}} = \begin{cases} -\Gamma\psi(x)e^T PB & \text{if} \|e\| > e_0 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

Where $\Gamma \in \mathbb{R}^{n_\theta \times n_\theta}$ is the adaptation gain and P is the solution of the Lyapanov equation $A^T P + PA = -Q$ for some $Q = Q^T > 0$. The idea is to stop the adaptation process when the norm of the tracking error becomes smaller than a prescribed value e_0 avoiding the parameters to drift and resetting the controller. e_0 can be determined with different methods, usually based on the known of the maximum disturbance amplitude, which are not reported here. Nonetheless, the main point is that the dead-zone modification alone is not able to ensure asymptomatically stability of the tracking error since in certain moments is switched off. Moreover, this modification is not Lipschitz and therefore it can cause chattering and other undesirable effects when $\|e\| \approx e_0$. On the contrary, it has the advantage to prevent the uncertain parameters from drifting to infinite.

2.3.2 σ - modification

The dead-zone modification previously presented needs the knowledge of the disturbance upper bound that is not so easy to identify. A possible solution to this problem is the σ-modification developed by Ioannou and Kokotovic in [14]. In this case the adaptive law has been modified as follow:

$$\dot{\hat{\theta}} = -\Gamma \left(\psi(x) e^T PB + \sigma \hat{\theta} \right), \quad \sigma > 0 \tag{2.8}$$

where σ is a strictly positive constant. In few words, this modification adds damping to the ideal adaptive law and forces all the signals to remain uniformly bounded. The drawback is that the parameter error will not converge to zero even in case of null disturbances. Furthermore, the modification works like a forgetting factor when $\|e\| \approx 0$. In fact, in that case the adaptive law becomes:

$$\dot{\hat{\theta}} \approx -\Gamma \sigma \hat{\theta}$$

in which the adaptive parameter unlearns all the previously information and converges to zero, although this is not the ideal value. Finally, the choice of σ is not easy and if not chosen correctly may worsens the tracking performances.

2.3.3 e - modification

In order to overcome the drawback of σ-modification, Narendra and Annaswamy in [15] developed the e-modification. The idea is to replace the constant damping gain σ in (2.8) with a term proportional to a linear combination of the system tracking errors, such as $\|e^T PB\|$. The rational for using an error-dependent damping is that it tends to zero as the regulated output error diminishes. The adaptive law becomes:

$$\dot{\hat{\theta}} = -\Gamma \left(\psi(x) e^T PB + \sigma \|e^T PB\| \hat{\theta} \right), \quad \sigma > 0 \tag{2.9}$$

where σ is again a design parameter. With this modification, when the disturbances are zero, the origin becomes an equilibrium of the system. Consequently, unlike the σ-modification, the error $e(t)$ can converge to and remain zero. Moreover, the parameters can converge to their ideal values. As before, a limitation is that the choice of the design parameter σ is not trivial.

2.3.4 The projection operator

All the strategies presented until now show that many adaptive laws cannot be robust to bounded disturbances, no matter how small they are. Therefore, the adaptive law has to be chosen carefully in such a way that it can enforce robustness in the presence of unmatched disturbances as, for example, a bounded process noise. A well known solution to this problem can be found by the use of the *Projection Operator*. Before introducing it and following [5], some definitions are needed in order to understand the mathematical background.

Definition 1: A set $E \subset \mathbb{R}^k$ is convex if

$$\lambda x + (1-\lambda)y \in E$$

whenever $x \in E, y \in E$, and $0 \leq \lambda \leq 1$.

Definition 2: A function $f : \mathbb{R}^k \to \mathbb{R}$ is convex if

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y) \quad \text{with} \quad 0 \leq \lambda \leq 1$$

Lemma 3: Let $f(\theta) : \mathbb{R}^k \to \mathbb{R}$ be a convex function. then for any constant $\delta > 0$

the subset $\Omega_\delta = \{\theta \in \mathbb{R}^k | f(\theta) \leq \delta\}$

Lemma 4: Let $f(\theta) : \mathbb{R}^k \to \mathbb{R}$ be a continuously differentiable convex function. Choose a constant $\delta > 0$ and consider $\Omega_\delta = \{\theta \in \mathbb{R}^k | f(\theta) \leq \delta\} \subset \mathbb{R}$. Let θ^* be an interior point of Ω_δ, i.e. $f(\theta^*) < \delta$. Choose θ_b as a boundary point so that $f(\theta_b) = \delta$. Then the following holds:

$$(\theta^* - \theta)^T \nabla f(\theta_b) \leq 0$$

where $\nabla f(\theta_b) = \left(\frac{\partial f(\theta)}{\partial \theta_1} \cdots \frac{\partial f(\theta)}{\partial \theta_k}\right)$ evaluated at θ_b

The Projection Operator for two vectors θ, $y \in \mathbb{R}^k$ can now be introduced as:
Definition 5:

$$\mathit{Proj}(\theta, y, f) = \begin{cases} y - \frac{\nabla f(\theta)(\nabla f(\theta))^T}{\|\nabla f(\theta)\|^2} y f(\theta) & \text{if} \quad f(\theta) > 0 \wedge y^T \nabla f(\theta) > 0 \\ y & \text{otherwise} \end{cases} \tag{2.10}$$

Where $f : \mathbb{R}^k \to \mathbb{R}$ is a convex function and $\nabla f(\theta) = \left(\frac{\partial f(\theta)}{\partial \theta_1} \cdots \frac{\partial f(\theta)}{\partial \theta_k}\right)$

In order to better understand how the Projection Operator works, it is reported below an its geometrical interpretation. In fact, it is possible to define a convex set Ω_0 and Ω_1 as:

$$\Omega_0 := \{\theta \in \mathbb{R}^k | f(\theta) \leq 0\} \qquad \Omega_1 := \{\theta \in \mathbb{R}^k | f(\theta) \leq 1\}$$

From the definition of equation (2.10), θ is not modified when $\theta \in \Omega_0$. So, we can now introduce the annulus region:

$$\Omega_A := \Omega_1 \setminus \Omega_0 = \{\theta \quad | \quad 0 < f(\theta) \leq 1\}$$

Inside Ω_A the projection algorithm subtracts a scaled component of y that is normal to the boundary $\{\theta \quad | \quad f(\theta) = \lambda\}$. When $\lambda = 0$, the scaled normal component is 0, and when $\lambda = 1$, the component of y that is normal to the boundary Ω_1 is entirely subtracted from y, so that $Proj(\theta, t, f)$ is tangent to the boundary $\{\theta \quad | \quad f(\theta) = 1\}$. This discussion can be visualized in figure 2.6.

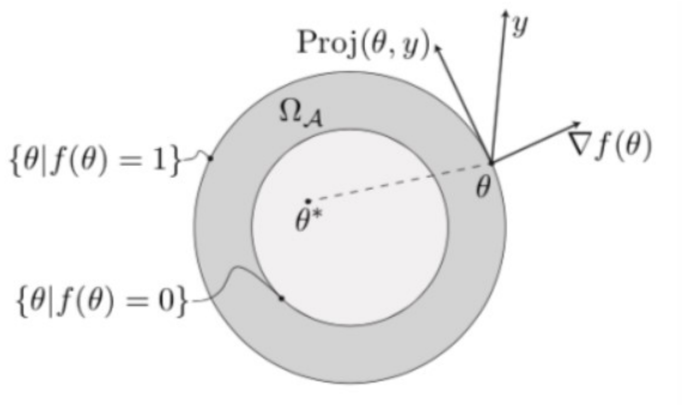

Figure 2.6: Geometrical interpretation of the Projection Operator in $\mathbb{R}^2$ [5]

Finally, after all the derivations, the Projection Operator can be now applied on the adaptive controller exploiting its properties in a suitable adaptive law. In particular, the uncertain parameters can be estimated as:

$$\dot{\hat{\theta}} = Proj(\hat{\theta}, y, f) \tag{2.11}$$

Taking $\theta(t = 0) = \theta_0 \in \Omega_1 = \{\theta \in \mathbb{R}^k | f(\theta) \leq 1\}$ and $f(\theta) : \mathbb{R}^k \rightarrow \mathbb{R}$ as a convex function. In this way we can be sure that $\theta(t) \in \Omega_1$, $\forall t \geqslant 0$.

Since an adaptive system is also characterized by the presence of non-linear integrators, such as the one showed in equation (2.11) which output constitutes the adaptive parameters, it can happen that the integrators saturate or give a too big response when a large change in the set point occurs. This phenomenon is known as integral *windup* and it can be prevented with special control techniques such as the Projection Operator which is, precisely, an *anti-windup* method.

2.4 Conclusion

Many different control architectures have been developed for the tiltrotor but without obtaining very satisfactory results. For this reason, a more complex controller capable to adapt itself can give good hopes in the performances improvement. Referring to the literature, it seems that the MRAC architecture offers more disadvantages than the CMRAC that suggests to be an its improved version. In reality, as shown in the previous section, in this thesis a modification of the latter twos approaches will be adopted. Therefore, since it is difficult to say *a priori* which architecture performs better, it has been chosen to test both the MRAC and CMRAC with the above architecture, which results will be showed in the next sections.

In conclusion, for this thesis it has been chosen to use the projection operator as the most suitable and advanced adaptive law. In fact, it avoids the uncertain parameters from drifting away, it has the capability to bound the overall adaptive process, and it prevents the integrators against undesirable *windup* problems. All this ensures the convergence of the tracking error.

Chapter 3

Uncertain Model

In this chapter, starting from the dynamic system showed in equation (1.11), a linearised plant will be derived, based on motivated simplifying assumptions. This linearised plant is used to retrieve the formulation presented in equation (2.2) and, since it will be discovered that the uncertainties appear only in the dynamic equations of the system, this allows the implementation of a suitable adaptive controller on the inner loop only.

3.1 Plant linear model

As previously discussed, an adaptive controller is something capable to contrast all the non predictable events that usually affect the UAV platforms. This can be achieved by telling to the system what are the uncertain variables and how they are modelled. In other words, the adaptive controller has to know which uncertain parameters it has to estimate and how they enter in the plant. In this way, by having an idea of how the uncertainties behave, the controller better counteracts them and the performances improve. For this reason, the first step to design an adaptive controller regards the derivation of the uncertain plant.

As it is possible to see in equation (1.11), the non linear dynamic of the tiltable UAV is very complex, consequently also the derivation of the non linear uncertain plant would be particularly complicated and maybe unnecessary. Therefore, it has been chosen to linearise the model in order to simplify all the computations. The non linear terms neglected should not affect too much the performances in the operative conditions of interest. Of course there is no assurance that a linear uncertain model will work in our case, but if it works and the performances will be as expected, the advantages will be many:

- Simpler model from a computation point of view
- Faster and easier to implement
- The linear word is always better because more predictable

- Linear algebra theorems work

It is a sort of engineering compromise, obviously it is possible to derive the complete non linear uncertain model but, before doing it, it is necessary to understand if it is worth. A very complex model maybe does not lead to very big improvements and consequently also a simplified version can do its job excellently saving time.

Before going on, some assumptions have to be made to allow the derivation of the linearised model. In particular, we assume:

- $J =$ Constant as anticipated in section 1.2
- $\|x_c\| << 1$. The distance between the center of mass and O_b is close to zero.
- $S(\omega)J\omega \approx 0$. The gyroscopic term is negligible.

The last two assumptions are valid for small scale UAVs, which are the ones under study in this work. Now, assuming that x, v, R and ω are infinitesimally small, it is possible to write:

$$\begin{cases} x \sim \delta x \\ v \sim \delta v \\ R \sim I_3 + S(\delta\alpha) \\ \omega \sim \delta\omega \end{cases} \tag{3.1}$$

where the symbol δ indicates an infinitesimal variation of the variable under interest and $\alpha \in \mathbb{R}^3$ is the vector containing the roll, pitch and yaw angles. So, making the above substitution inside equation (1.11), one can write:

$$\begin{cases} \delta\dot{x} = \delta v \\ S(\delta\dot{\alpha}) = (I + S(\delta\alpha))S(\delta\omega) \\ m\delta\dot{v} - m(I + S(\delta\alpha))S(x_c)\delta\dot{\omega} = -mge_3 - m(I + S(\delta\alpha))S(\delta\omega)^2 x_c + d_F + Rf_c \\ J\delta\dot{\omega} + mS(x_c)(I + S(\delta\alpha))^T\delta\dot{v} = -mgS(x_c)(I + S(\delta\alpha))^T e_3 + d_T + \tau_c \end{cases} \tag{3.2}$$

Therefore, neglecting the second order terms and reminding that x_c is assumed to be small, it is possible to obtain the linearised tiltrotor model showed in equation (3.3).

$$\begin{cases} \delta\dot{x} = \delta v \\ \delta\dot{\alpha} = \delta\omega \\ m\delta\dot{v} = -mge_3 + d_F + Rf_c \\ J\delta\dot{\omega} = -mgS(x_c)e_3 + d_T + \tau_c \end{cases} \tag{3.3}$$

It is very important to note that, in the above retrieved linear model, the attitude dynamic is completely independent from the position one. This is totally in agreement with what we said in section 1.2: the tiltrotors are special platforms capable to independently control their attitude with respect to the position that they have to reach.

3.2 The uncertain model

3.2.1 Uncertain parameters

The first step to derive the uncertain model regards the choice of the uncertain parameters. Referring to equation (1.11), in general for small scale UAVs the four variables that are the most difficult to estimate and that affect more the dynamic of the system are:

- The tensor inertia J
- The center of mass x_c
- The disturbances d_F and d_T

Referring to the linear system just derived and considering the above uncertain variables, an important remark that has to be highlighted lies in the fact that the only two equations affected by uncertainties are the position and the attitude dynamic equations, more precisely the third one and the last one. As it will be shown later, this will be very useful for the implementation of the adaptive controller.

Going on, as previously mentioned, the idea of an adaptive controller is to tell it what we know about the physical system and let the algorithm correct all the remaining uncertainties. Under this perspective, since a very rough estimation of J and x_c is known, it is viable to write:

$$\begin{cases} \tilde{J} = J - \bar{J} \\ \tilde{x}_c = x_c - \bar{x}_c \end{cases} \tag{3.4}$$

where $\tilde{J} \in \mathbb{R}^{3\times 3}$ and $\tilde{x}_c \in \mathbb{R}^3$ are respectively the gap between the physical tensor inertia J and the estimated or baseline $\bar{J} \in \mathbb{R}^{3\times 3}$, and the gap between the real center of mass x_c and the estimated or baseline $\bar{x}_c \in \mathbb{R}^3$. More precisely, the variables with the tilde above them represent their uncertain part.

In order to be able to apply an adaptive controller to the system, it is necessary to rearrange the tiltrotor plant in a form similar to the one showed in equation (2.2), which splits the nominal contributions from the uncertain ones. In particular, since we are augmenting the system with the baseline controller and some disturbances, the uncertain model has to include them. Therefore, the final uncertain state space model will take the following structure:

$$\dot{X} = \underbrace{AX + B_r r}_{Nominal} + \underbrace{B\Lambda(W\theta + u_{ad} + d)}_{Uncertain} \tag{3.5}$$

where in the case of our problem, $X \in \mathbb{R}^{12}$ is the state vector containing the UAV position, velocity, attitude angles and angular velocities, $A \in \mathbb{R}^{12\times 12}$ and

$B_r \in \mathbb{R}^{12\times12}$ describe the desired closed-loop dynamics, $r \in \mathbb{R}^{12}$ is desired set point vector, $B \in \mathbb{R}^{12\times3}$ is a known constant matrix, $\Lambda \in \mathbb{R}^{12\times3}$ is unknown, $u_{ad} \in \mathbb{R}^{12}$ is the adaptive control input, $d \in \mathbb{R}^{12}$ is the vector of the disturbance terms, $W \in \mathbb{R}^{3\times n_\theta}$ represents the modelling of the uncertain parameters collected in the $\theta \in \mathbb{R}^{n_\theta}$ vector. It should be noted that the latter state space model is particularly useful for adaptive controller applications because of its ease of understanding. In fact, in this kind of systems, the adaptive control input u_{ad} can be simply designed in such a way that it can be opposed to the uncertain terms. For example: $u_{ad} = -W\hat{\theta} - \hat{d}$, where the hat symbol indicates that the parameters have to be estimated with suitable adaptive laws.

It is also possible to see that the disturbance vector is completely separated from the other uncertain variables that need some sort of mathematical modelling, for example the ones showed in equation (3.8). Indeed, the design of x_c and J will be decoupled from the design of the disturbance terms that are independent from the state. In other words, all the state dependent variables are used to derive the form of the B and W matrices, which represent the uncertainty in the model, while the state independent term of d will be modelled separately and treated as a constant offset.

3.2.2 Uncertain model derivation

Referring to equation (3.3), it should be clear that the moment of inertia and the center of mass enter in the attitude dynamic equation only. For this reason only the 4^{th} equation needs to be rearranged to recover the uncertain model. Instead, the other threes do not depend on J and x_c. Consequently, from the adaptive control point of view, if we do not consider the disturbances, they are considered as exact. Moreover, as mentioned in section 3.1, the attitude and translation dynamics in the linear plant are independent and therefore, it has sense to split the control design. Of course, in reality, as can be seen in the non linear system (1.11), there exist some couplings in the system and a complete separated approach should not be satisfied. At this stage, since splitting the attitude and the position design is simpler, more intuitive and avoids the use of non linear dynamics, the hope is that, at the end, this way of work will give good results demonstrating the avoidance of a coupled design.

Attitude dynamic

Considering only the attitude dynamic in the system (3.3), since the major uncertain contribution comes from x_c and J, the disturbance d_T can be assumed independent on the state. Consequently, it can be seen as a simple offset or *step* function added to the dynamic avoiding the need of further mathematical computations.

So, substituting equation (3.4) and equation (2.6) in the linear plant and taking d_T as a separated component, one can write:

$$J\delta\dot{\omega} = -mgS(\tilde{x}_c + \bar{x}_c)e_3 + \tau_c^b + \tau_c^a + d_T = -mgS(\tilde{x}_c)e_3 - mgS(\bar{x}_c)e_3 + \tau_c^b + \tau_c^a + d_T$$

Reminding that the overall idea is to split the nominal contribution from the uncertain one, it is convenient to bring J on the right hand of the above equation and to add and to subtract the nominal terms for trying to recover $\tilde{J}$.

$$\begin{aligned}\delta\dot{\omega} &= J^{-1}(-mgS(\bar{x}_c)e_3 + \tau_c^b) + J^{-1}(-mgS(\tilde{x}_c)e_3 + \tau_c^a + d_T)\\ &= \bar{J}^{-1}(-mgS(\bar{x}_c)e_3 + \tau_c^b) - \bar{J}^{-1}(-mgS(\bar{x}_c)e_3 + \tau_c^b) + J^{-1}(-mgS(\bar{x}_c)e_3 + \tau_c^b) +\\ &\quad + J^{-1}(-mgS(\tilde{x}_c)e_3 + \tau_c^a + d_T)\\ &= \bar{J}^{-1}(-mgS(\bar{x}_c)e_3 + \tau_c^b) + (J^{-1} - \bar{J}^{-1})(-mgS(\bar{x}_c)e_3 + \tau_c^b) +\\ &\quad + J^{-1}(-mgS(\tilde{x}_c)e_3 + \tau_c^a + d_T)\end{aligned} \quad . \tag{3.6}$$

But the inertia term inside the round brackets can be rearranged as:

$$J^{-1} - \bar{J}^{-1} = J^{-1}(I - J\bar{J}^{-1}) = J^{-1}(I - (\tilde{J} + \bar{J})\bar{J}^{-1}) = J^{-1}(I - I - \tilde{J}\bar{J}) = J^{-1}(-\tilde{J}\bar{J})$$

That substituting inside equation (3.6) leads to:

$$\delta\dot{\omega} = \bar{J}^{-1}\left(-mgS(\bar{x}_c)e_3 + \tau_c^b\right) + J^{-1}\left[(-\tilde{J}\bar{J}^{-1})(-mgS(\bar{x}_c)e_3 + \tau_c^b) - mgS(\tilde{x}_c)e_3 + \tau_c^a + d_T\right]$$

Since the estimated center of mass $\bar{x}_c$ is assumed to be very small, it can be neglected:

$$\delta\dot{\omega} \approx \underbrace{\bar{J}^{-1}\tau_c^b}_{Nominal} + \underbrace{J^{-1}\left[(-\tilde{J}\bar{J}^{-1})\tau_c^b - mgS(\tilde{x}_c)e_3 + \tau_c^a + d_T\right]}_{Uncertain}. \tag{3.7}$$

Equation (3.7) represents the uncertain model of the linearised attitude dynamic of the tiltrotor. As it is possible to note, the nominal contribution, in which the estimated variables such as $\bar{J}$ appear, is now completely split from the uncertain ones where, on the other hand, uncertain unknowns such as $\tilde{J}$, $\tilde{x}_c$ and d_T are present.

As previously said, since d_T is assumed to be a constant unknown variable added to the model, it is possible to define the vector of the uncertain parameters θ_a as a function of the uncertain center of mass and moment of inertia:

$$\theta_a = \left[\tilde{J}_{11}, \tilde{J}_{12}, \tilde{J}_{13}, \tilde{J}_{22}, \tilde{J}_{23}, \tilde{J}_{33}, \tilde{x}_{c1}, \tilde{x}_{c2}, \tilde{x}_{c3}\right]^T \in \mathbb{R}^9. \tag{3.8}$$

In order to recover the model showed in equation (3.5), the first two terms inside the square brackets of equation (3.7), which are the ones that depend on the state and on the uncertain parameters $\tilde{J}$ and $\tilde{x}_c$, can be finally rewritten as the product $W_a\theta_a$, where $W_a \in \mathbb{R}^{3\times 9}$ is computed as:

$$W_a = jacobian\left[(-\tilde{J}\bar{J}^{-1})\tau_c - mgS(\tilde{x}_c)e_3\right] \tag{3.9}$$

where the $jacobian$ is calculated with respect to θ_a as showed in equation (3.10):

$$jacobian = \begin{bmatrix} \dfrac{\partial f_1}{\partial \theta_1} & \cdots & \dfrac{\partial f_1}{\partial \theta_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial \theta_1} & \cdots & \dfrac{\partial f_m}{\partial \theta_n} \end{bmatrix}. \tag{3.10}$$

Position dynamic

Always referring to equation (3.3) and as mentioned in the last section, the only uncertain variable that enters in the position dynamic is the disturbance. Differing from the attitude, since there are no other uncertain model parameters, d_F has been modelled in a more complex way with respect to the attitude case. In particular, the disturbances have been treated as a second order polynomial function of the translational velocity, as shown in equation (3.11):

$$d_F = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} + d_4 \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + d_5 \begin{bmatrix} v_1|v| \\ v_2|v| \\ v_3|v| \end{bmatrix} \tag{3.11}$$

where $d_1 \in \mathbb{R}$, $d_2 \in \mathbb{R}$, $d_3 \in \mathbb{R}$, $d_4 \in \mathbb{R}$ and $d_5 \in \mathbb{R}$ are the polynomial coefficients, which are unknown and consequently treated as uncertain variables. Then, it is possible to define the vector:

$$\theta_p = [d_1, \quad d_2, \quad d_3, \quad d_4, \quad d_5]^T \quad \in \mathbb{R}^5 \tag{3.12}$$

which collects, as in the attitude case, all the uncertain variables, and the vector:

$$\theta_{p4,5} = [d_4, \quad d_5]^T \quad \in \mathbb{R}^2 \tag{3.13}$$

that is given by the last two components of θ_p. One should also note that the first component of equation (3.11) is constant, while the last twos depend on the state, in particular on the translational velocity. Therefore, in order to apply a suitable adaptive controller, the equation can be better rewritten as a sum of an offset plus a state dependent term:

$$d_F = d_{off} - f(X)\theta_{p4,5} \tag{3.14}$$

where:

$$d_{off} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} \in \mathbb{R}^3 \qquad f(X) = \begin{bmatrix} v_1 & v_1|v| \\ v_2 & v_2|v| \\ v_3 & v_3|v| \end{bmatrix} \in \mathbb{R}^{3\times 2}.$$

As it will be showed in the next section, this subdivision of the disturbance terms is useful to write the final uncertain state space model. In fact, the offset term is the corresponding of d_T in the attitude dynamic, while $f(x)\theta_{p4,5}$ is the corresponding of the state dependent term $W_a\theta_a$. Moreover, $f(x)\theta_{p4,5}$ represents the aerodynamic resistance that has been taken with the minus sign, since it opposes to the translational velocity.

3.2.3 Complete uncertain model

Considering all the derivations done until now, by selecting equation (3.7) and (3.14) and taking into account equation (3.9) and (2.6), it is possible to write the tilt-arm final uncertain model:

$$\begin{cases} \dot{x} = v \\ \dot{\alpha} = \omega \\ m\dot{v} = -mge_3 + d_{off} + f(X)\theta_{p4,5} + R(f_c^b + f_c^a) \\ \dot{\omega} = \bar{J}^{-1}\tau_c^b + J^{-1}(W_a\theta_a + \tau_c^a + d_T). \end{cases} \tag{3.15}$$

From the uncertain model just derived and from equation (3.3), it should be clear that the first two equations, which refer to the kinematic of the system, are in cascade with respect to the last twos, which instead refer to the dynamic. In other words, the translational and angular velocity, output of the dynamic equations, are respectively an input for the position and the attitude kinematic equations. This allows the implementation of a cascade baseline controller, like the CPID illustrated in figure 1.4, with a similar structure to the dynamic system. Furthermore, keeping in mind this aspect and considering the fact that, as cited in section 3.2.1, the dynamic equations are the only ones affected by system uncertainties and disturbances, this allows the implementation of an adaptive controller referring on the attitude and position dynamic only. In fact, as it will be shown in the next chapters, the adaptive law will take the translational and angular velocity to compute the adaptive error, but not the position or the attitude.

Moreover, as previously highlighted and as demonstrated by the linearised model derived in equation (3.3), the position and attitude dynamics of the tiltable propellers UAV are decoupled. This allows to design each block of the adaptive architecture separately. Therefore, in the next two chapters the following elements will be designed independently for both the position and the attitude:

- The baseline controller
- The reference model
- The adaptive law.

3.3 Conclusion

In conclusion, it is possible to note that the derived uncertain model and control architecture is the one that is actually implemented in the simulators. The idea of splitting the baseline control input from the adaptive one allows the implementation of different adaptive control strategies without changing the baseline controller. Some simplifications such as the linearisation of the system have been made to calculate the final state space model. In the next chapters, since in the linearised model the position dynamic is decoupled from the attitude one, a separated design approach will be carried on. More precisely, we will try to understand if an adaptive law based on a linearised dynamic gives sufficiently good performances and which is the effort for its tuning.

Chapter 4

Attitude Adaptive Control System

As previously mentioned, thanks to the linearised decoupled UAV model, the controller acting on the attitude will be designed separately from the position. In particular, in this chapter we refer specifically to the attitude dynamic. Firstly the baseline controller and the reference model will be shown. Secondly, the details about the uncertain state space model and the attitude control strategy will be illustrated together with some numerical results computed by the simplified simulator. More specifically, an analysis on the sensitivity of the most important parameters will be carried on. Then, the first results for the MRAC and CM-RAC architectures will be showed, clarifying the peculiarities encountered and highlighting all the possible drawbacks. Finally, the results coming from the non linear simplified simulator and later from the non linear complete one will be presented, in order to verify the chosen control strategy in increasingly complex numerical environments.

4.1 Baseline controller

For all the reasons explained in section 1.5 the CPID and, more precisely, an its simplified version, has been chosen as the baseline controller for the adaptive architecture explained in 2.2. In this latter, the baseline controller is the one responsible of bringing the system close to the desired trajectory. Given that the reference model represents the ideal dynamic that we want the UAV to have, it should be clear that the same baseline controller has also to be implemented inside the reference system. In this way we are able to simulate the system behaviour in case of null disturbances and uncertainties. Therefore, the attitude baseline controller becomes:

$$\begin{cases} \omega_v = -K_{po}^A sign(q_e)\mathbf{q_e} \approx -\frac{1}{2}K_{po}^A \alpha_x \\ \tau_c^b = K_{ff}^A \omega_v + K_{pi}^A(\omega_V - \omega). \end{cases} \tag{4.1}$$

As can be seen, the controller takes the same exact architecture presented in equation (1.12) but considering the attitude portion of the system and without the integrative and derivative terms, since their tasks are moved to the adaptive controller.

4.2 Uncertain state space model

Once the baseline controller and the linear uncertain plant have been derived, it is possible to write the complete uncertain state space model in a form equivalent to the one showed in equation (3.5). Indeed, considering only the attitude equations presented in (3.15), one should obtain:

$$\begin{aligned}\begin{bmatrix}\dot{\alpha}\\ \dot{\omega}\end{bmatrix} &= \underbrace{\begin{bmatrix}0_3 & 0_3 & 0_3 & I_3\\ 0_3 & 0_3 & 0_3 & -\bar{J}^{-1}K_{pi}^{A}\end{bmatrix}}_{A}\underbrace{\begin{bmatrix}\alpha\\ \omega\end{bmatrix}}_{X}+\\ &+\underbrace{\begin{bmatrix}0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & \bar{J}^{-1}(K_{ff}^{A}+K_{pi}^{A})\end{bmatrix}}_{B_r}\underbrace{\begin{bmatrix}0_{31}\\ \omega_v\end{bmatrix}}_{r}+\\ &+\underbrace{\begin{bmatrix}0_3\\ \bar{J}^{-1}\end{bmatrix}}_{B}\underbrace{\begin{bmatrix}0_3\\ \bar{J}J^{-1}\end{bmatrix}^T}_{\Lambda}\underbrace{\left(\begin{bmatrix}0_{31}\\ W_a\theta_a\end{bmatrix}+\begin{bmatrix}0_{31}\\ \tau_c^a\end{bmatrix}+\begin{bmatrix}0_{31}\\ d_T\end{bmatrix}\right)}_{Uncertain}\end{aligned} \tag{4.2}$$

where as mentioned in section 3.2.1, the baseline controller appears inside the A and B_r matrix, the r vector collects the desired angular velocity generated by the controller and, the last row collects the uncertain portion of the model. In particular, since the nominal $\bar{J}$ is known, B is known, on the other hand Λ is unknown given that $J = \tilde{J} + \bar{J}$ is unknown. Finally, it is possible to see that the uncertain contribution is split in three different terms: a first one that depends on the state and on the desired dynamic, a second term that represents the adaptive control input and finally a third one that collects all the constant disturbances that are independent on the state.

4.3 Reference model

Referring to section 2.2, we have seen that the control system chosen is made by a reference model that generates the desired state on-line. This latter is responsible of the main difference between the MRAC and the CMRAC approach. In fact, in the first one it is simply computed by integrating a simplified version of the linear dynamic equations showed in (3.3), while in the second, as illustrated in equation (2.5), an error dependent term has to be added. Since the idea is to have an "ideal" state, more precisely a reference model output which is as perfect as possible, referring to the attitude dynamic only, it has been decided to assume:

- Negligible disturbance: $d_T = 0$
- $x_c \approx [0, 0, 0]$
- Exact inertia tensor: $J = \bar{J}$ with $\bar{J}$ equals to equation (4.10).

Therefore, the MRAC differential reference model equation describing the behaviour of the quadrotor angular velocities $\omega_m \in \mathbb{R}^3$ becomes simply:

$$J\omega_m = \tau_c^b = K_{ff}^A \omega_v + K_{pi}^A(\omega_V - \omega_m) \tag{4.3}$$

while in the case of the CMRAC:

$$J\omega_m = \tau_c^b + \lambda_a e_a. \tag{4.4}$$

Where $\lambda_a \in \mathbb{R}^{3\times3}$ is the attitude closed-loop gain and $e_a = \omega - \omega_m$ is the adaptation error. It is possible to notice that equation (4.1) has been substituted inside equation (4.3), but changing ω with ω_m. As can be seen in figure 2.4, this has been done to highlight the fact that the adaptation acts on the inner loop only. Consequently the angular velocity has to be taken directly from the reference model itself. Instead, since ω_v is computed from the quaternion or the attitude angles, it refers to the outer loop of the system where there is no type of adaptation. Therefore, even for the reference model, q or α are given by the plant.

As it will be showed in the next sections, in order to better understand the behaviour of the system under study, some simulations will be computed with the complete tilt-arm simulator. In this context, the results obtained with the model with the CPID controller will be compared to the ones with the adaptive architecture. Therefore, since the reference model represents the desired performances that we want the plant to have, it has been decided to tune the reference model gains in such a way that its behaviour gets as close as possible to the CPID one. This will be achieved by looking directly to the plots, but also with the help of special tables in which the maximum absolute error and root mean square error will be reported for all the state variables under study. In this optic, considering the error Δz between the values predicted by a model or an estimator and the values observed, the maximum absolute error is defined as:

$$e_{abs} = max(\|\Delta z(t)\|) \tag{4.5}$$

while the root mean square error is generally defined as:

$$e_{rms} = \sqrt{\left(\frac{\sum \Delta z_i^2}{N_{tot}}\right)} \tag{4.6}$$

where N_{tot} is the total number of the ith samples. The values of the tuned gains can be found in appendix A table A.2. Concerning the set points, the desired

Euler angles and position are showed in figure 4.1. The trajectory chosen is the so called Paoll trajectory, which can be performed only by a tiltrotor. After a first step in position, the UAV has to perform two opposite 90 degrees rotations around itself while changing its roll and pitch angles and maintaining the position reached. At the end, it will perform another final position change along x_1.

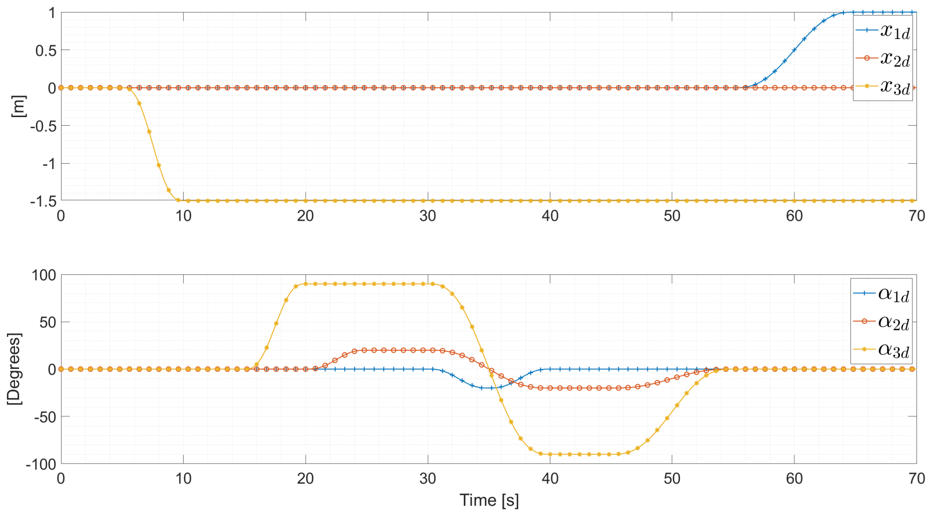

Figure 4.1: Paol1 trajectory: desired position (on top) and Euler angles (bottom)

Figure 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, and table 4.1, 4.2 show the results obtained. They refer to the attitude angles, the angular velocities, the control input torques and the brackets β angles, since in this chapter we analyse the attitude dynamic only.

e_{abs}	**U.M.**	**CPID**	**Ref. Model**
$\alpha_{\mathbf{x1}}$	[deg]	3.1809	3.0994
$\alpha_{\mathbf{x2}}$	[deg]	5.0513	4.9113
$\alpha_{\mathbf{x3}}$	[deg]	17.4469	15.8781
$\omega_{\mathbf{x1}}$	[deg/s]	5.9910	5.9277
$\omega_{\mathbf{x2}}$	[deg/s]	11.9238	11.9520
$\omega_{\mathbf{x3}}$	[deg/s]	10.9417	9.5080

Table 4.1: Comparison of the maximum absolute error e_{abs} between the system with the CPID and the reference model for the attitude dynamic.

As it is possible to see, the behaviour of the two simulations and the errors reported in the tables are very similar.

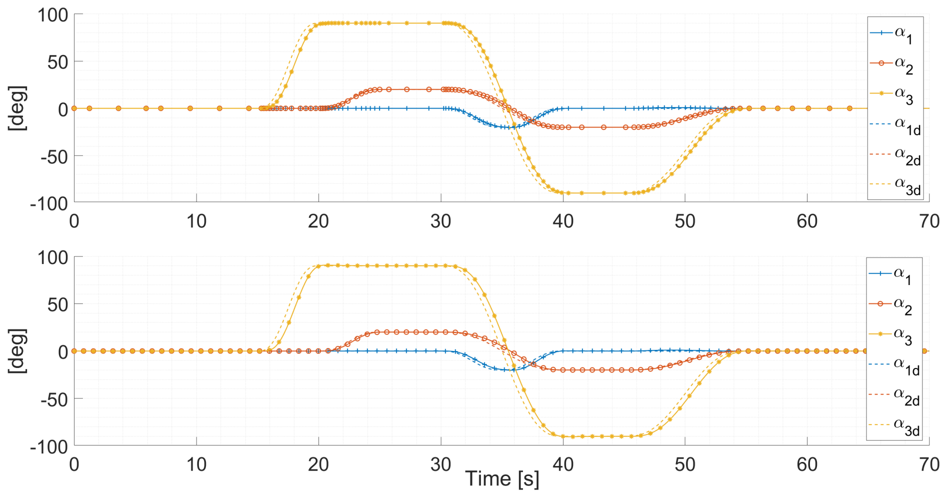

Figure 4.2: Plant and desired Euler angles for the reference model (on top) and for the system with the CPID controller (bottom).

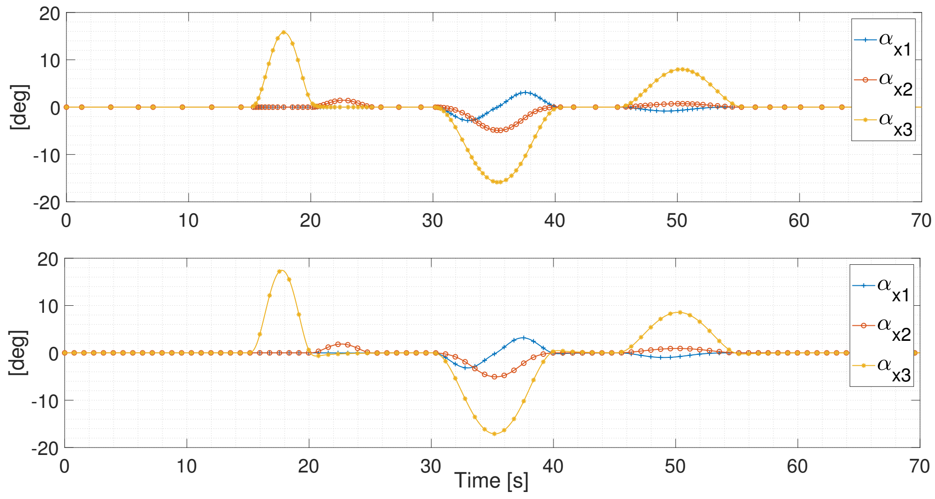

Figure 4.3: Euler angles error α_x for the reference model (on top) and for the system with the CPID controller (bottom).

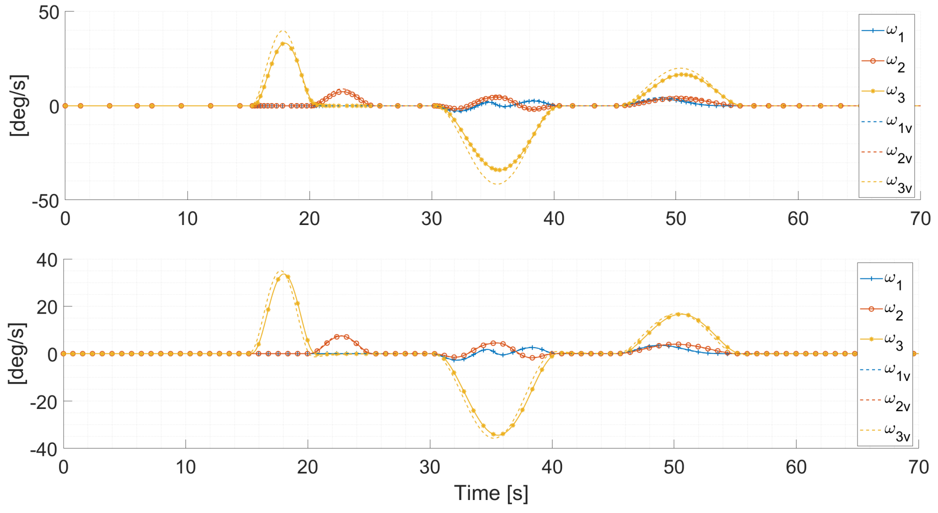

Figure 4.4: Plant and desired angular velocities for the reference model (on top) and for the system with the CPID controller (bottom).

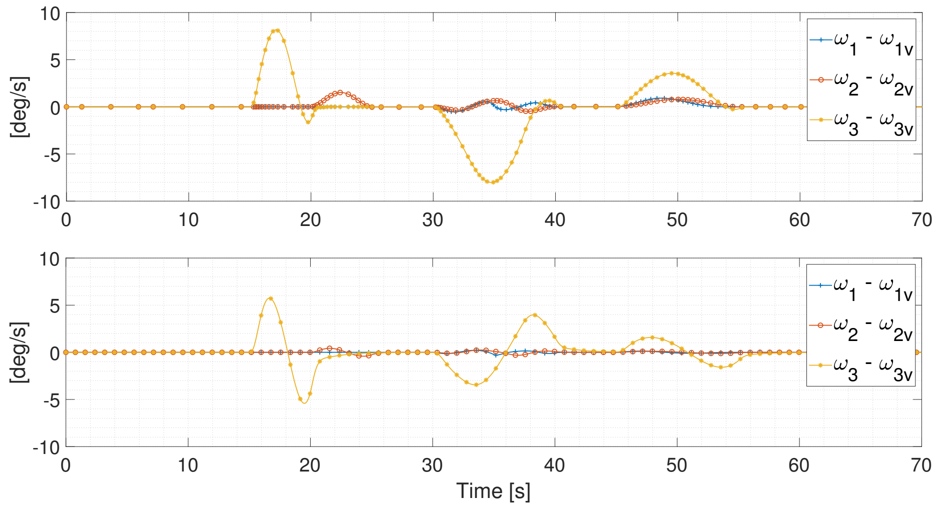

Figure 4.5: Angular velocity error with respect to the outer-loop ω_v for the reference model (on top) and for the system with the CPID controller (bottom).

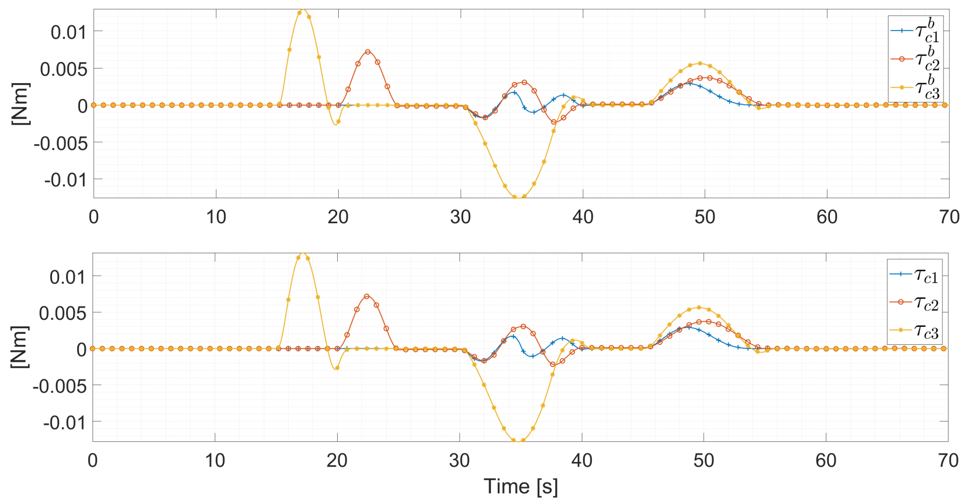

Figure 4.6: Torque control input $\tau_c = \tau_c^b + \tau_c^a$ for the reference model (on top) and for the system with the CPID controller (bottom).

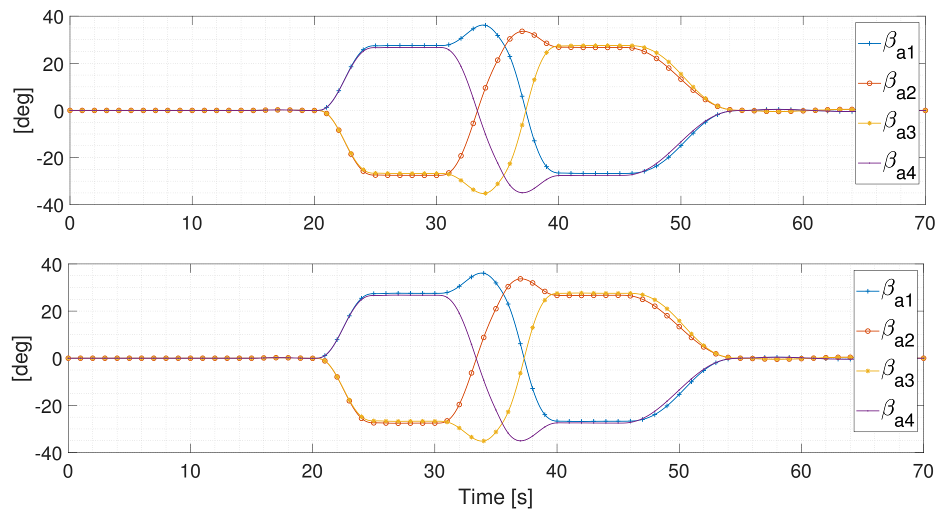

Figure 4.7: Servo-actuators angular position β_a requested by the four UAV's propellers for the reference model (on top) and for the system with the CPID controller (bottom).

e_{rms}	**U.M.**	**CPID**	**Ref. Model**
$\alpha_{\mathbf{x1}}$	[deg]	0.8024	0.9753
$\alpha_{\mathbf{x2}}$	[deg]	1.1525	1.4469
$\alpha_{\mathbf{x3}}$	[deg]	5.4493	6.3896
$\omega_{\mathbf{x1}}$	[deg/s]	0.0118	0.0061
$\omega_{\mathbf{x2}}$	[deg/s]	0.0187	0.0100
$\omega_{\mathbf{x3}}$	[deg/s]	0.0202	0.0094

Table 4.2: Comparison of the root mean square error e_{rms} between the system with the CPID and the reference model for the attitude dynamic.

4.4 Attitude adaptive laws

As mentioned before, since it has been discovered that the position and the attitude dynamic in the linear plant of (3.3) are decoupled, the idea is to design separately the two contributions, given that this approach is simpler and more intuitive. The hope is that this way of work will give, however, good results. For this reason, since in this chapter we are studying the attitude dynamic only, the position has been taken as ideal. More precisely, referring to the position dynamic, it has been assumed:

- Null disturbances: $d_F = 0$
- Null center of mass position in the third equation of (1.11)

As previously said, the idea behind an adaptive controller is to estimate the uncertain variables through the information contained in the plant state, the reference model and the desired set point. This estimation is usually achieved thanks to adaptive control laws, which will be derived in such a way that they are valid for both the MRAC and the CMRAC, so that they ensure robustness, good estimation of the uncertain parameters and tracking. Referring to the attitude dynamic only, as discussed in the previous chapter, this can be accomplished by taking the control input such that it eliminates the uncertain contributions $W_a\theta_a$ and d_T showed in equation (4.2). Therefore, one can write:

$$\tau_c^a = -W_a\hat{\theta}_a - \hat{d}_T \tag{4.7}$$

where $\hat{\theta}_a$ and $\hat{d}_T$ are the estimated values of θ_a and d_T. This estimation will be done on-line exploiting the Projection Operator. The reason of this choice lies in the fact that, as illustrated in section 2.2, the Projection Operator avoids drifting of the uncertain parameters, *wind up* problems for the integrators and, more important, bounds the overall adaptive process. Therefore, as showed in equation (4.8), the attitude adaptive law becomes:

$$\dot{\hat{\theta}}_a = \gamma_a \cdot Proj\left(\hat{\theta}_a, W_a^T B_a^T P_a e_a\right) \qquad \dot{\hat{d}}_T = \gamma_a \cdot Proj\left(\hat{d}_T, B_a^T P_a e_a\right) \tag{4.8}$$

where $e_a = \omega - \omega_m \in \mathbb{R}^3$ is the angular velocity error between the plant and the reference model, P_a is the solution of:

$$A_a^T P_a + P_a A_a = -Q$$

$\gamma_a \in \mathbb{R}$ is the attitude adaptation gain, A_a, Q and B_a are chosen for simplicity by looking at the last component of the state space model showed in equation (4.2), given that in this section we are focusing on the adaptive controller acting on the attitude dynamic only. Therefore:

$$\begin{cases} B_a = \bar{J}^{-1} \\ A_a = -K_{pi}^A \bar{J}^{-1} \\ Q = I_3. \end{cases}$$

The attitude reference model used to compute the above cited errors has been implemented respectively for the MRAC and the CMRAC architecture as the one showed in equation (4.3) and (4.4).

4.5 Tilt-arm simplified simulators

The simulator presented in the first chapter has not been the only one developed. Other two less complicated versions of it have been built from scratch. They have been used for the validation of the attitude control system only, since it is the dynamic more affected by uncertainties. Conversely, the position has been validated through the complex non linear simulator alone. In particular, there have been developed:

- A Linear Simplified Simulator
- A Non Linear Simplified Simulator

Their architecture can be seen in figure 4.8. The plant generates the tiltrotor state, which is used by the W Generator and reference model block to compute respectively the W_a matrix and the error vector e. Then, these data are used by the adaptive controller block to calculate τ_c^a that, together with the baseline control inputs τ_c^b and f_c^b re-enter in the plant. Referring to the above item list, the simulators are characterized by the word "simplified", which means the UAV plant only has been integrated without any sort of modelling for the propellers and the servo actuators. In other words, the controllers generate the force and the torque that directly enter in the dynamic system previously discussed. This means that in these two simulators the allocation algorithm that transforms f_c and τ_c to θ_a and ω_r is not present, and with it all the saturations and actuators dynamics. Furthermore, the physical data given to the simulators are in agreement with common small UAVs but they do not respect any real prototype. The two

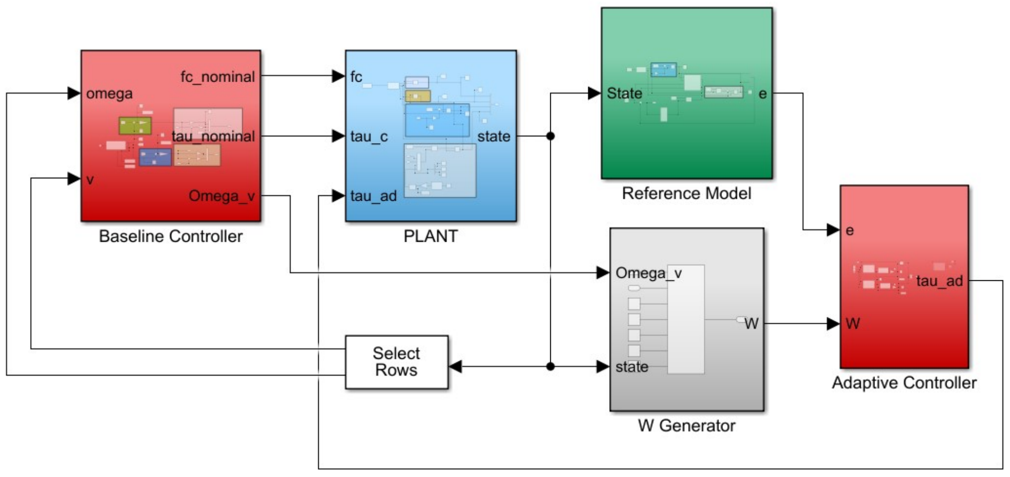

Figure 4.8: Simplified numerical model of the tiltrotor

cited simulators differ only by the fact that, inside the plant block, in the first the linear dynamic equations are implemented, while in the second the non linear ones. These two software plus the one showed in section 1.4 should be seen as a way to find the right controller for the tiltrotor. Due to its simplicity, the first one can be used to address all the major limitations and possible problems for each approach, giving the possibility to select the best control system between the MRAC and the CMRAC. Being completely linear, it can be also used to verify that the results of the theory and of the literature are respected. Then, the second one and later the simulator showed in the first chapter are used to verify the complete performances of the system in each time more and more complex numerical environment.

4.6 Adaptive gains tuning

As previously said, the two control architectures that have been implemented in this work are a modification of the MRAC and CMRAC system, as presented in section 2.2. In this section, the first numerical results for both the MRAC and the CMRAC controllers will be showed highlighting how the different adaptive controller parameters affect the behaviour of the system. For this first approach, the simplified linear simulator has been used with some physical data that do not respect real values of a real prototype, as reported in table 4.3. Moreover, always referring to the same table, it has been assumed that the tiltrotor is affected by a constant disturbance torque. As a reminder, the position for now is not taken into account since, in this section, only the attitude is under study. Instead, the plant J, which represents the actual tiltrotor inertia that is unknown in the real

Mass	1 Kg
x_c	$[0.01 \quad 0.01 \quad 0.01]^T$ m
d_T	$[0.01 \quad 0.01 \quad 0.01]^T \cdot \text{step}(t)$ Nm

Table 4.3: Physical variables used in the first simulations

world, for simulation reasons it has been taken as:

$$J = \begin{bmatrix} 3\times10^{-3} & 1\times10^{-5} & 1\times10^{-5} \\ 1\times10^{-5} & 3\times10^{-3} & 1\times10^{-5} \\ 1\times10^{-5} & 1\times10^{-5} & 3\times10^{-3} \end{bmatrix} \tag{4.9}$$

while the nominal inertia, which is the one that is estimated and used in the reference model, it has been taken as a diagonal matrix with values slightly different from J, in order to simulate the fact that a rough inertia estimation can be usually obtained:

$$\bar{J} = \begin{bmatrix} 3.1\times10^{-3} & 0 & 0 \\ 0 & 3.2\times10^{-3} & 0 \\ 0 & 0 & 2.8\times10^{-3} \end{bmatrix}. \tag{4.10}$$

The simulator initial conditions for all the states is reported in table 4.4. Instead,

x_0	$[0.3 \quad 0.3 \quad 0.3]^T$ m
α_0	$[0.1 \quad 0.1 \quad 0.1]^T$ rad
v_0	$[0.1 \quad 0.1 \quad 0.1]^T$ m/s
ω_0	$[0.2 \quad 0.2 \quad 0.2]^T$ rad/s

Table 4.4: Initial conditions used in the first simulations

the baseline controller gains have been chosen, as reported in table 4.5, big enough in such a way that the transients are as fast as needed, in order to allow the tracking in a sufficiently small time interval. On the contrary, small enough to not generate instability phenomena or a too much stiff system. Finally, as showed

Attitude		**Position**	
K^A_{ff}	0.02	K^P_{ff}	2.5
K^A_{pi}	0.02	K^P_{pi}	2.5
K^A_{po}	0.5	K^P_{po}	0.4

Table 4.5: Baseline controller gains used in the first simulations

in equation (4.11), the desired set point has been chosen as a piecewise constant function constituted by a sequence of steps with a duration of 10 seconds both on the position and attitude. In particular: $x_{1d} = x_{2d} = x_{3d} = F(t)$ and $\alpha_{1d} = \alpha_{2d} = \alpha_{3d} = F(t)$. This permits to evaluate the transients behaviour of the system and its response in different conditions. It is possible to notice that, different from the fixed propeller UAV, in a tiltrotor the desired attitude can be set independently from the position.

$$F(t) = \begin{cases} 1 & \text{if} \quad 0 \leqslant t < 10 \\ 0.8 & \text{if} \quad 10 \leqslant t < 20 \\ 0.2 & \text{if} \quad 20 \leqslant t < 30 \\ 0.5 & \text{if} \quad 30 \leqslant t < 40 \\ 0.9 & \text{if} \quad 40 \leqslant t < 50 \\ 1.4 & \text{if} \quad 50 \leqslant t < 60 \\ 1 & \text{if} \quad 60 \leqslant t < 70 \\ 0.2 & \text{if} \quad 70 \leqslant t < 80 \\ -0.1 & \text{if} \quad 80 \leqslant t < 90 \\ 0 & \text{if} \quad 90 \leqslant t < 100 \end{cases} \tag{4.11}$$

4.6.1 MRAC approach - Adaptation gain sensitivity

With the data previously presented, we now consider the MRAC architecture illustrated in figure 2.4 and we try to understand if this architecture is suitable for a tiltrotor and how, the only tuning parameter γ_a affects the performances of the system. For all the different attempts, since only the attitude is under study, the α_x, ω and τ_c^a plots will be used as reference for all the different analysis. The overall idea is to show how the UAV dynamic behaves when the tuning parameter has a very low or high value, in order to correctly end up to what we think is the correct γ_a number for this particular application.

$$\boldsymbol{\gamma_a = 0.001}$$

We start from setting $\gamma_a = 0.001$, which represents a quite low value. In figures 4.9, 4.10 and 4.11 are respectively reported the α_x, ω and τ_c^a plots.

As highlighted in the previous chapters and showed in the τ_{c1}^a zoom in figure 4.12, the MRAC architecture, even with a very low adaptation rate, presents some oscillations during the transients. These vibrations usually increase incrementing the adaptation gain and they are present also on the angular velocity. Referring to 4.9, since γ_a is very small, it is possible to see that the tiltrotor takes more or less 20 seconds to annul the error, which is obviously too much for any kind of UAV applications. The adaptive torque does not show very big transients during the step change, but it seems that it converges to a certain offset and then, with a small torque changes, it tries to make the tracking. This latter effect is maybe

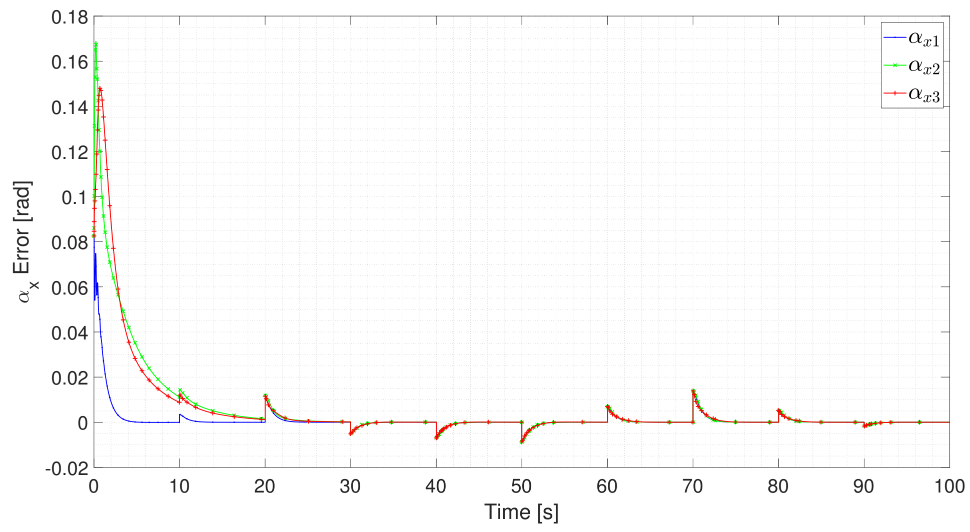

Figure 4.9: $\alpha_x = \alpha - \alpha_d$ error for the MRAC architecture with $\gamma_a = 0.001$

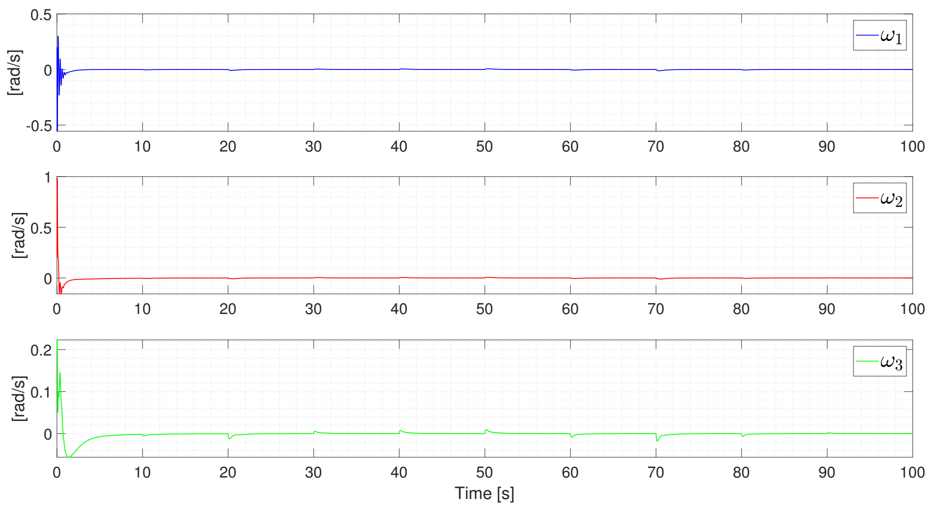

Figure 4.10: ω for the MRAC architecture with $\gamma_a = 0.001$

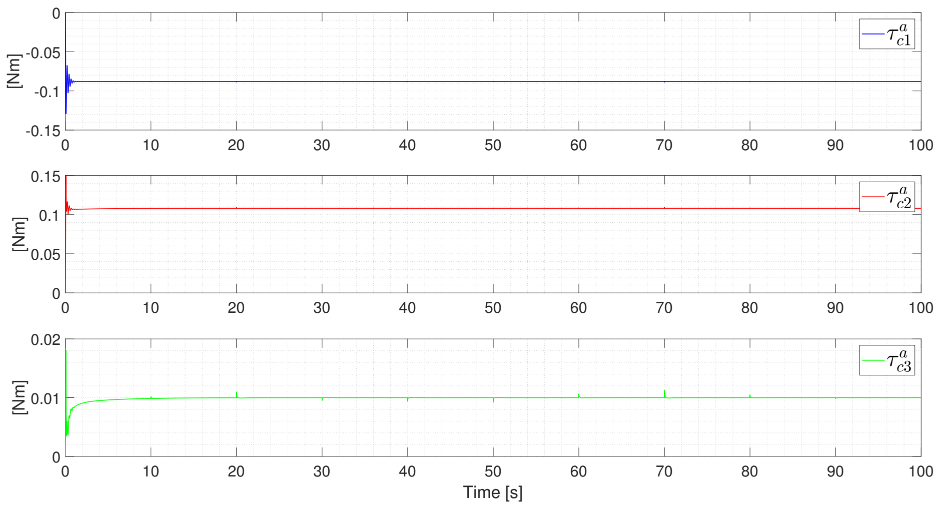

Figure 4.11: τ_c^a for the MRAC architecture with $\gamma_a = 0.001$

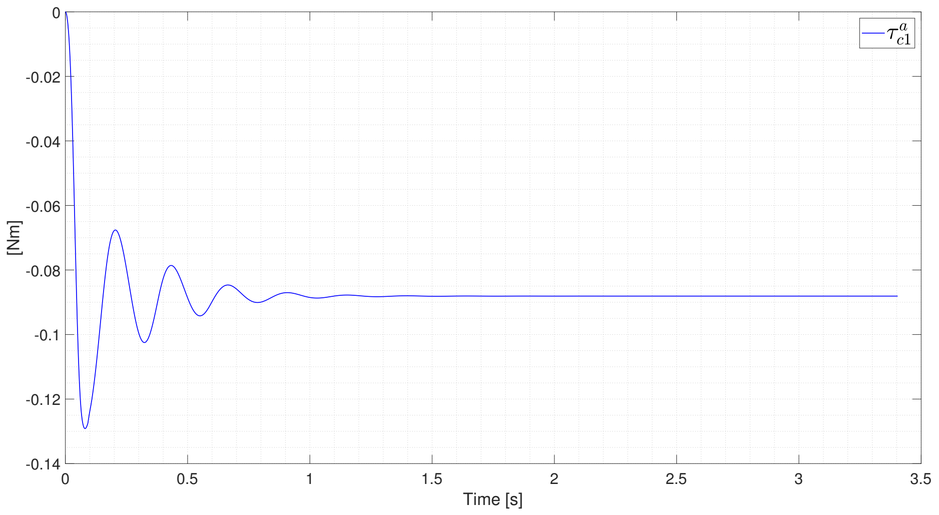

Figure 4.12: τ_{c1}^a Zoom for the MRAC architecture with $\gamma_a = 0.001$ to highlight oscillations

due to two different reasons: the small adaptation gain makes the system affected by a very smooth behaviour and the simplified linear simulator has been used, therefore all the non linearities are not taken into account.

$\gamma_\mathbf{a} = \mathbf{10}$

Now, we set an adaptation gain value four orders bigger than the previous one: $\gamma_a = 10$. As before, the α_x, ω and τ_c^a plots are respectively reported in figures 4.13, 4.14 and 4.15.

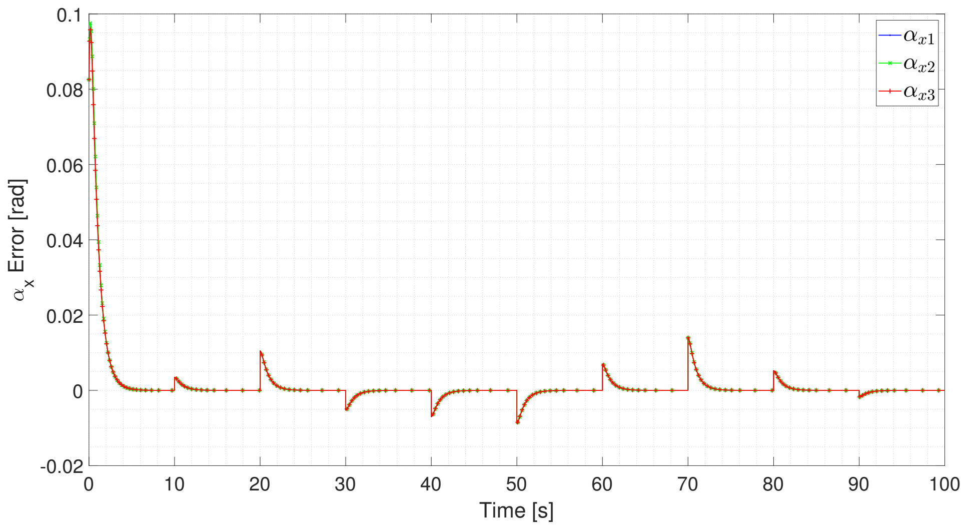

Figure 4.13: $\alpha_x = \alpha - \alpha_d$ error for the MRAC architecture with $\gamma_a = 10$

The first different thing that one can note from the previous case is the increased number of oscillations in the adaptive torque. The high value of γ_a generates very high frequency vibrations that can be dangerous for resonance phenomena, mechanical stiffness and computational power. In fact, the numerical simulator, in order to track this very fast dynamic, has to increase the number of points that translates in a bigger CPU effort. Nevertheless, it is possible to see that the α_x error converges very fast. The transients have a duration of a maximum of two seconds. It should be clear now that a compromise is needed. One would like to have the fastest convergence possible but with the fewest number of oscillations.

$\gamma_\mathbf{a} = \mathbf{0.05}$

Finally, after several attempts it has been decided to set $\gamma_a = 0.05$. This latter comes from the trade off explained before. Since it is not possible to have a fast error convergence without vibrations with a MRAC architecture, it has been tried to maintain a sufficiently high speed during the transients in order to allow the

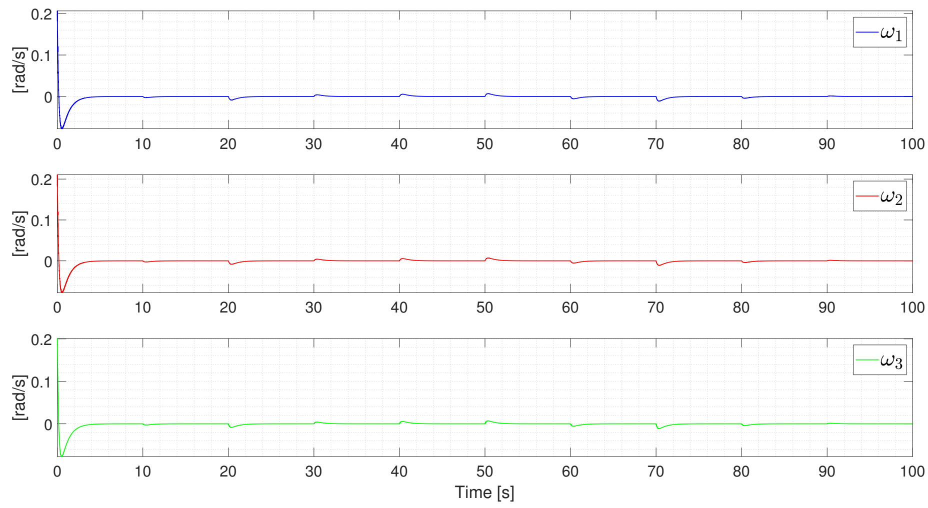

Figure 4.14: ω for the MRAC architecture with $\gamma_a = 10$

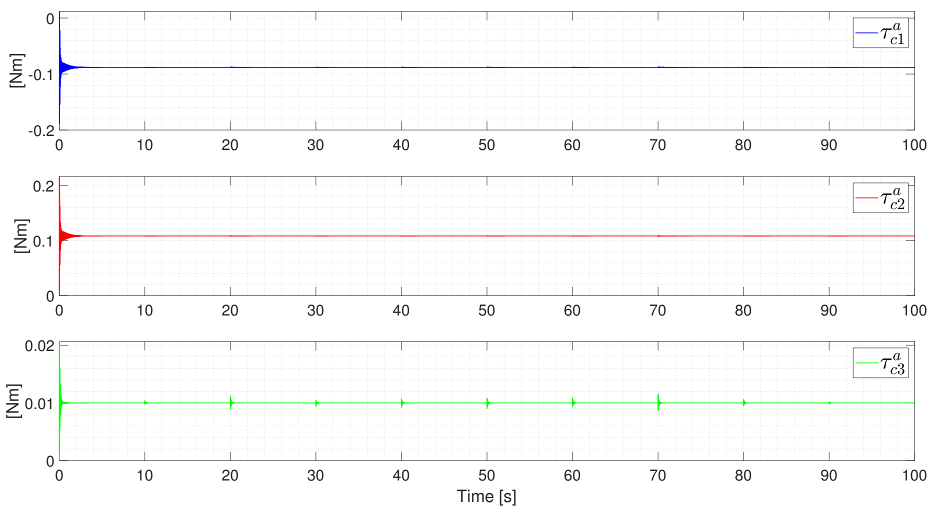

Figure 4.15: τ_c^a for the MRAC architecture with $\gamma_a = 10$

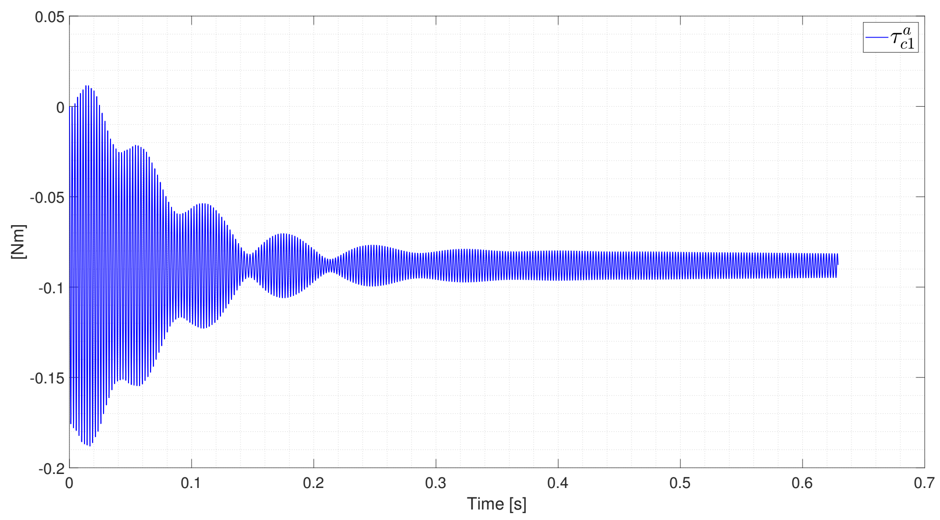

Figure 4.16: τ^a_{c1} Zoom for the MRAC architecture with $\gamma_a = 10$ to highlight oscillations

tracking in a suitable amount of time and, in the meanwhile, to decrease the high frequency oscillations as much as possible.

Different from the above two cases, here all the results obtained are showed in order to permit a complete overview of the outcomes achieved. As it is possible to see from figures 4.17, 4.18, 4.19 and 4.20, the UAV demonstrates very good tracking capabilities with no overshoots both in position and attitude. The convergence time is fast enough to track all the reference steps: on the attitude we have transients of about 3 seconds, while on the position of about 4/5 seconds, since the requested set point is quite large. Moreover, the translational and angular velocity showed in 4.21 and 4.22, like all the other variables, are inside the typical range of small scale UAVs. The torque control inputs, both the baseline in 4.23 and the adaptive in 4.24, still present high frequency oscillations, as highlighted in 4.25, but less intense with respect to the previous case. Also the angular velocity exhibits high frequency vibrations, specially at the beginning. Instead, since all the uncertainties and disturbances are only implemented on the attitude dynamic and, in the linear case, it is completely decoupled from the position, the baseline force control input in figure 4.26 does not display any kind of oscillations. In fact, it has also to be reminded that, in this case, the adaptation acts only on the attitude. The behaviours of the variables that are estimated by the adaptive law are instead reported in figures 4.27 and 4.28. The θ_a plot is made by two different scales. One on the left, representing the components of the moments of inertia in $kg \cdot m^2$ and one on the right, which represents the center of mass position with respect to O_b in meters. It is possible to see that, after a short oscillatory transient at the beginning, $\hat{d}_T$ and $\hat{\theta}_a$ both converges to a certain stable

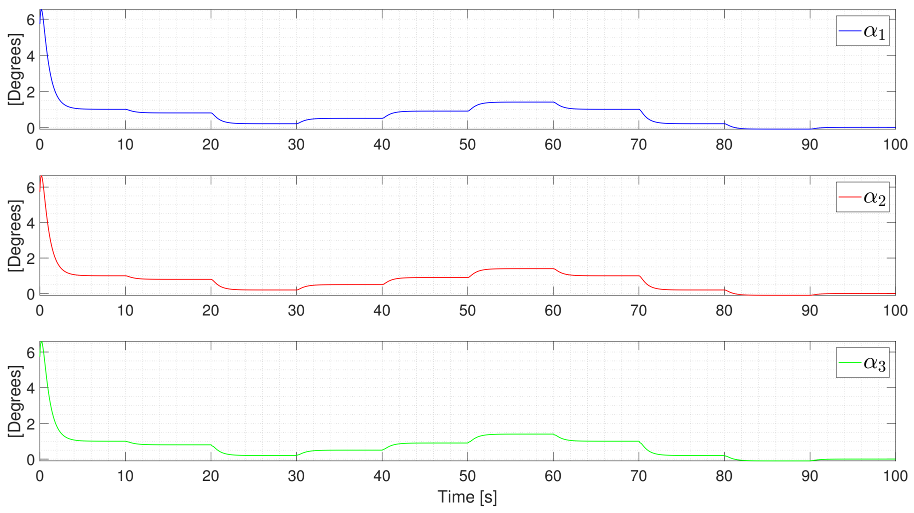

Figure 4.17: Attitude angles α for the MRAC architecture with $\gamma_a = 0.05$

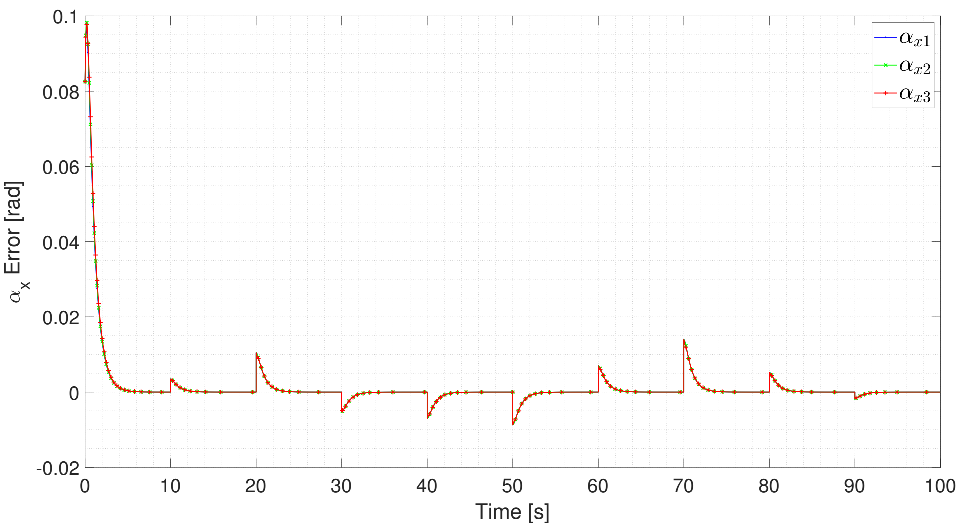

Figure 4.18: $\alpha_x = \alpha - \alpha_d$ error for the MRAC architecture with $\gamma_a = 0.05$

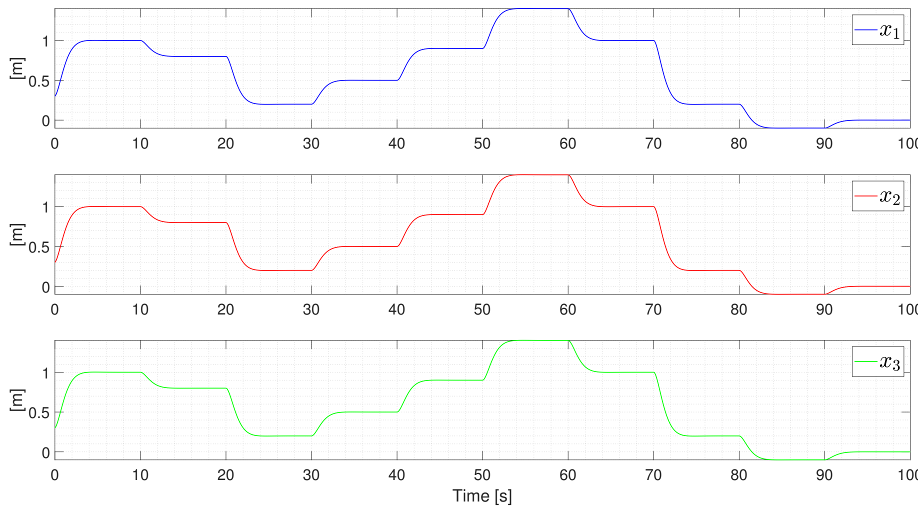

Figure 4.19: Position vector x for the MRAC architecture with $\gamma_a = 0.05$

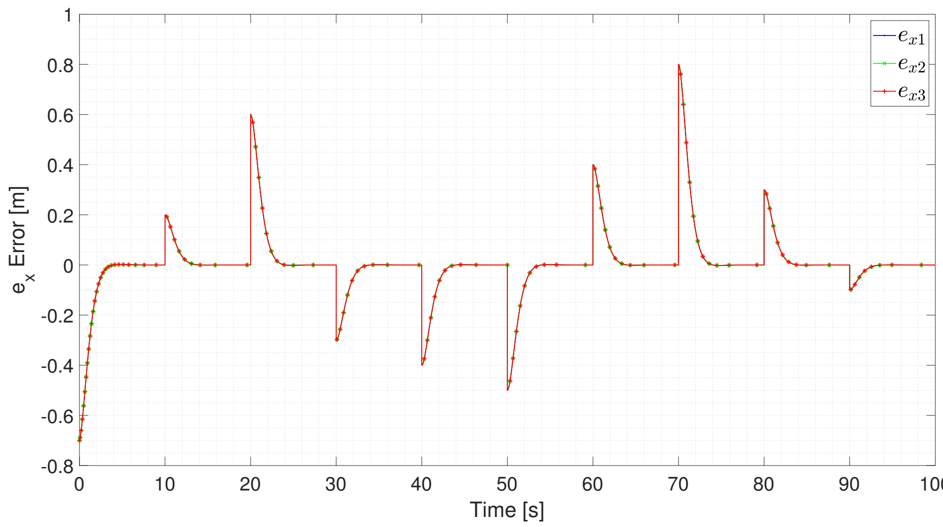

Figure 4.20: Position error $e_x = x - x_d$ for the MRAC architecture with $\gamma_a = 0.05$

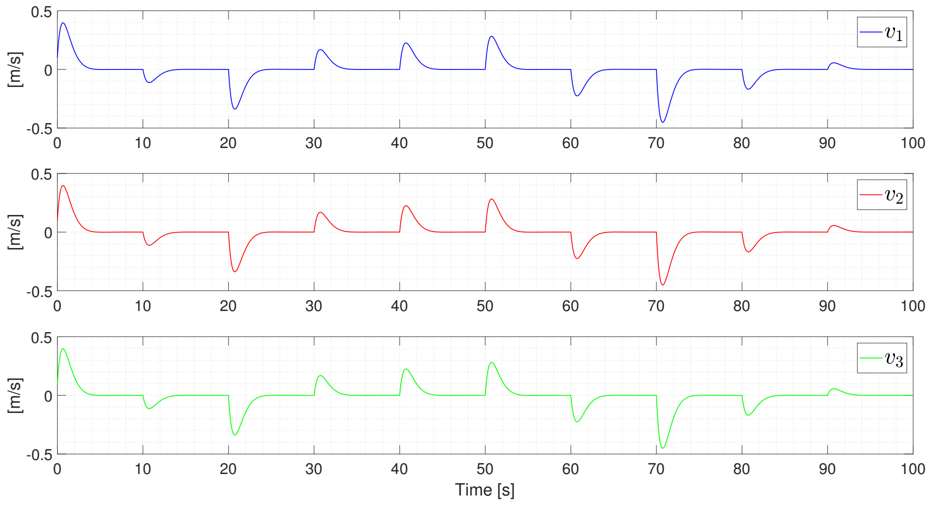

Figure 4.21: Velocity vector v for the MRAC architecture with $\gamma_a = 0.05$

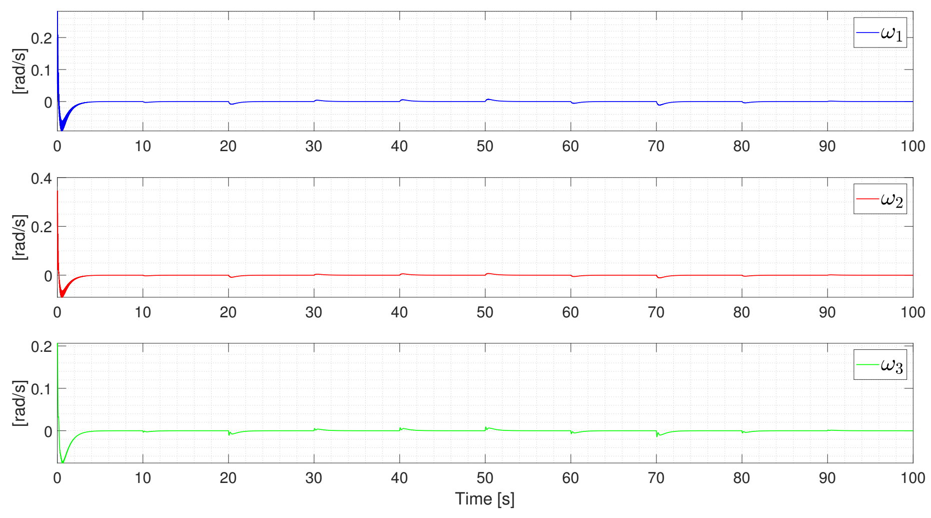

Figure 4.22: Angular velocity vector ω for the MRAC architecture with $\gamma_a = 0.05$

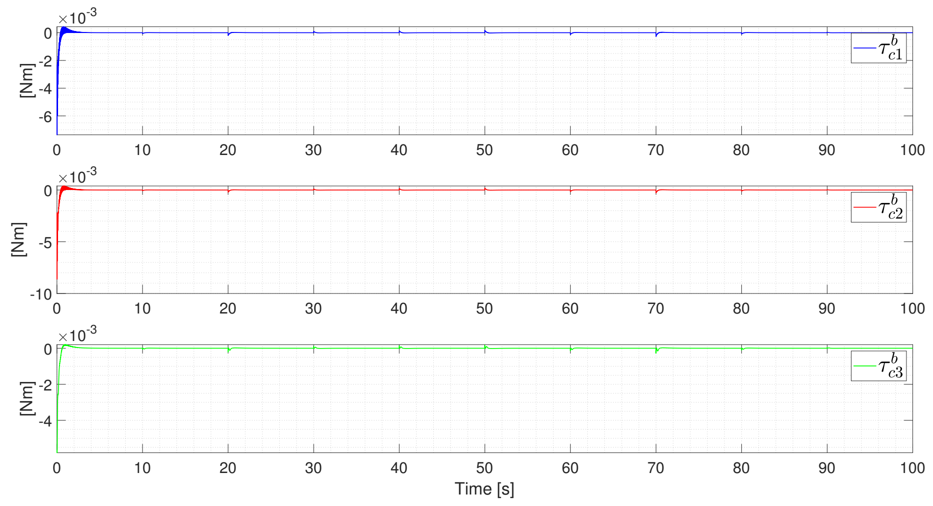

Figure 4.23: Baseline control torque τ_c^b for the MRAC architecture with $\gamma_a = 0.05$

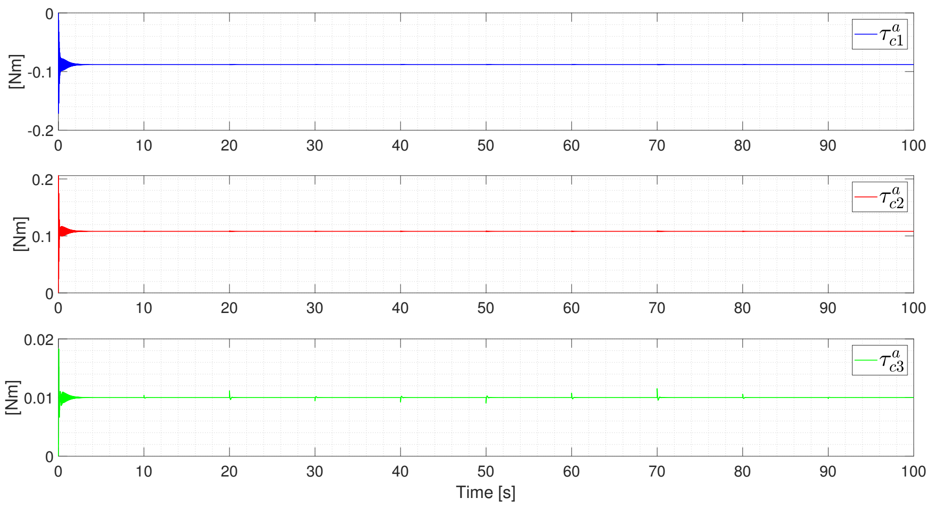

Figure 4.24: Adaptive control torque τ_c^a for the MRAC architecture with $\gamma_a = 0.05$

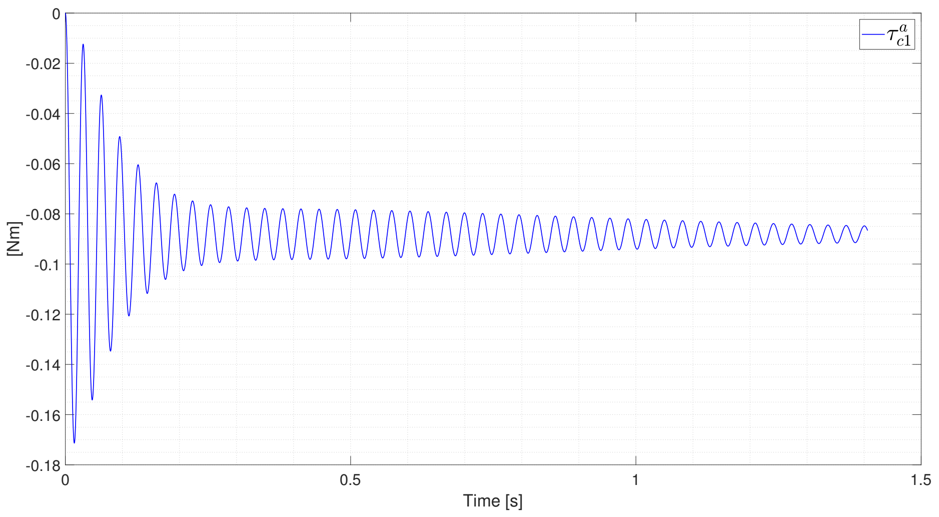

Figure 4.25: τ_{c1}^a Zoom for the MRAC architecture with $\gamma_a = 0.05$

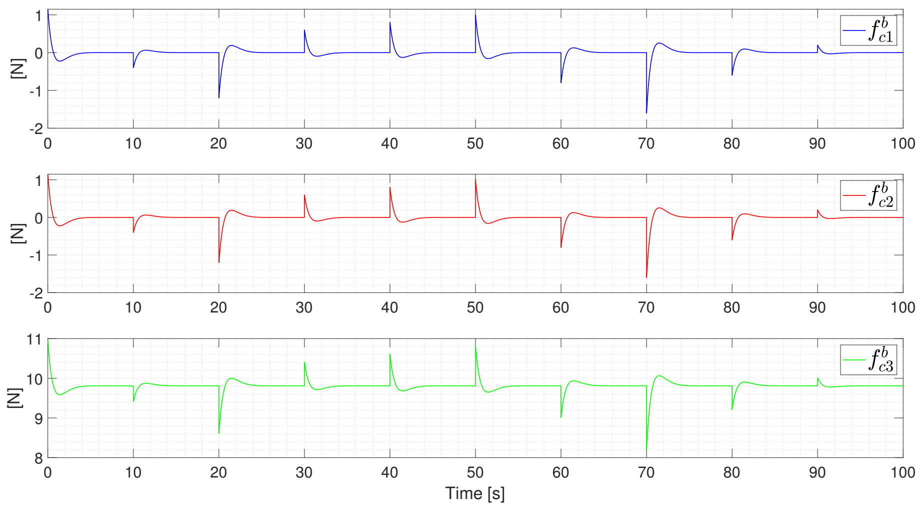

Figure 4.26: Baseline control force f_c^b for the MRAC architecture with $\gamma_a = 0.05$

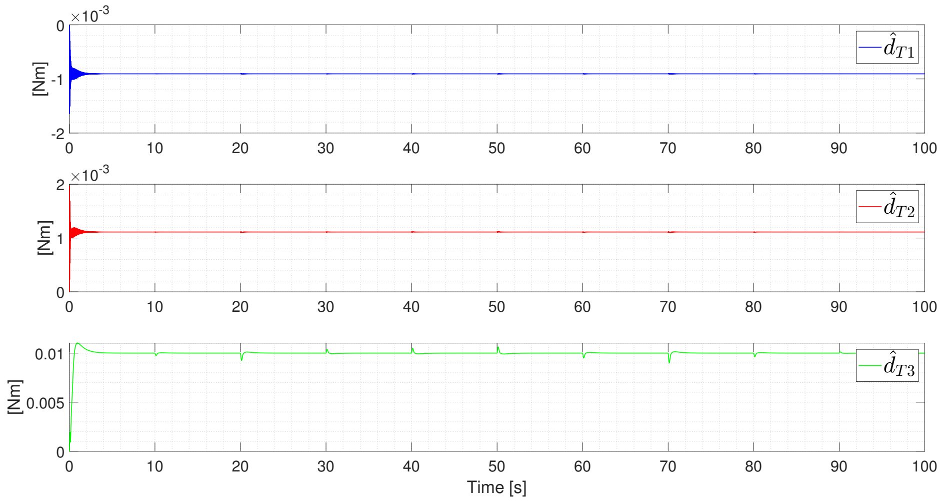

Figure 4.27: Estimation of the attitude disturbance vector d_T for the MRAC architecture with $\gamma_a = 0.05$

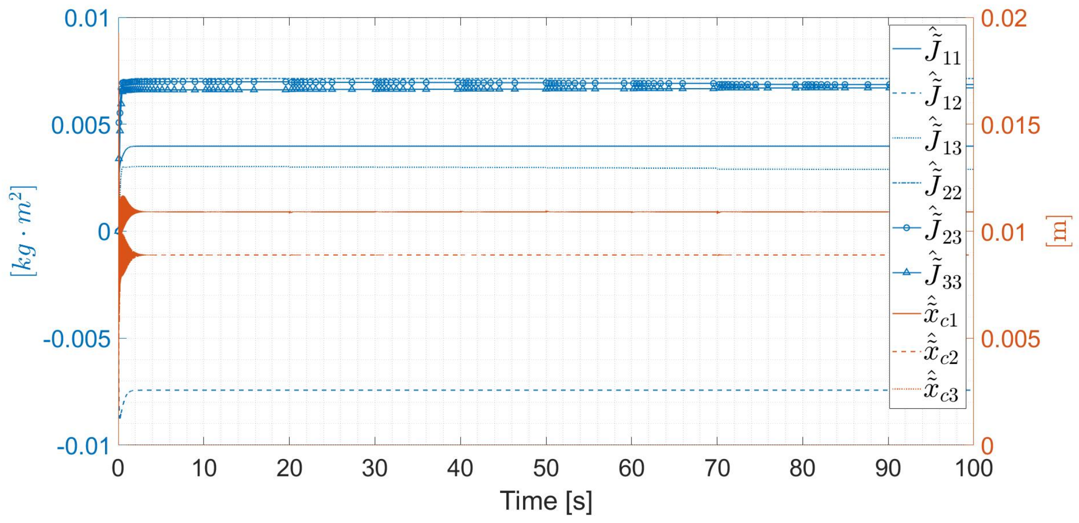

Figure 4.28: Estimation of the uncertain parameters contained in the θ_a vector for the MRAC architecture with $\gamma_a = 0.05$

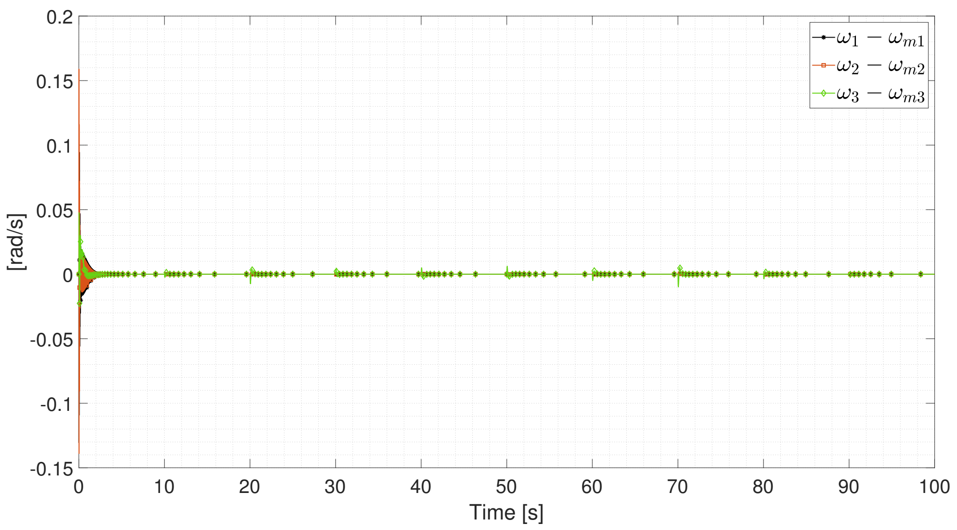

Figure 4.29: Error between the plant and reference model angular velocity ω and ω_m for the MRAC architecture with $\gamma_a = 0.05$

value without feeling too much the step changes. Finally, as explained before, the purpose of the adaptive controller is to generate a control input capable to ensure that the UAV follows the reference model performances. As illustrated in figure 4.29, this is easily achieved by the system after the first 5 seconds, and can be also appreciated after every step change.

4.6.2 CMRAC approach - Closed-loop gain sensitivity

In literature, one possible well known solution to the MRAC fast oscillations problem is the CMRAC approach. Unlike the other, in this case we have an additional degree of freedom: the closed-loop gain λ_a, which has to be studied in order to understand how it affects the tiltrotor performances. The approach is the same as before: since the attitude is under study, only the α_x, ω and the τ_c^a plots will be shown as reference for the analysis. In this case, given that there are two degrees of freedom: γ_a and λ_a, three different attempts will be shown before ending up to a good compromise for this system. Each test has been particularly useful to understand how the different parameters combinations influence the dynamic.

$$\boldsymbol{\gamma_a = 1 \quad , \quad \lambda_a = 10 \cdot I_3}$$

We begin from setting $\gamma_a = 1$ and $\lambda_a = 10 \cdot I_3$ as initial guess to understand how the system behaves. In figures 4.30, 4.31 and 4.32, the α_x, ω and τ_c^a plots are respectively reported.

From this first test, it is possible to see that, since the adaptation gain is an high value, the convergence of the attitude angles is very fast. However, some high

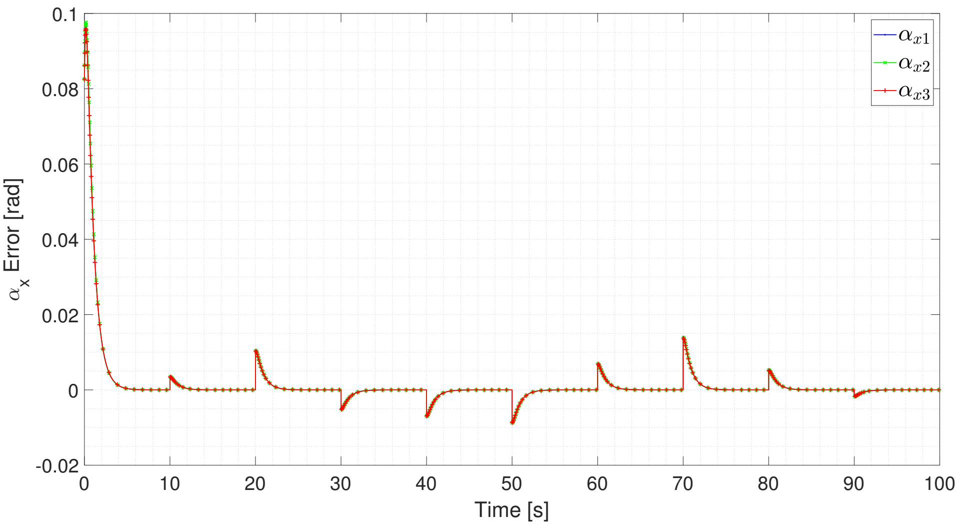

Figure 4.30: $\alpha_x = \alpha - \alpha_d$ error for the CMRAC architecture with $\gamma_a = 1$ and $\lambda_a = 10 \cdot I_3$

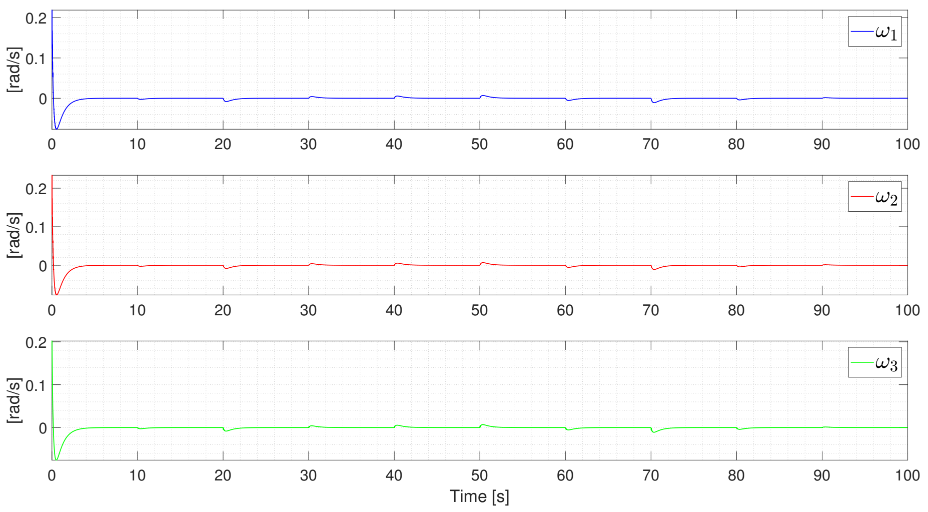

Figure 4.31: ω for the CMRAC architecture with $\gamma_a = 1$ and $\lambda_a = 10 \cdot I_3$

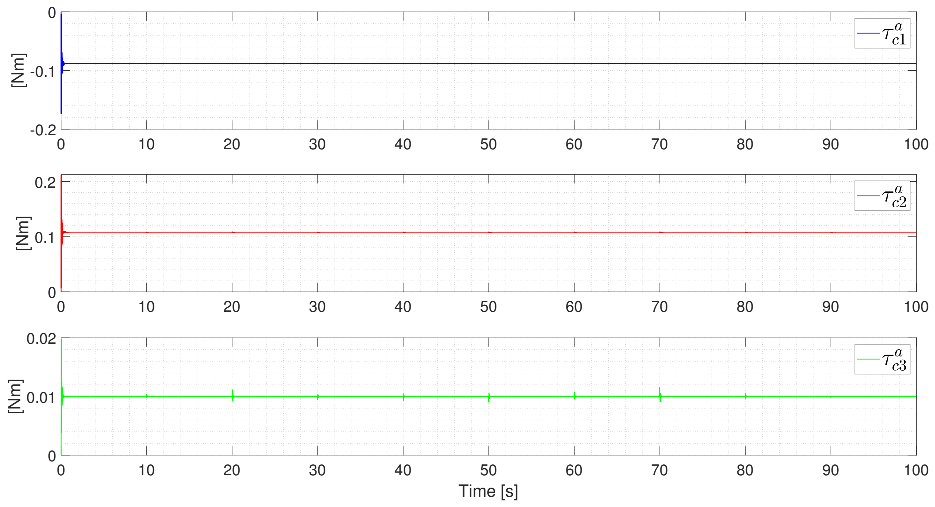

Figure 4.32: τ_c^a for the CMRAC architecture with $\gamma_a = 1$ and $\lambda_a = 10 \cdot I_3$

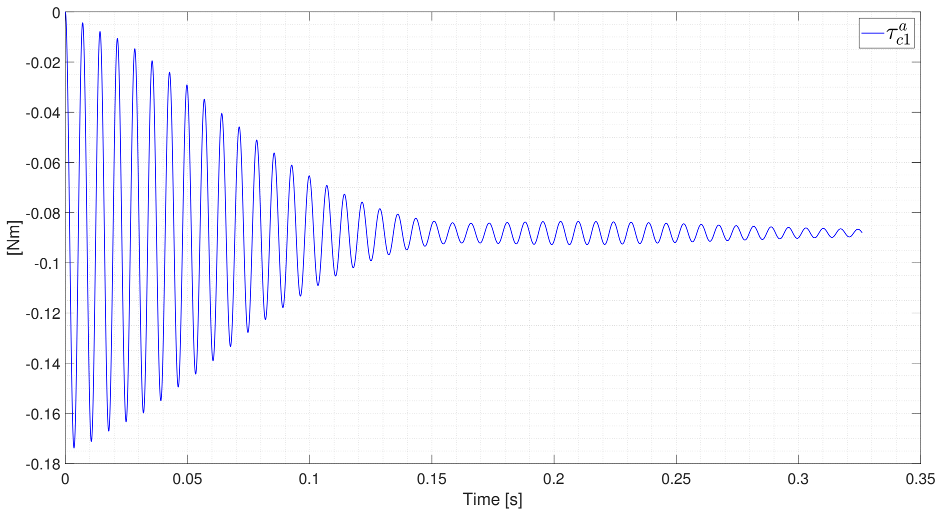

Figure 4.33: τ_{c1}^a Zoom for the CMRAC architecture with $\gamma_a = 1$ and $\lambda_a = 10 \cdot I_3$ to highlight oscillations

frequency oscillations are still present in the control input but with an important difference with respect to the previous case: considering the fact that γ_a is large, the vibrations are not as fast as in the MRAC architecture. This to say that the new degree of freedom λ_a seems to have the expected effect. Therefore, in general, it is possible to say that an high adaptation rate decreases the convergence time but the system becomes stiffer and the oscillations increase. In order to reduce them, we have to choose an higher λ_a. From the simulations, as will be better shown in the next attempts, it has been noticed that the closed-loop gain has to be chosen a couple of order of magnitude bigger than γ_a. Of course there is a limit. In fact, increasing too much γ_a and therefore λ_a, the system response becomes too fast for the actuators generating instability. In this optic, it is possible to see that the first initial transient of τ_c^a showed in figure 4.33 has a duration of some cents of seconds, which is a quite fast dynamic for the propellers. Moreover, the CPU effort has been discovered to be quite high since the presence of the oscillations requires an increased number of points to be simulated.

$$\boldsymbol{\gamma_a = 1000 \quad , \quad \lambda_a = 9 \times 10^4 \cdot I_3}$$

In this case, $\gamma_a = 1000$ and $\lambda_a = 9 \times 10^4 \cdot I_3$. These values are chosen in such a way to show one possible extreme case that can be achieved. As mentioned before, the difference between the two coefficients is little less than two orders of magnitude.

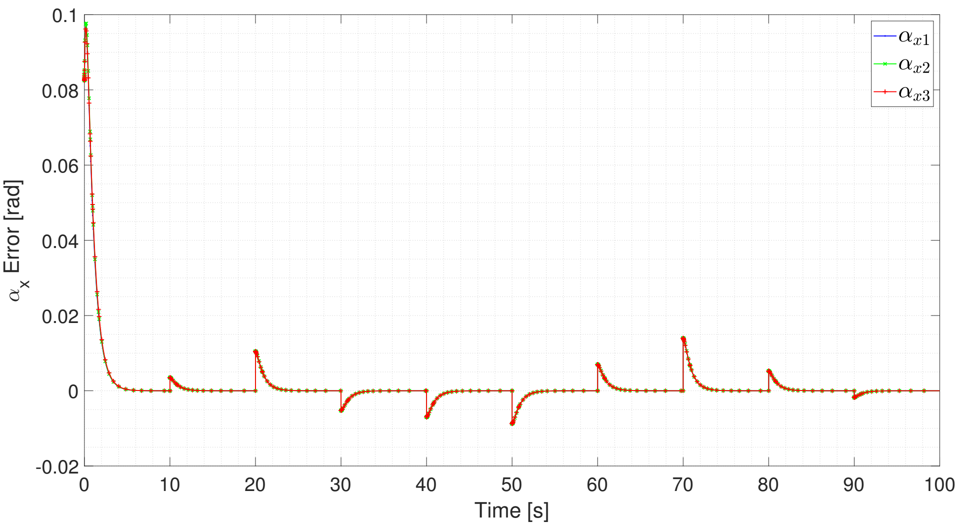

Figure 4.34: $\alpha_x = \alpha - \alpha_d$ error for the CMRAC architecture with $\gamma_a = 1000$ and $\lambda_a = 9 \times 10^4 \cdot I_3$

Referring to figure 4.34, one should note that the convergence error does not improve very much with respect to the other case and the CPU effort becomes very high, since the simulator has to deal with a very fast dynamic. Therefore,

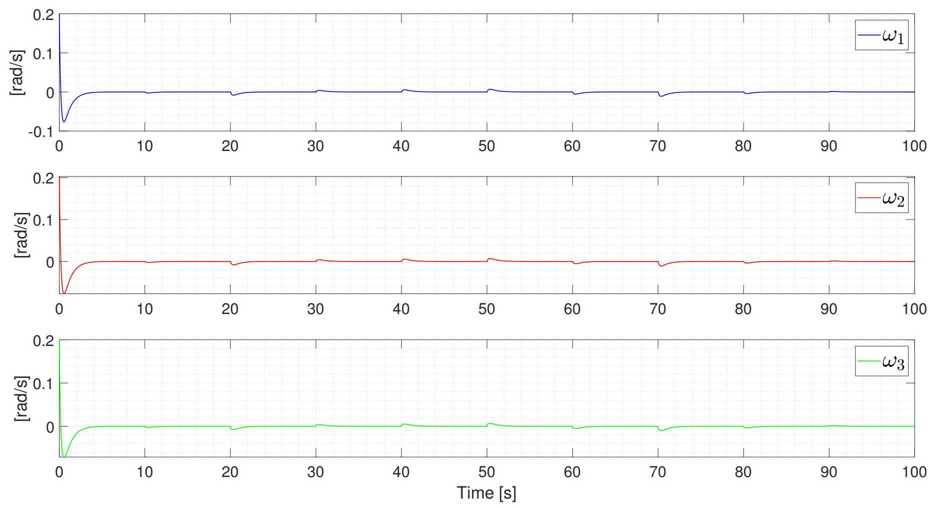

Figure 4.35: ω for the CMRAC architecture with $\gamma_a = 1000$ and $\lambda_a = 9 \times 10^4 \cdot I_3$

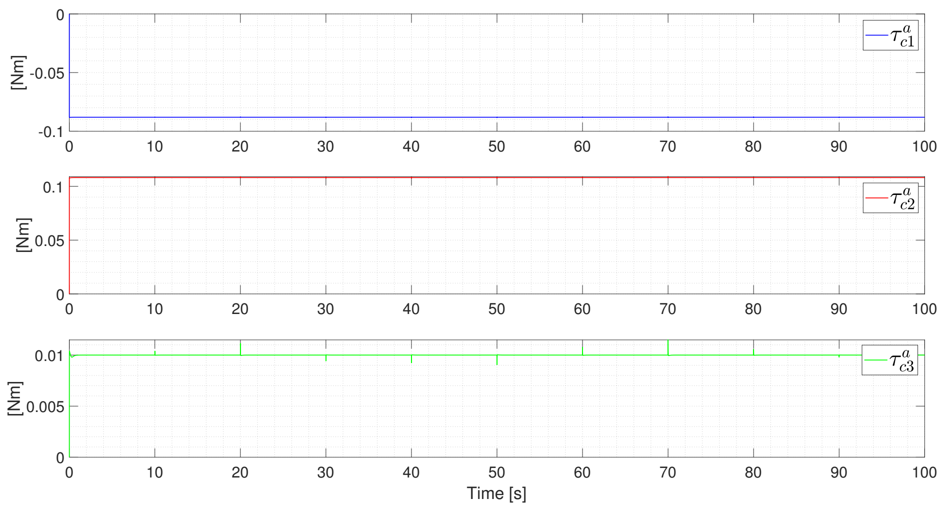

Figure 4.36: τ^a_c for the CMRAC architecture with $\gamma_a = 1000$ and $\lambda_a = 9 \times 10^4 \cdot I_3$

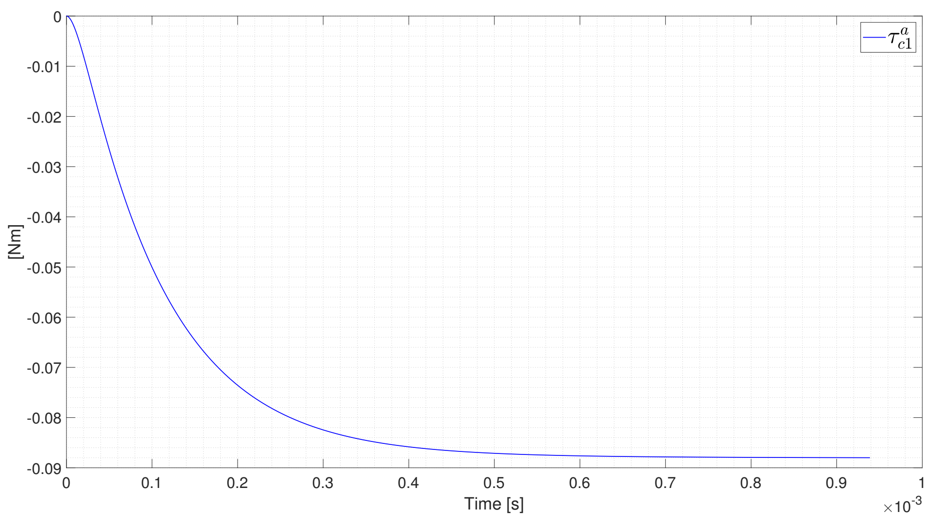

Figure 4.37: τ^a_{c1} Zoom for the CMRAC architecture with $\gamma_a = 1000$ and $\lambda_a = 9 \times 10^4 \cdot I_3$ to highlight oscillations

it does not make sense to increase too much the adaptation gain. On the other hand, considering that λ_a is very large, the oscillations in τ^a_c illustrated in figures 4.36 and 4.37 are completely vanished. The only problem is that the transient has a duration in the order of 10^{-3} seconds, which is really too fast for the actuators. This can be solved by taking a smaller adaptation gain that, as we have seen up to now, is related to the dynamic speed. As final remark, it is also possible to notice that both the angular velocity and the adaptive control input do not feel too much the step changes. They show a small and fast transient every 10 seconds but they are in the correct ranges for small scale UAVs.

$$\boldsymbol{\gamma_a = 0.001 \quad , \quad \lambda_a = 0.001 \cdot I_3}$$

As opposite case, we study what happens if we take the two gains under study with a small value: $\gamma_a = 0.001$ and $\lambda_a = 0.001 \cdot I_3$. As before figures 4.38, 4.39 and 4.40 show respectively the α_x, ω and τ^a_c plots.

Since the adaptation gain is small, it was discovered that λ_a has not to be two orders of magnitude bigger to cancel out all the oscillations. In fact, as also seen in the previous sections, a smaller γ_a generates less vibrations and therefore, a reduced closed-loop gain is enough. However, it is possible to see that the convergence of the attitude angles is very slow. It takes at least 20 seconds to bring the tracking error to zero. This to say that, different from the previous case in which the gains were too large, also too small values do not give satisfactory performances. Even if the CPU effort is low since the system dynamic is slow. Instead, the controller input shows a very smooth behaviour. It presents some oscillations but the transient has a duration of 1 second. It has a dynamic three

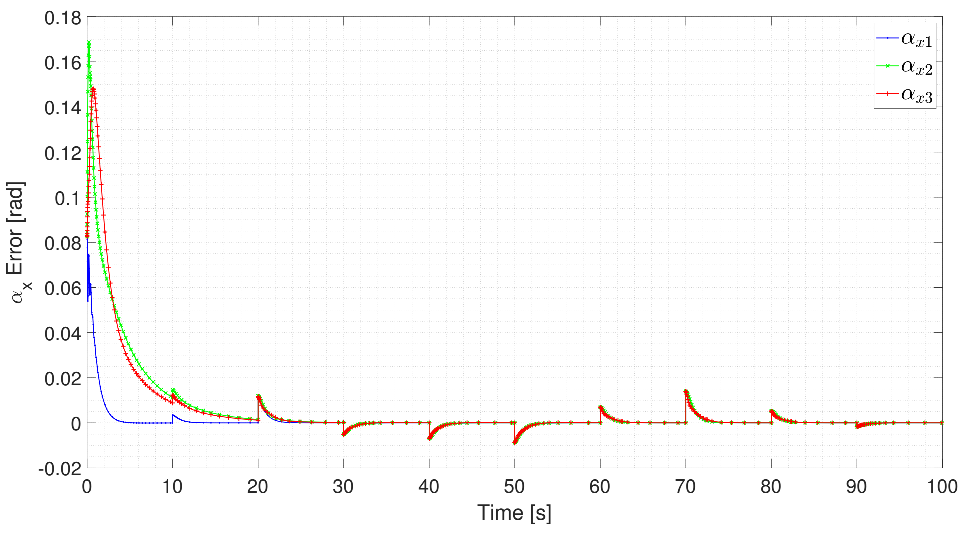

Figure 4.38: $\alpha_x = \alpha - \alpha_d$ error for the CMRAC architecture with $\gamma_a = 0.001$ and $\lambda_a = 0.001 \cdot I_3$

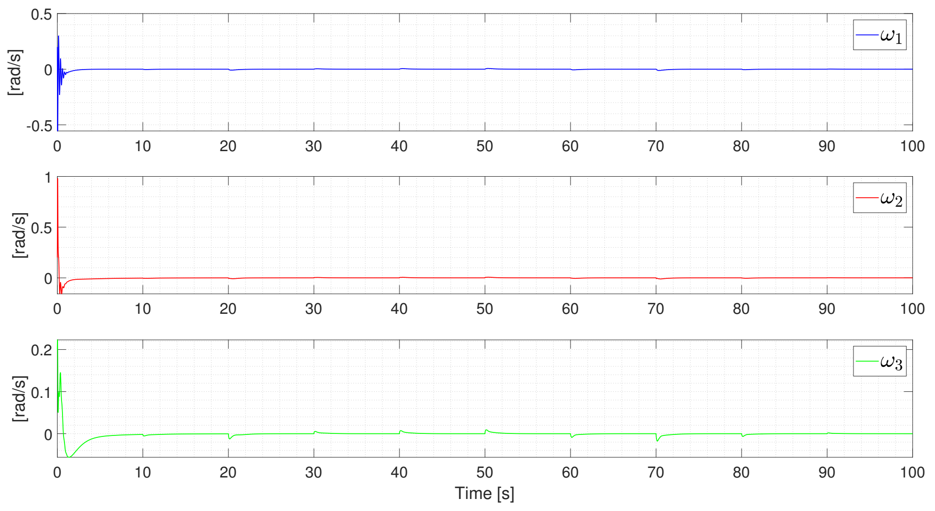

Figure 4.39: ω for the CMRAC architecture with $\gamma_a = 0.001$ and $\lambda_a = 0.001 \cdot I_3$

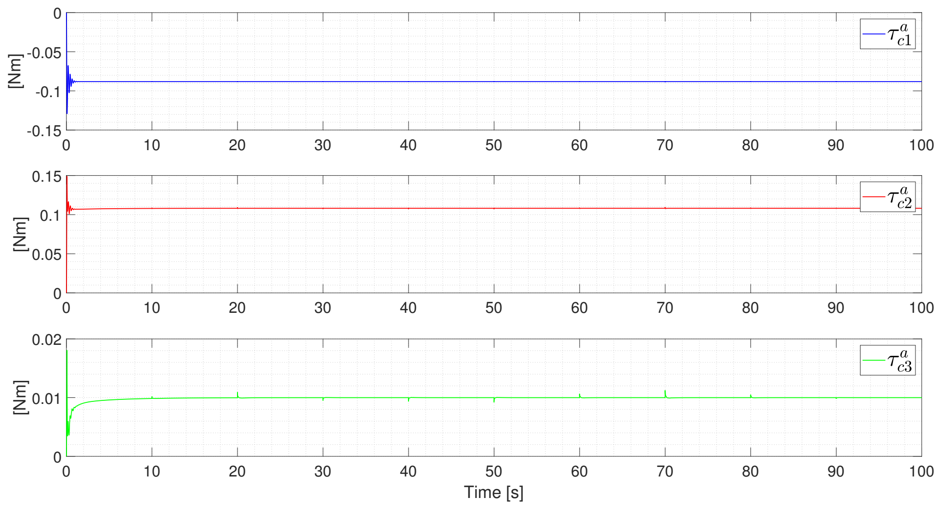

Figure 4.40: τ_c^a for the CMRAC architecture with $\gamma_a = 0.001$ and $\lambda_a = 0.001 \cdot I_3$

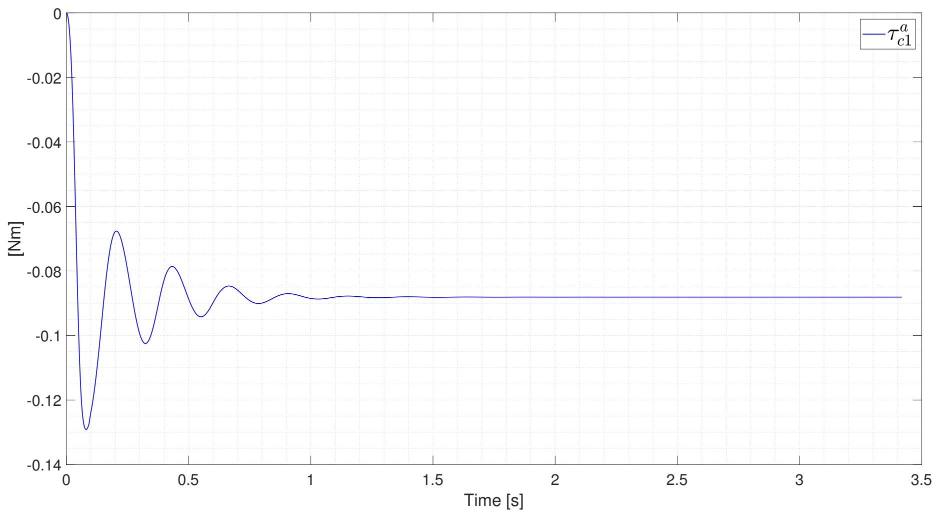

Figure 4.41: τ_{c1}^a Zoom for the CMRAC architecture with $\gamma_a = 0.001$ and $\lambda_a = 0.001 \cdot I_3$ to highlight oscillations

order of magnitude slower than the previous case. The same can be said regarding the angular velocity.

$$\boldsymbol{\gamma_a = 10 \quad , \quad \lambda_a = 9000 \cdot I_3}$$

In this section, the final results obtained with the CMRAC approach will be analysed. In particular, it has been chosen to set $\gamma_a = 10 \cdot I_3$ and $\lambda_a = 9000 \cdot I_3$. It should be noted that the latter is nearly three order of magnitude bigger than the adaptation gain. This choice comes from a trade off between speed of adaptation and consequently of convergence, number of high frequency oscillations in the control variables, duration of the transients and CPU effort.

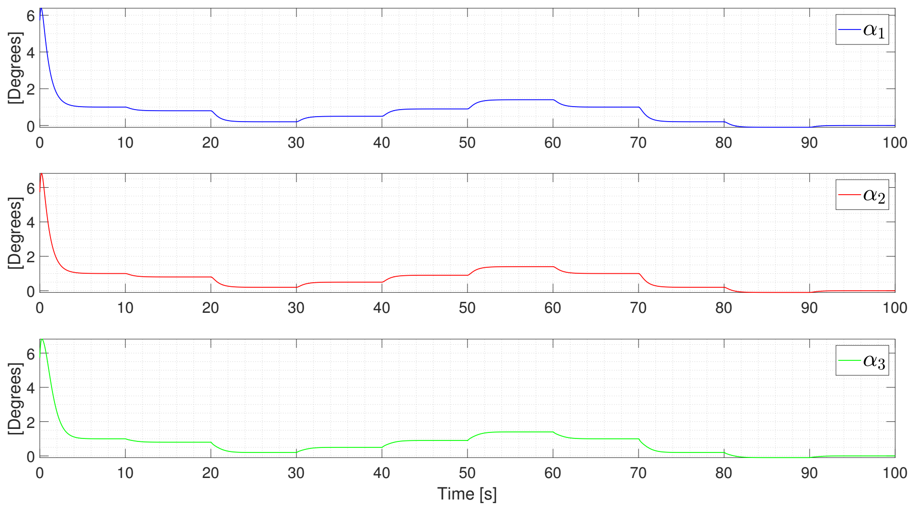

Figure 4.42: Attitude angles α for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

As it is possible to notice from figures 4.42 and 4.43, the attitude error shows a very good tracking performances. Also in the first step response, the transient has a duration of a maximum of 4 seconds. The roll, the pitch and the yaw angles present a smooth behaviour without overshoots. The same can be said about the angular velocity illustrated in 4.47. Regarding the position showed in figures 4.44 and 4.45, one can see that the behaviour is equivalent to the one illustrated in the MRAC approach. This is mainly due to the fact that, in all these tests, the position and consequently the velocity in figure 4.46 is set as ideal without any sort of uncertainties or disturbances. Furthermore, the adaptation acts only on the attitude dynamic and, since in the linear plant the position is decoupled from the attitude, it makes sense that the CMRAC shows the same results of the MRAC. Instead, concerning the baseline and the adaptive control inputs respectively showed in figures 4.48, 4.51 and 4.49, the transient oscillations

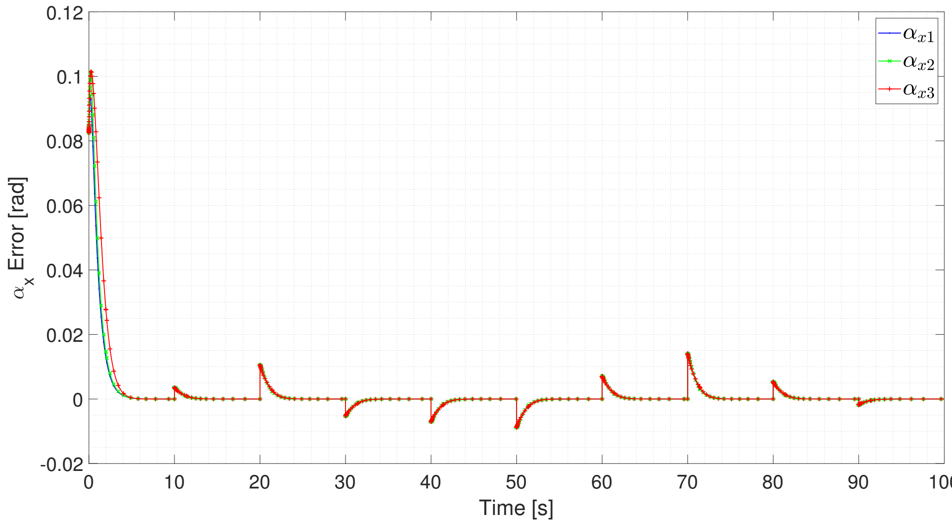

Figure 4.43: $\alpha_x = \alpha - \alpha_d$ error for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

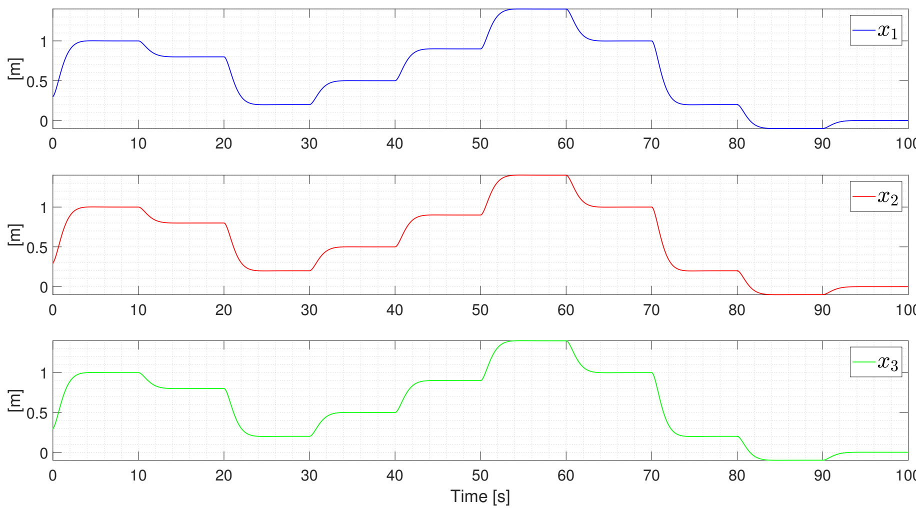

Figure 4.44: Position vector x for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

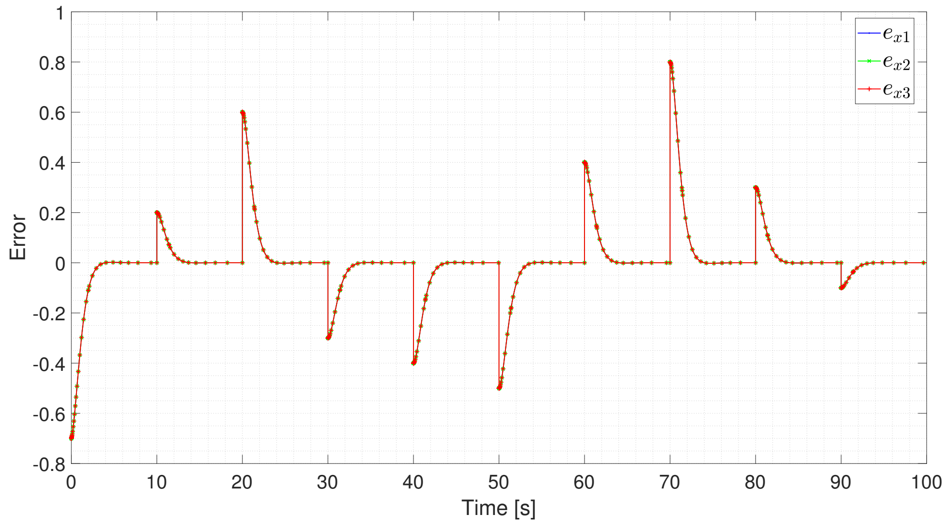

Figure 4.45: Position error $e_x = x - x_d$ for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

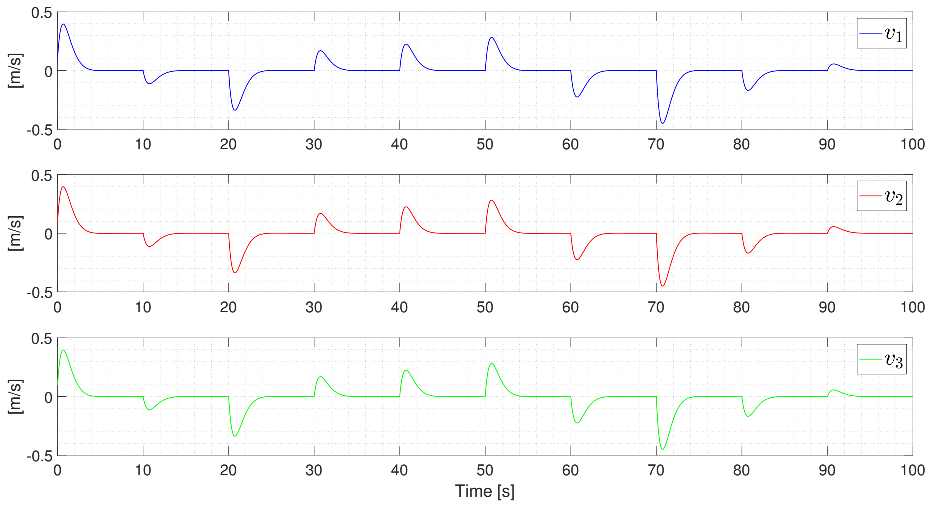

Figure 4.46: Velocity vector v for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

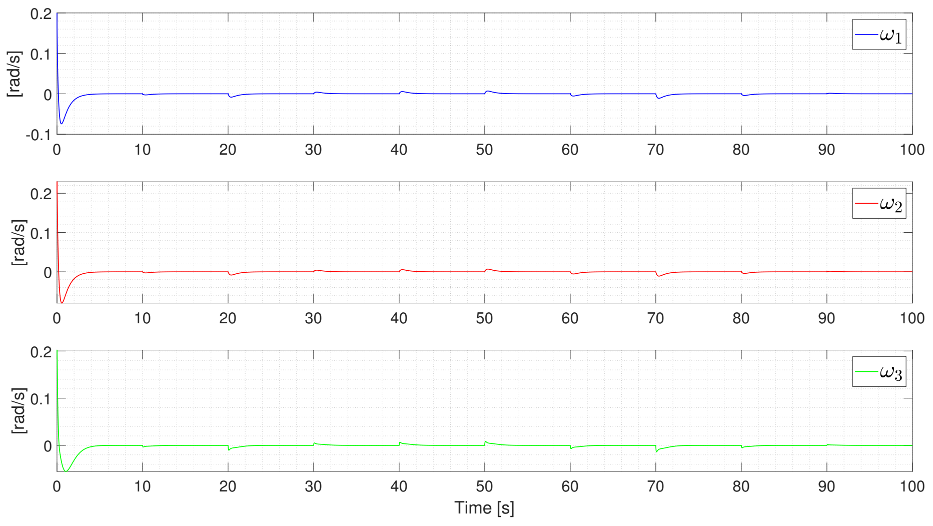

Figure 4.47: Angular velocity vector ω for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

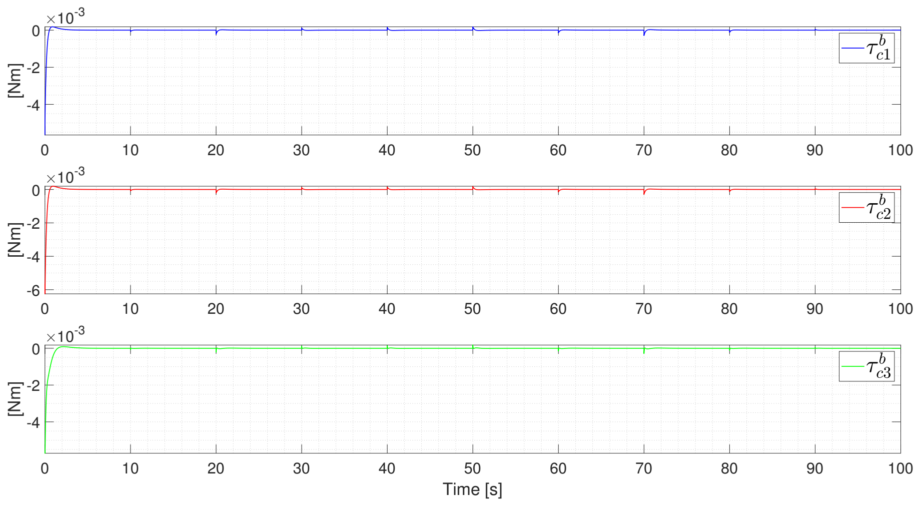

Figure 4.48: Baseline control torque τ^b_c for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

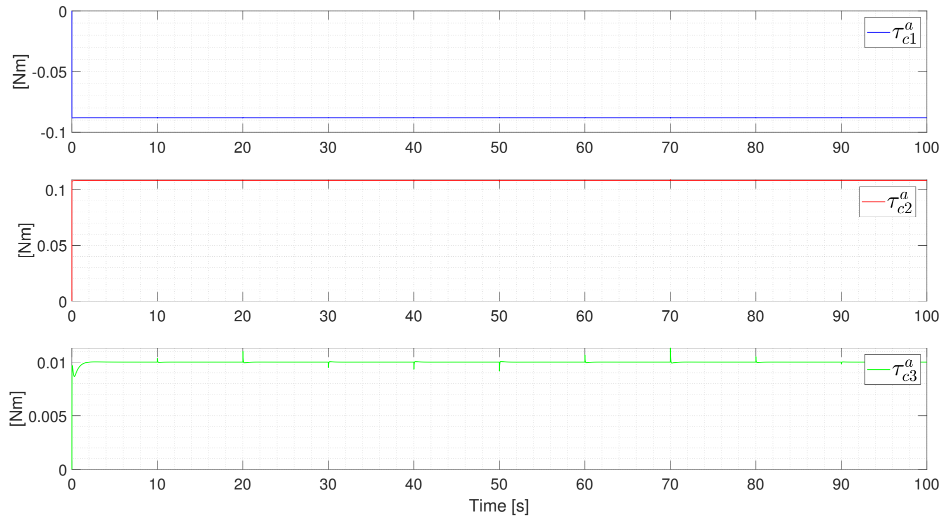

Figure 4.49: Adaptive control torque τ_c^a for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

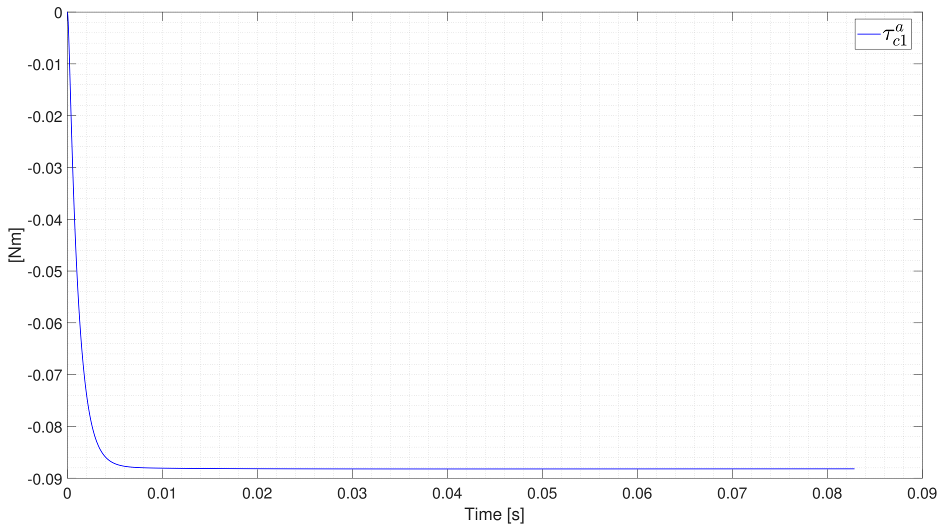

Figure 4.50: τ_{c1}^a Zoom for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

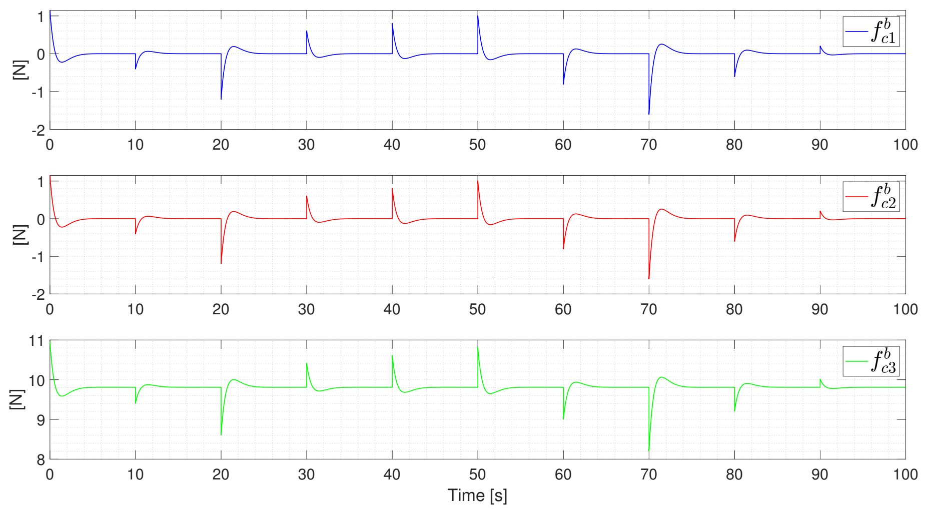

Figure 4.51: Baseline control force f^b_c for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

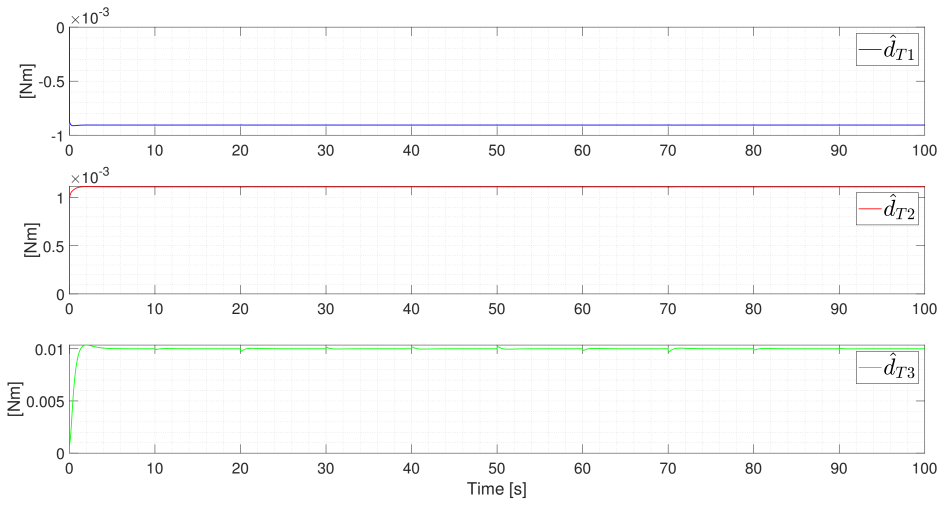

Figure 4.52: Estimation of the attitude disturbance vector d_T for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

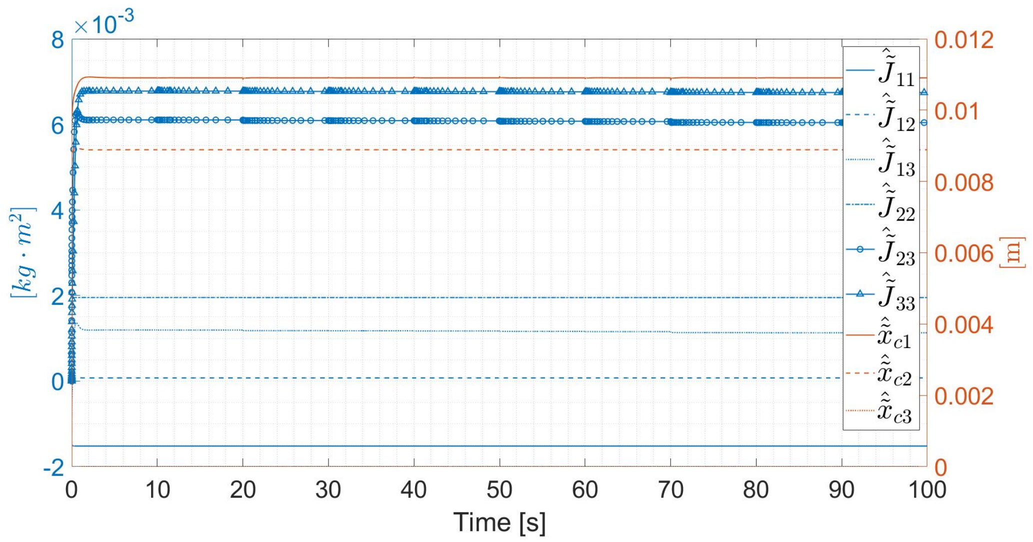

Figure 4.53: Estimation of the uncertain parameters contained in the θ_a vector for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

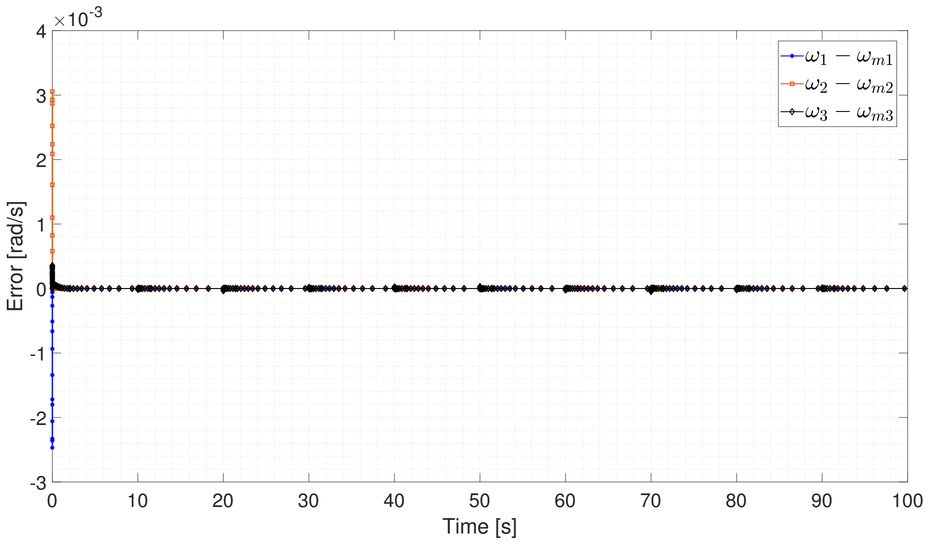

Figure 4.54: Error between the plant and reference model angular velocity ω and ω_m for the CMRAC architecture with $\gamma_a = 10$ and $\lambda_a = 9000 \cdot I_3$

are completely vanished, as better highlighted in 4.50, and they have a duration in the order of magnitude of 10^{-2} seconds, which is suitable for small scale UAV. Moreover, also the amplitudes are in the correct ranges. Regarding the estimation of d_T and θ_a respectively showed in figures 4.52 and 4.53, it is possible to notice that unlike the MRAC case, the variables converge very fast without showing any oscillatory behaviour. This helps the algorithm to better control the system without the presence of fast dynamics. In general, all the variables do not show very high picks during the step changes: they converge to a certain offset value and then they correct it a bit to permit the tracking. Finally, as in the MRAC case, in figure 4.54 the error between the angular velocity of the plant and the reference model is showed. As it is possible to see, after an initial transient, the tiltrotor follows very closely the reference system.

4.7 CMRAC: Verification considering non-linear dynamic effects

From the previous analysis, it should be clear that the most suitable controller for the tiltrotor is the CMRAC architecture, since it allows the designer to have an additional degree of freedom that can be exploited to reduce the possible presence of high frequency oscillations and to regulate the transient behaviours. As explained before, once the most satisfactory controller has been selected, it has to be proven on each time more complex numerical environment. For this reason, the CMRAC architecture has been implemented in the simplified simulator showed in section 4.5 but with the non linear plant reported in equation (1.11). In particular, in this section a comparison between the results obtained with the linear and the non-linear dynamic will be showed. For simplicity, unlike the other cases, the desired set point has been chosen as an elementary single step:

$$\begin{cases} x_{des} = [0.1 \quad 0.05 \quad 0.2]^T \cdot \text{step}(t-10) \quad \text{m} \\ \alpha_{des} = [5 \quad 4 \quad -3]^T \cdot \text{step}(t-10) \quad \text{degrees} \end{cases} \tag{4.12}$$

It has also to be noted that, in order to ensure the equality between the linear and the non-linear model, in this latter τ_c^b of the plant and of the reference model has been multiplied by a factor of 2. This is due to the fact that, in the linearisation, $q \approx \frac{1}{2}\alpha$. Instead, all the other parameters have been left identical to the previous attempt.

Only in this section, in order to allow the reader to better understand to which model we are referring to, the variables will be enhanced with the superscript “L” or “N” if they are identified respectively by the linear or the non-linear simulation.

Referring to the attitude tracking error and the Euler angles step response showed respectively in figures 4.55 and 4.56, one should note that the behaviour of the linear and the non-linear model is very similar. This is very important because it means that the CMRAC adaptive controller is working well. It is able

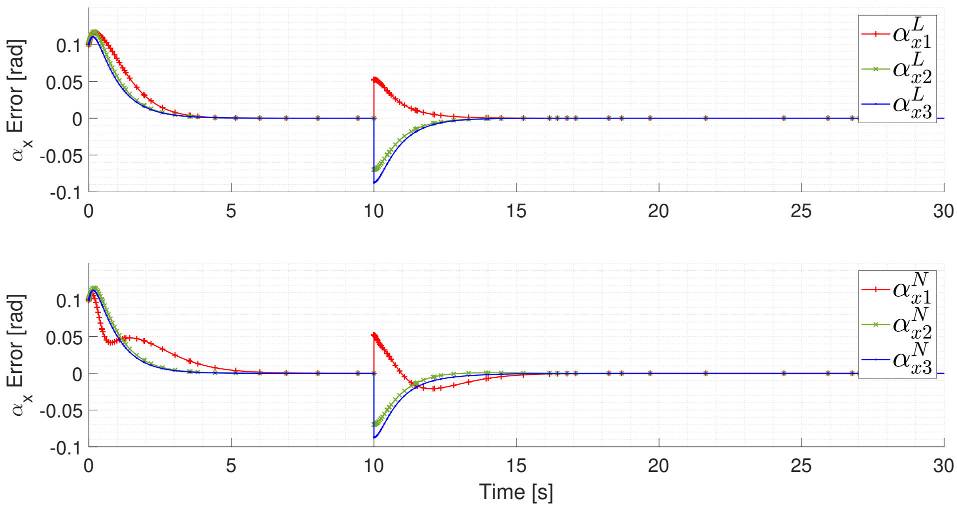

Figure 4.55: Attitude tracking error α_x for the linear (on top) and the non-linear (bottom) model

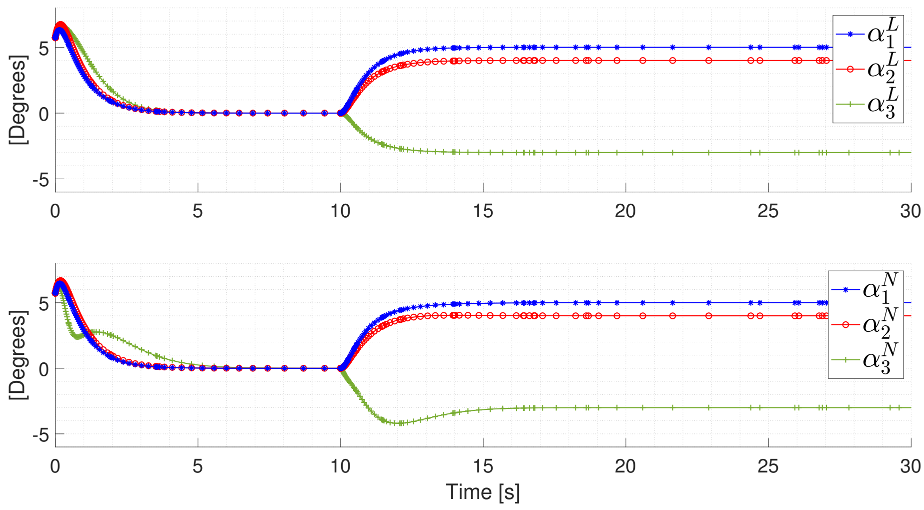

Figure 4.56: Euler angles α_i for the linear (on top) and the non-linear (bottom) model

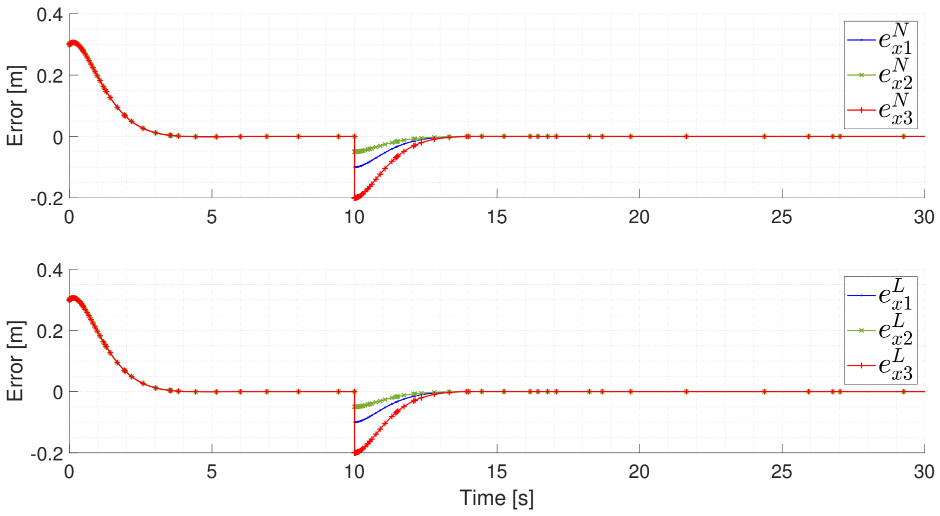

Figure 4.57: Position tracking error e_x for the non-linear (on top) and the linear (bottom) model

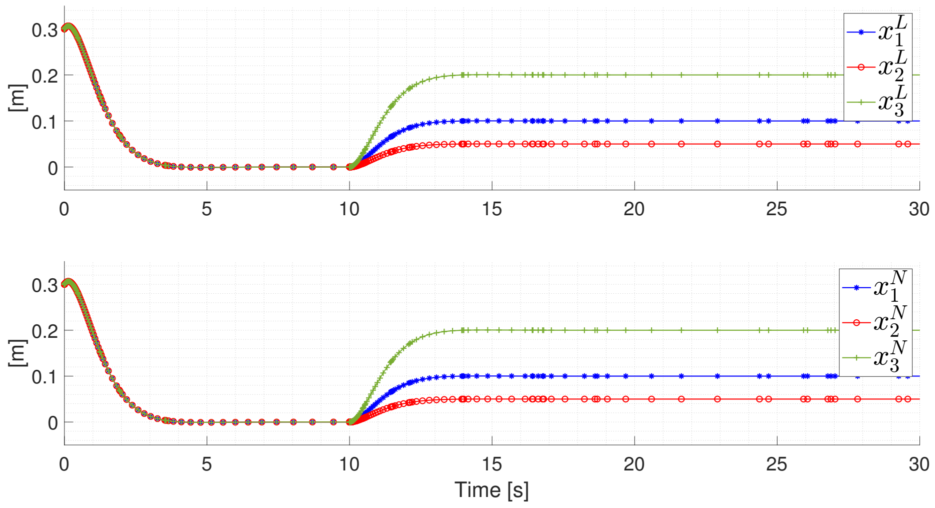

Figure 4.58: Position vector x for the linear (on top) and the non-linear (bottom) model

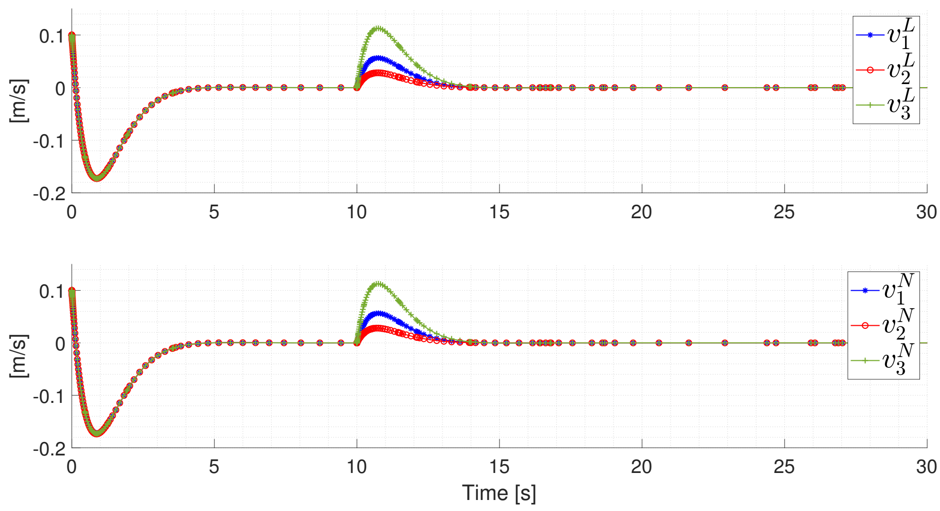

Figure 4.59: Translational velocity vector v for the linear (on top) and the non-linear (bottom) model

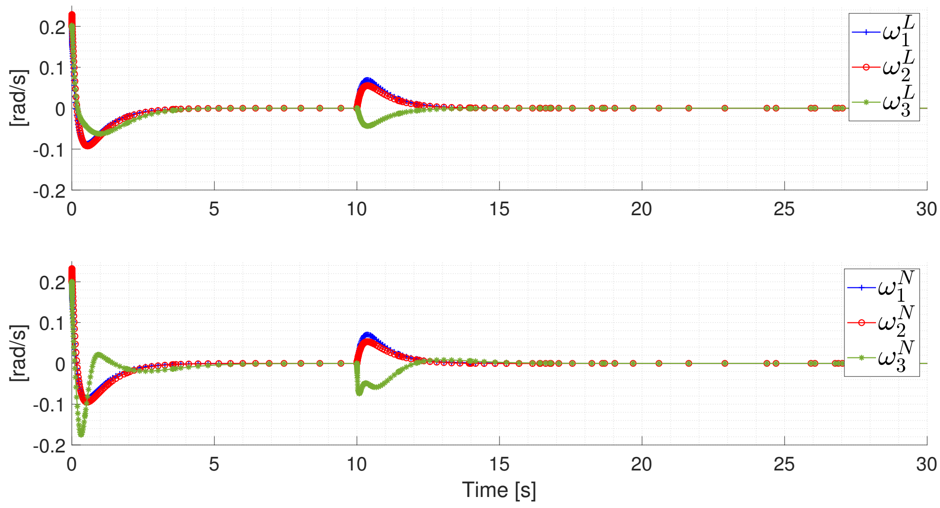

Figure 4.60: Angular velocity ω for the linear (on top) and the non-linear (bottom) model

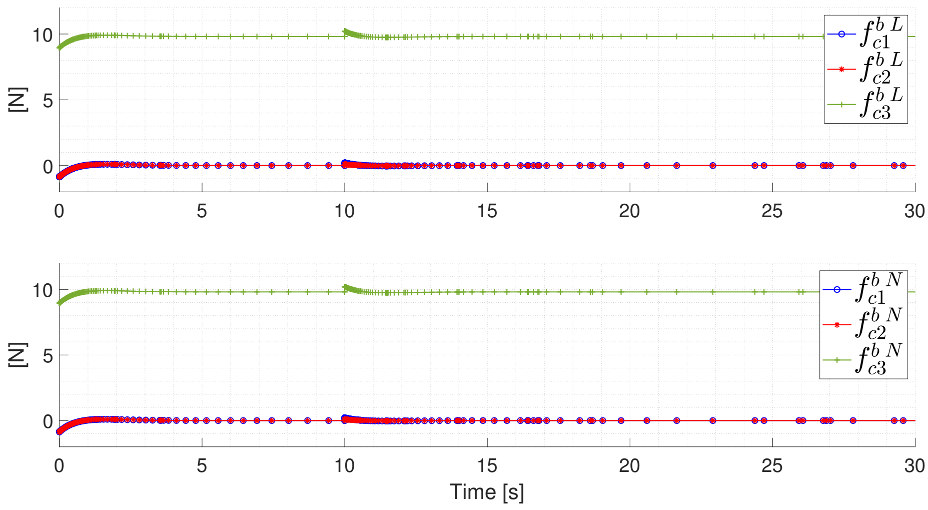

Figure 4.61: Baseline force control input f_c^b for the linear (on top) and the non-linear (bottom) model

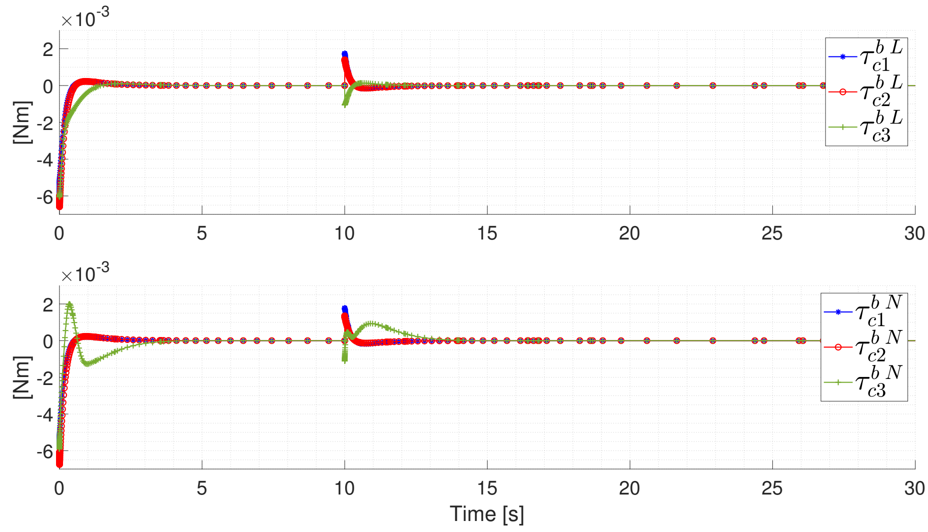

Figure 4.62: Baseline torque control input τ_c^b for the linear (on top) and the non-linear (bottom) model

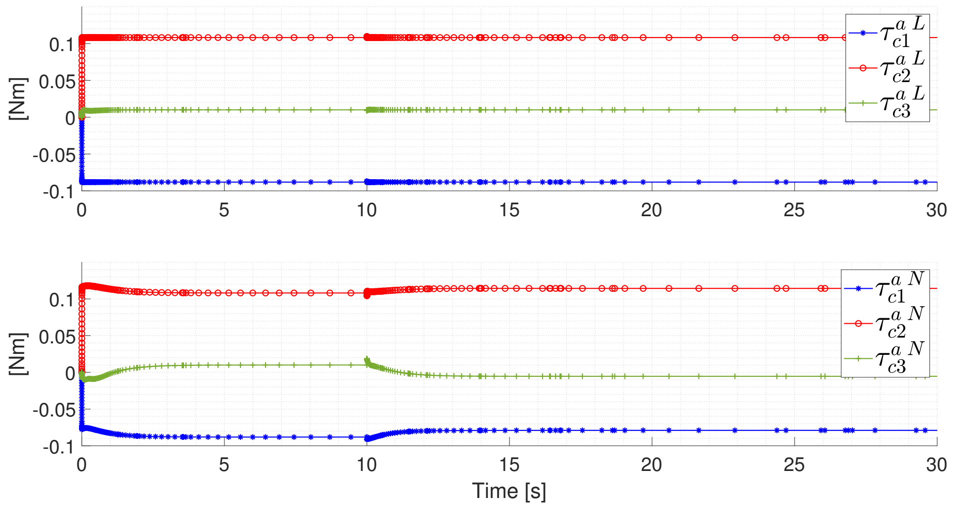

Figure 4.63: Adaptive torque control input τ_c^a for the linear (on top) and the non-linear (bottom) model

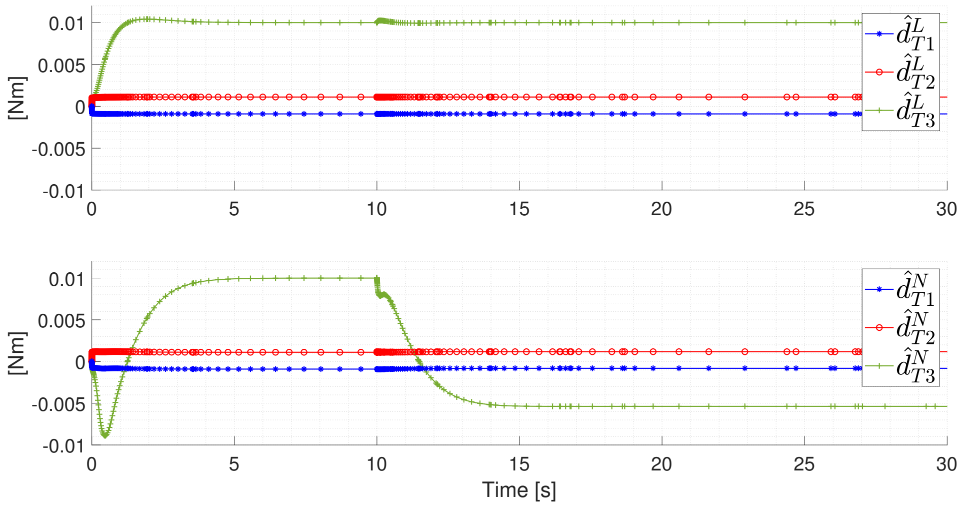

Figure 4.64: On-line estimation of the disturbance vector d_T for the linear (on top) and the non-linear (bottom) model

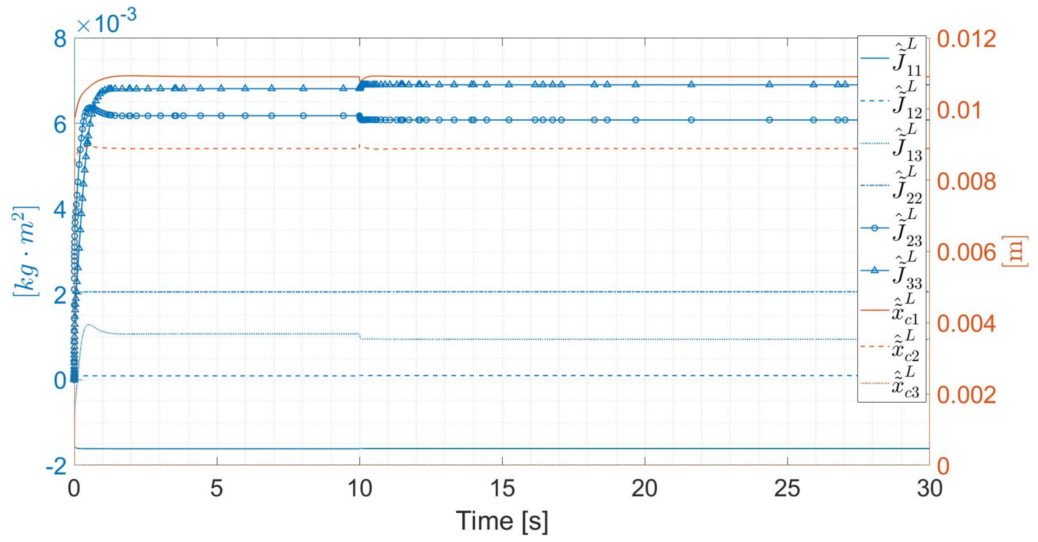

Figure 4.65: On-line estimation of the uncertain parameters contained in θ_a for the linear model

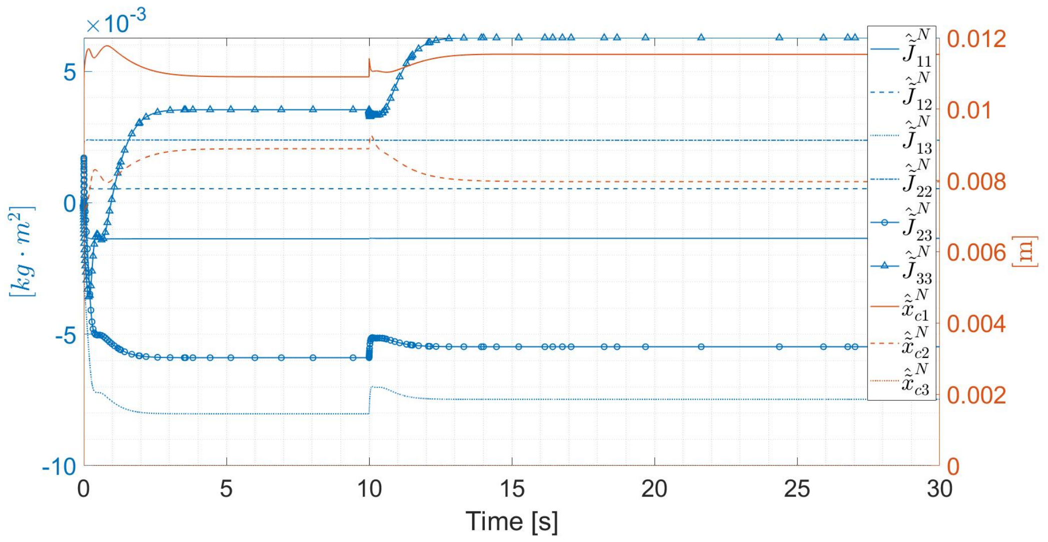

Figure 4.66: On-line estimation of the uncertain parameters contained in θ_a for the non-linear model

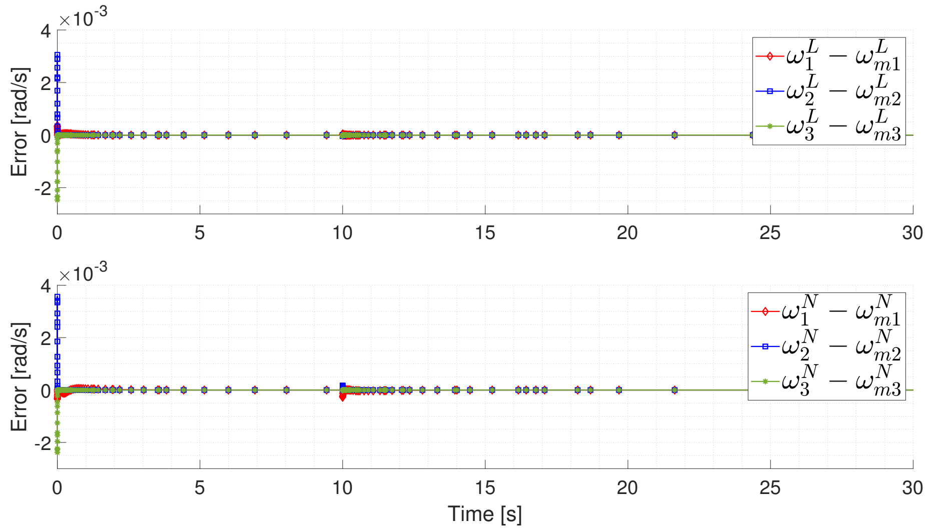

Figure 4.67: Error between the plant and the reference model angular velocity for the linear (on top) and the non-linear (bottom) model

to reject all the model non-linearities and bring the response very close to the linear case. In particular, the pitch and the yaw angles are equivalent, instead of the roll that presents a small overshoot in the non-linear plant. As regards to the position illustrated in figures 4.57 and 4.58, the response is more or less equivalent. The same goes for the translational and the angular velocities in 4.59 and 4.60. Regarding the control inputs, as in all the previous cases, the control force main contribution comes from the third component, as showed in 4.61, since it has to compensate the gravity force. Instead, the torque control inputs illustrated in 4.62 and 4.63, both in the linear and non linear-case, do not present high frequency oscillations. Therefore, also maintaining the same gains and increasing the complexity of the simulator, the adaptive controller is still able to control the system with good transient performances. Concerning the estimation of the uncertain parameters such that d_T showed in 4.64, and θ_a showed in 4.65 and 4.66, it is possible to notice that some of the variables converge to different values. For example $\hat{d}_{T3}$, which in the linear case converges to 0.001 and in the non-linear one, after that the step occurs, it converges to -0.005. A similar behaviour can be also seen in the estimation of the inertia tensor and center of mass components. Finally, in figure 4.67, as in the previous sections, the angular velocity error with respect to the reference model is reported. Even in the presence of the non-linearities, the CMRAC has showed to be very good in following the ideal ω_m. The only difference is the presence of intenser transients.

4.8 Tilt-arm simulator

In this section, the selected CMRAC approach will be verified on the attitude dynamic trough the use of the complete simulator presented in chapter 1. As before the position dynamic has been left ideal, while opposite to all the other cases, the data used in these simulations respect an existing tiltrotor prototype. As previously said, the CPID architecture has been chosen as the reference system with respect to which all the results have to be compared. In fact, its performances respects many currently state of the art approaches for the problem of controlling a tiltrotor. Therefore, the idea is to try to choose λ_a and γ_a such that the results will be better or at least comparable with the ones given from the CPID, taking into consideration of possible high frequency oscillations that can arise. In fact, for this simulation the adaptive laws have been rewritten as:

$$\dot{\hat{\theta}}_a = \gamma_{\theta_a} Proj\left(\hat{\theta}_a, W_a^T B_a^T P_a e_a\right) \qquad \dot{\hat{d}}_T = \gamma_{dist} Proj\left(\hat{d}_T, B_a^T P_a e_a\right) \tag{4.13}$$

where $\gamma_{\theta_a} = [\gamma_{inertia}, \quad \gamma_{CoM}]$ in which $\gamma_{inertia} \in \mathbb{R}^6$ and $\gamma_{CoM} \in \mathbb{R}^3$ are the adaptation gains that respectively refer to the inertia and center of mass components of the uncertain vector θ_a. Instead, $\gamma_{dist} \in \mathbb{R}^3$ is the adaptation gain that acts on the estimation of the disturbance d_T. For simplicity, all these contributions can be collected in a single vector as:

$$\gamma_A = [\gamma_{inertia}, \quad \gamma_{CoM}, \quad \gamma_{dist}] \quad \in \mathbb{R}^{12}. \tag{4.14}$$

For these particular simulations it has been chosen to set:

$$\begin{cases} \gamma_{inertia} = [0.03, \quad 0.03, \quad 0.03, \quad 0.03, \quad 0.03, \quad 0.03] \\ \gamma_{CoM} = [0.09, \quad 0.09, \quad 0.09] \\ \gamma_{dist} = [12, \quad 12, \quad 6] \end{cases} \tag{4.15}$$

Instead, the closed-loop gain has been chosen as follow:

$$\lambda_a = \begin{bmatrix} 35 & 0 & 0 \\ 0 & 35 & 0 \\ 0 & 0 & 40 \end{bmatrix} \tag{4.16}$$

The initial conditions used to integrate (4.13) are reported in A.3. Concerning the desired trajectory, the Paoll trajectory showed in 4.1 has been selected to evaluate the attitude performances. Moreover, the values of the center of mass position has been left equal to the one showed in table 4.3, while $\bar{J}$ has been taken as a diagonal matrix which components differ by 20% from the J diagonal elements. As a reminder, for a complete overview of the simulator data one should refer to appendix A. Unlike the previous cases, the performances will not be only evaluated by looking directly at the plots, but also by looking at similar tables to the ones presented in 4.3, in which the maximum absolute error (4.5) and root mean square error (4.6) will be reported for both the CPID and the CMRAC adaptive architectures. This is to permit the comparison between the two different approaches.

For a better evaluation of the tiltrotor behaviour, the simulations will be split in two different cases:

- A simulation without disturbances: $d_T = [0, \quad 0, \quad 0]$
- A simulation with a step disturbance: $d_{T1} = d_{T2} = d_{T3} = 0.1 \cdot \text{step}(t-7)$

The first case can help to understand the UAV tracking performances. In fact, since we are looking at the absolute and root mean square errors, a sudden disturbance would rapidly increase the error amplitudes giving distorted results, more precisely a bigger e_{abs} and e_{rms}, which do not reflect the actual tracking performances. Instead in this context, the second case permits to separate the evaluation of the disturbance rejection capability of the system. Indeed, since the root mean square error is a sort of average of the error in time, it is not the most correct parameter to assess the disturbance rejection capacity. For this reason, in this second simulation, only the absolute error will be computed. The sudden d_T step will increase the maximum absolute error and it will be bigger the less the two approaches under study are able to oppose to the disturbance.

4.8.1 Simulation without disturbances

As previously anticipated, in order to have a better understanding of the UAV tracking performances, the simulations have to be carried out considering null disturbances. Figures 4.68, 4.69, 4.70, 4.71, 4.72, 4.73, 4.74, 4.75, 4.76, 4.77, 4.78, 4.79, 4.80, 4.81, 4.82 and 4.83 show the results obtained.

Tables 4.6 and 4.7 report respectively the maximum absolute and root mean square errors defined in (4.5) and (4.6) for all the state components. In contrast to the errors illustrated in the figures, where they are calculated with respect to the outer-loop variables ω_v and v_v, in the tables they have been computed as the difference between the plant state and the one computed by the trajectory generator mentioned in section 1.4. In this way, the reference is always the same for all the simulations and this permits a better comprehension of the data. Therefore, it is possible to define $v_x = v - v_d \in \mathbb{R}^3$ and $\omega_x = \omega - \omega_d \in \mathbb{R}^3$ where v_d and ω_d are respectively the desired translational and angular velocity computed by the trajectory generator. These new variables are the ones that are actually used in the tables.

Referring to the outcomes of the numerical simulation, the adaptive architecture shows better performances both for the maximum absolute and the root mean square error. It should be reminded that the adaptation is acting only on the attitude dynamic but, since the plant implemented is non-linear and it is affected by some couplings between attitude and position, this latter also presents

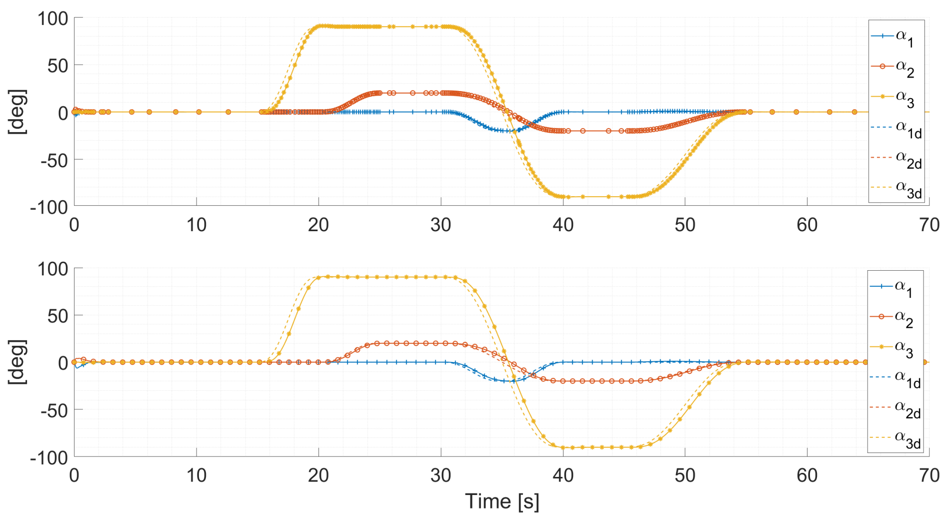

Figure 4.68: Plant and desired Euler angles for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

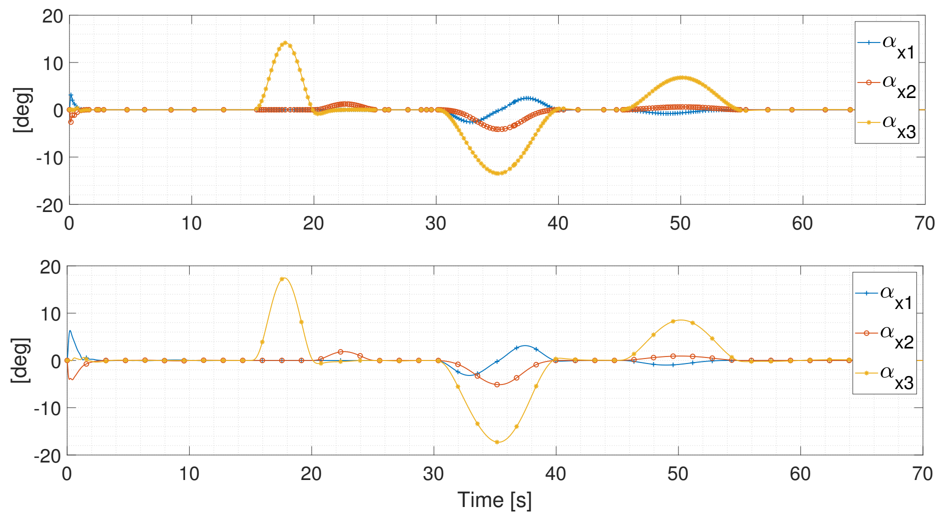

Figure 4.69: Euler angles error α_x for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

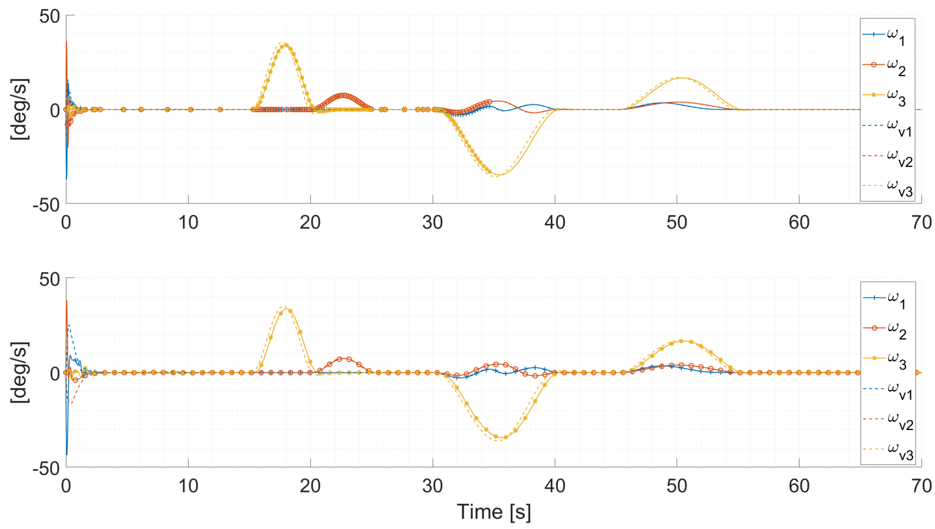

Figure 4.70: Plant and desired angular velocities for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

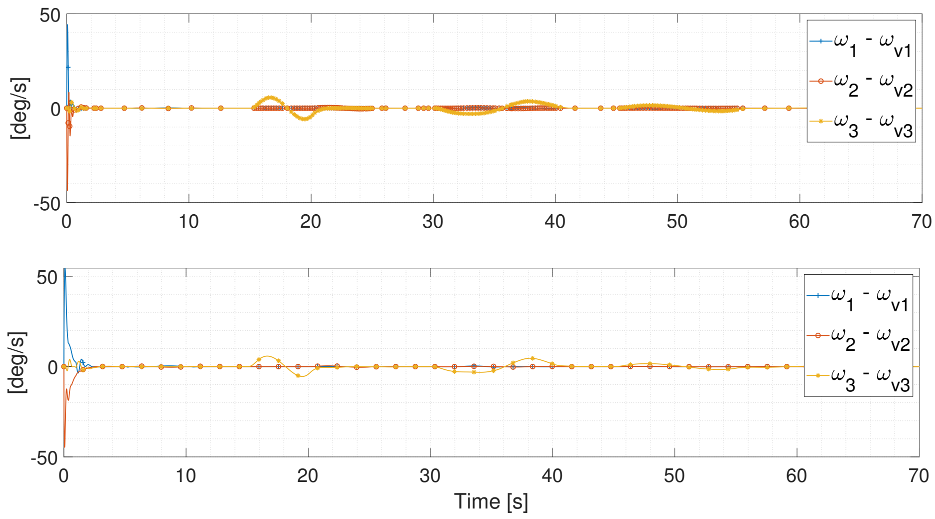

Figure 4.71: Angular velocity error with respect to the outer-loop ω_v for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

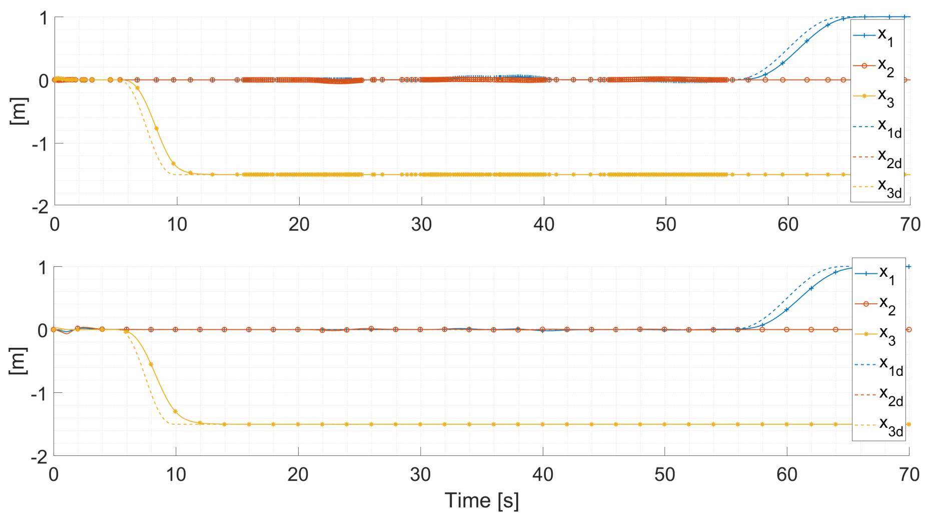

Figure 4.72: Plant and desired position for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

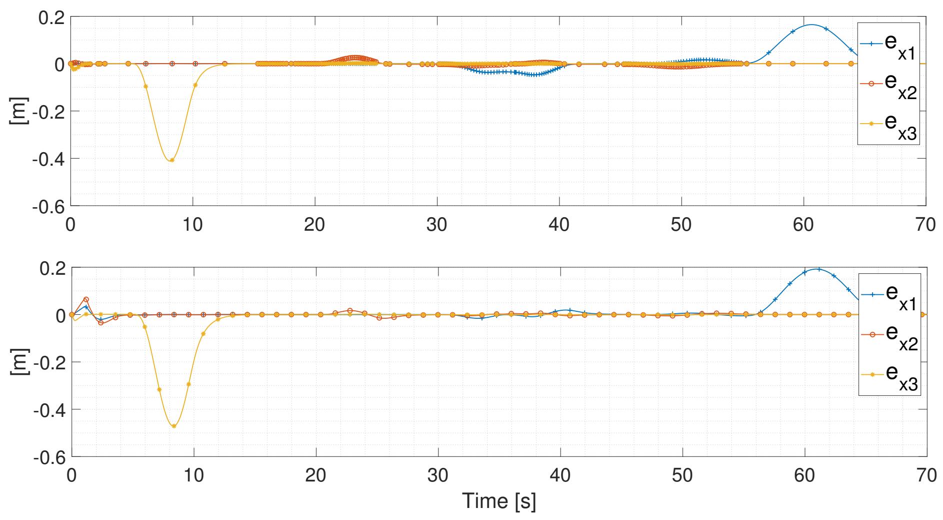

Figure 4.73: Position error e_x for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

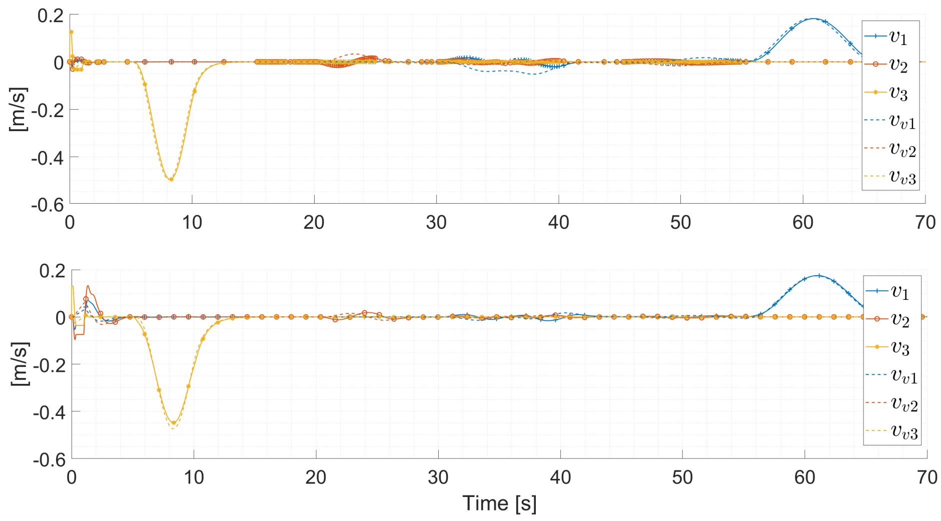

Figure 4.74: Plant and desired translational velocity for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

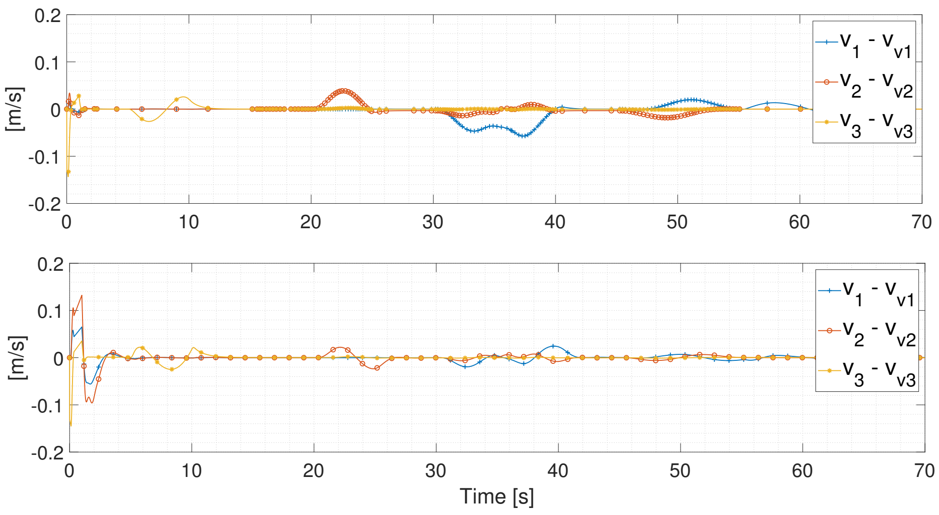

Figure 4.75: Translational velocity error with respect to the outer-loop v_v for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

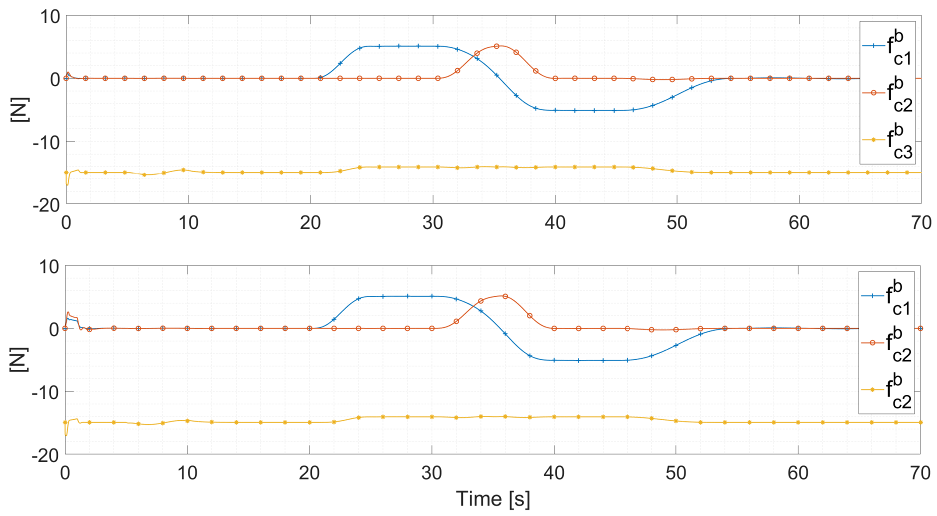

Figure 4.76: Force control input $f_c = f_c^b + f_c^a$ for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

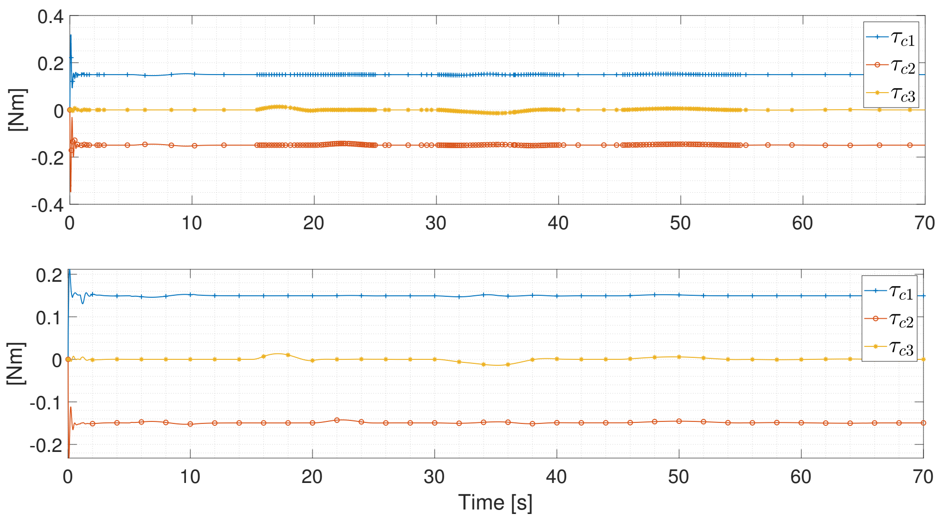

Figure 4.77: Torque control input $\tau_c = \tau_c^b + \tau_c^a$ for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

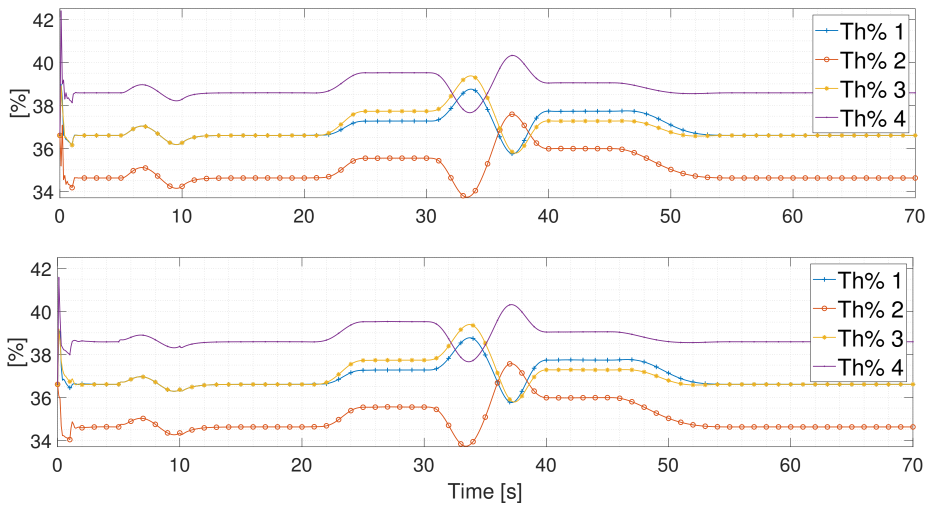

Figure 4.78: $Th\%$ requested by the four UAV's propellers for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

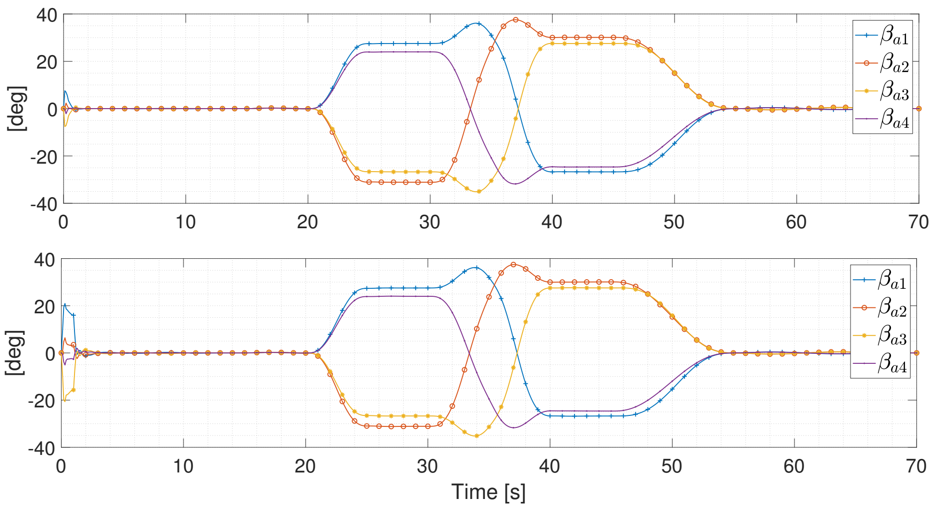

Figure 4.79: Servo-actuators angular position β_a requested by the four UAV's propellers for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation without disturbance.

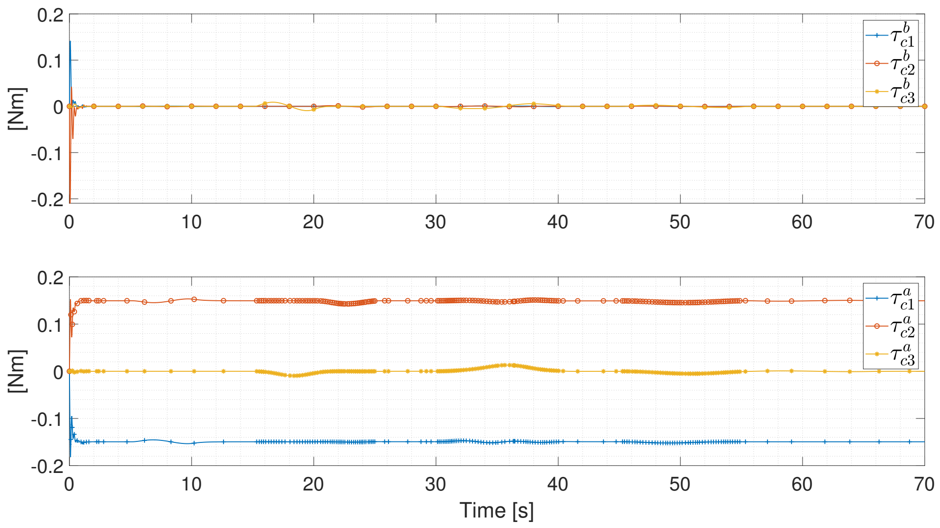

Figure 4.80: Baseline (on top) and adaptive (bottom) control input torque for the model with the CMRAC adaptive controller. Simulation without disturbance.

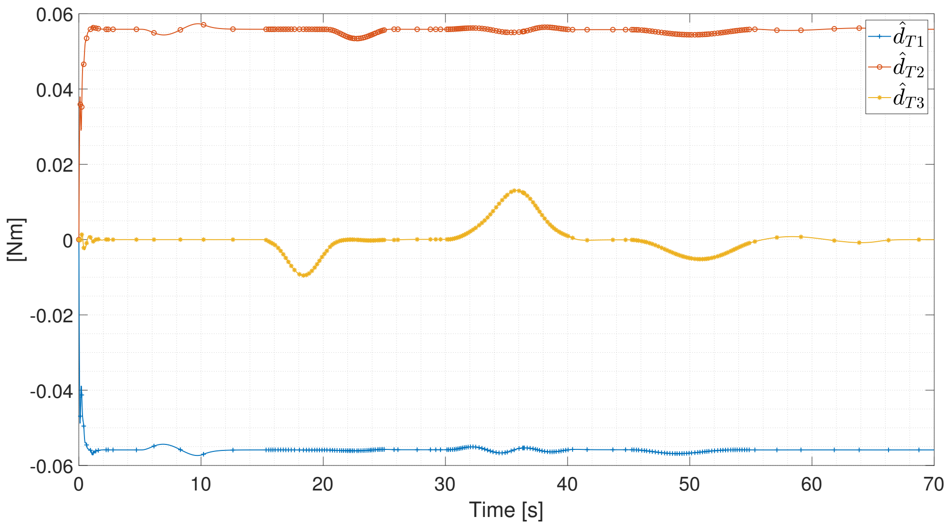

Figure 4.81: On-line estimation of the disturbance $\hat{d}_T$ for the model with the CMRAC adaptive controller. Simulation without disturbance.

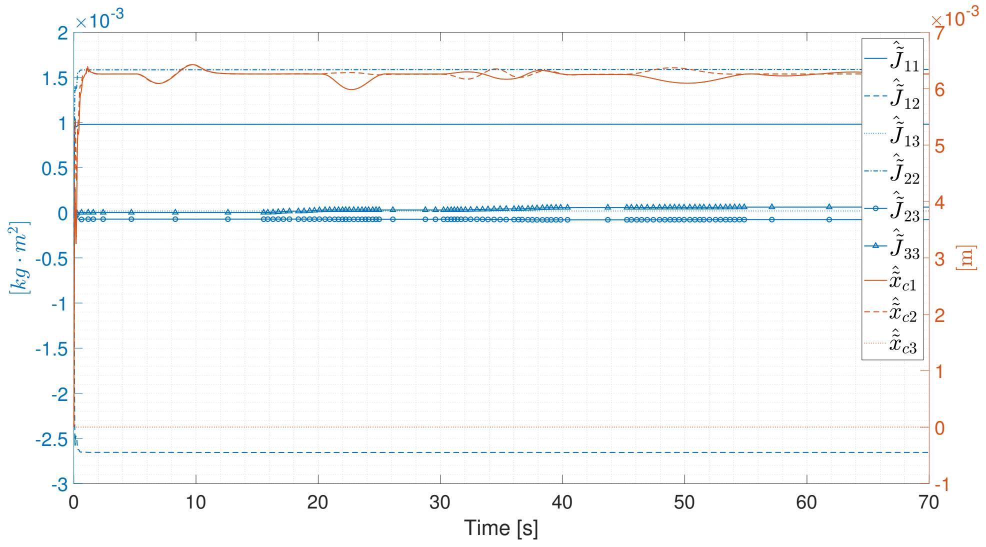

Figure 4.82: On-line estimation of the uncertain parameters contained in θ_a for the model with the CMRAC adaptive controller. Simulation without disturbance.

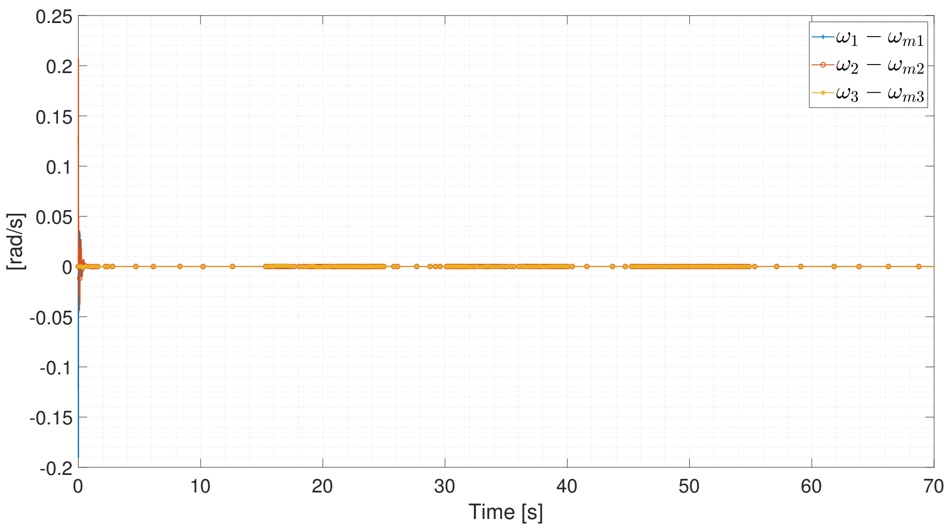

Figure 4.83: Error between the plant and the reference model angular velocity for the model with the CMRAC adaptive controller. Simulation without disturbance.

e_{abs}	**U.M.**	**CPID**	**Adaptive**
$\mathbf{e_{x1}}$	[m]	0.1923	0.1650
$\mathbf{e_{x2}}$	[m]	0.0676	0.0256
$\mathbf{e_{x3}}$	[m]	0.4713	0.4114
$\mathbf{v_{x1}}$	[m/s]	0.0704	0.0493
$\mathbf{v_{x2}}$	[m/s]	0.1332	0.0312
$\mathbf{v_{x3}}$	[m/s]	0.2487	0.2230
$\alpha_{\mathbf{x1}}$	[deg]	6.3290	3.1721
$\alpha_{\mathbf{x2}}$	[deg]	5.1242	4.1324
$\alpha_{\mathbf{x3}}$	[deg]	17.4496	14.1660
$\omega_{\mathbf{x1}}$	[deg/s]	43.6134	36.9546
$\omega_{\mathbf{x2}}$	[deg/s]	38.4120	36.2783
$\omega_{\mathbf{x3}}$	[deg/s]	10.9505	9.0946

Table 4.6: Comparison of the maximum absolute error e_{abs} between the model with the CPID and the CMRAC adaptive controller acting on the attitude. Simulation without disturbance.

e_{rms}	**U.M.**	**CPID**	**Adaptive**
$\mathbf{e_{x1}}$	[m]	0.0476	0.0231
$\mathbf{e_{x2}}$	[m]	0.0086	0.0084
$\mathbf{e_{x3}}$	[m]	0.0847	0.0307
$\mathbf{v_{x1}}$	[m/s]	0.0176	0.0085
$\mathbf{v_{x2}}$	[m/s]	0.0153	0.0056
$\mathbf{v_{x3}}$	[m/s]	0.0495	0.0229
$\alpha_{\mathbf{x1}}$	[deg]	0.9386	0.9179
$\alpha_{\mathbf{x2}}$	[deg]	1.2295	1.2006
$\alpha_{\mathbf{x3}}$	[deg]	5.4514	5.5089
$\omega_{\mathbf{x1}}$	[deg/s]	0.0187	0.0069
$\omega_{\mathbf{x2}}$	[deg/s]	0.0214	0.0086
$\omega_{\mathbf{x3}}$	[deg/s]	0.0203	0.0073

Table 4.7: Comparison of the root mean square error e_{rms} between the model with the CPID and the CMRAC adaptive controller acting on the attitude. Simulation without disturbance.

some improvements. The adaptive controller is able to enhance also the position performances even if only a proportional baseline controller is acting on it. In general the variables do not show important high frequency oscillations meaning that, considering the good tracking performances, the CMRAC architecture is working well. The estimated parameters $\hat{d}_T$ and $\hat{\theta}_a$ converge to some stable values after a short initial transient and the angular velocity error, with respect to the reference model, is very small for all the simulation.

4.8.2 Simulation with a step disturbance

As mentioned before, in order to better understand the UAV disturbance rejection capability, one should evaluate how the tiltrotor performs when a sudden step disturbance is applied to the system. For this purpose, it is possible to look to figures 4.84, 4.85, 4.86, 4.87, 4.88, 4.89, 4.90, 4.91, 4.92, 4.93, 4.94, 4.95, 4.96, 4.97, 4.98, 4.99 and table 4.8, which show the comparison between the model with the CPID and the adaptive CMRAC controller. In this case, for the reasons explained before, only the absolute error will be reported.

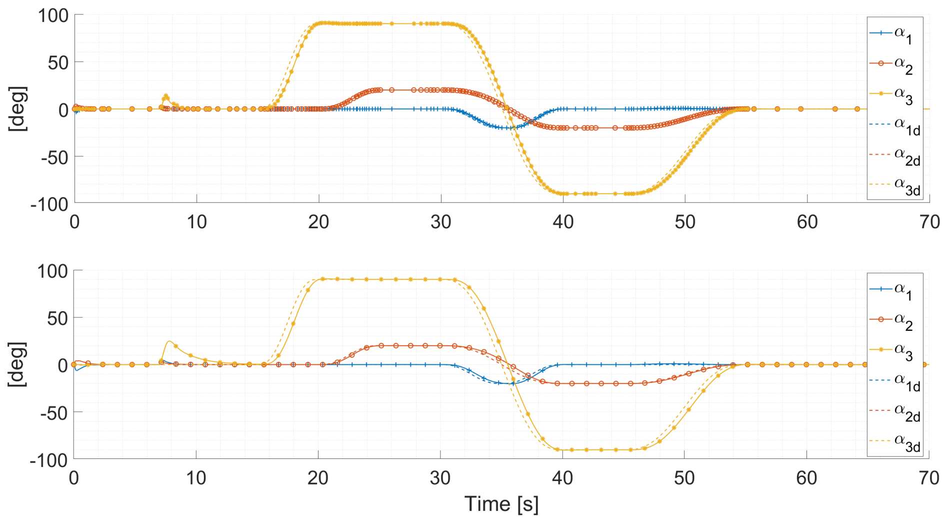

Figure 4.84: Plant and desired Euler angles for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

As it is possible to notice from the results obtained, the adaptive controller shows better performances with respect to the CPID approach. In particular, the maximum absolute error demonstrates lower values for the CMRAC. As before, even if the adaptation is acting on the attitude dynamic only, some improvements in the position are also present. The reaction of the system against the step disturbance is quite good since the transient vanishes fast and there are not so

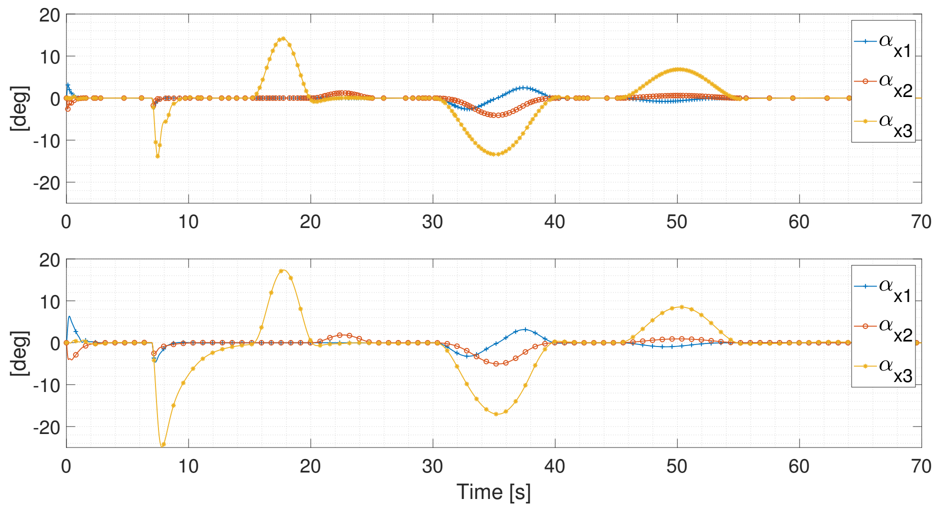

Figure 4.85: Euler angles error α_x for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

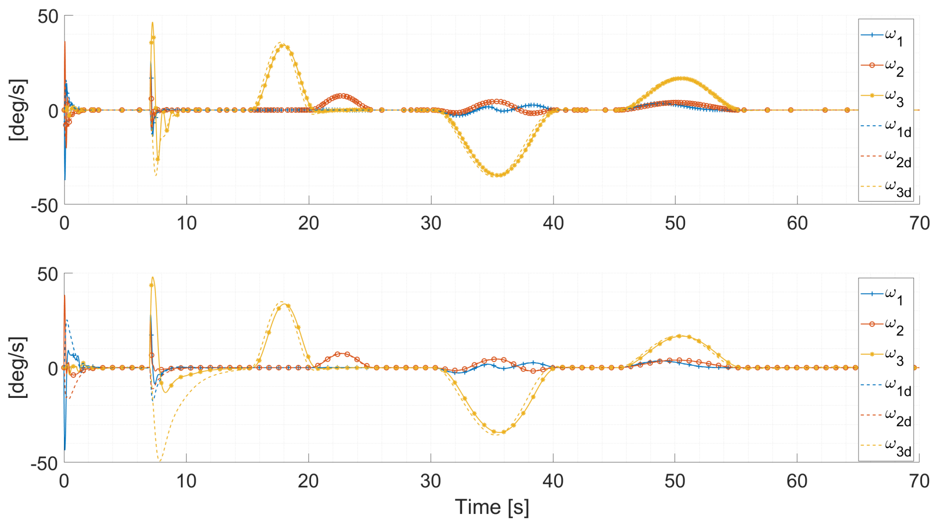

Figure 4.86: Plant and desired angular velocities for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

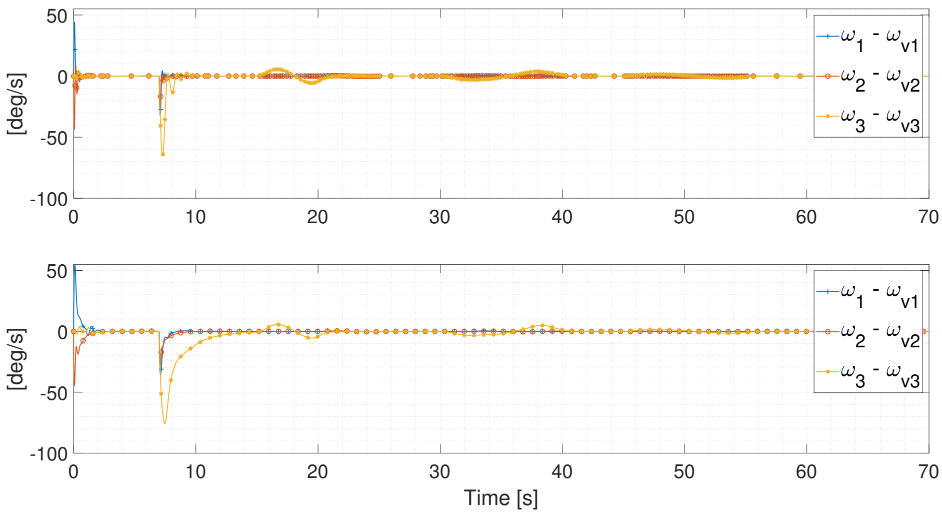

Figure 4.87: Angular velocity error with respect to the outer-loop ω_v for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

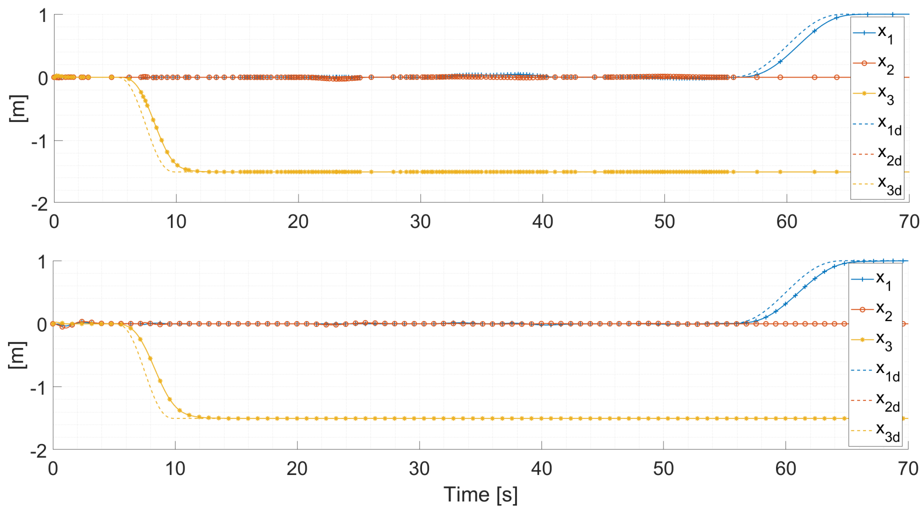

Figure 4.88: Plant and desired position for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

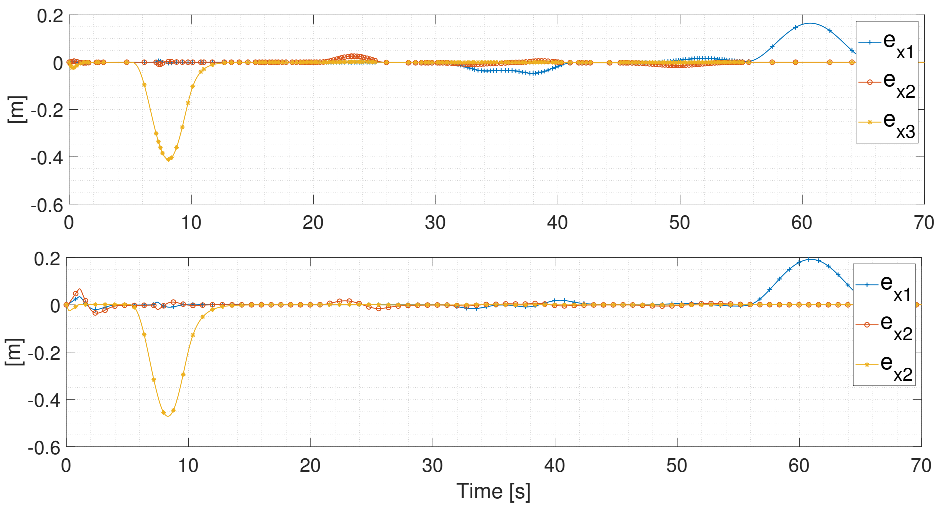

Figure 4.89: Position error e_x for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

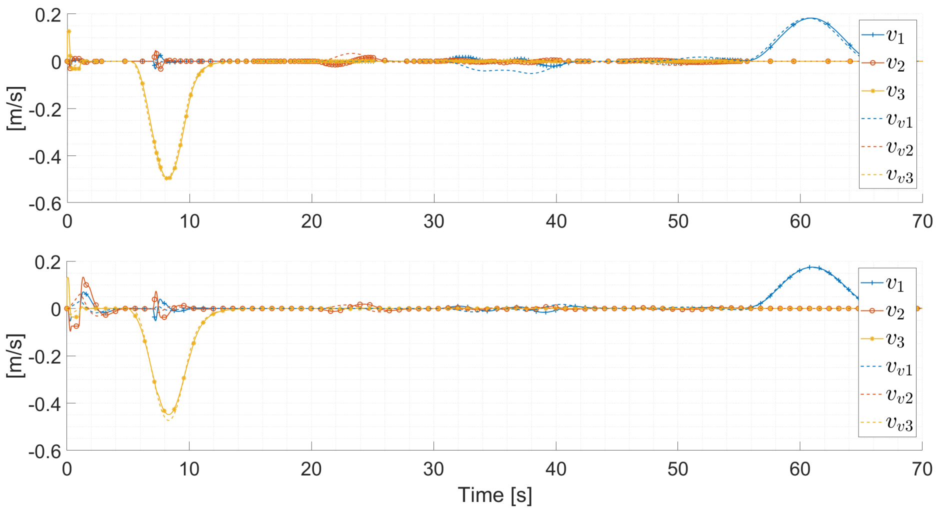

Figure 4.90: Plant and desired translational velocity for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

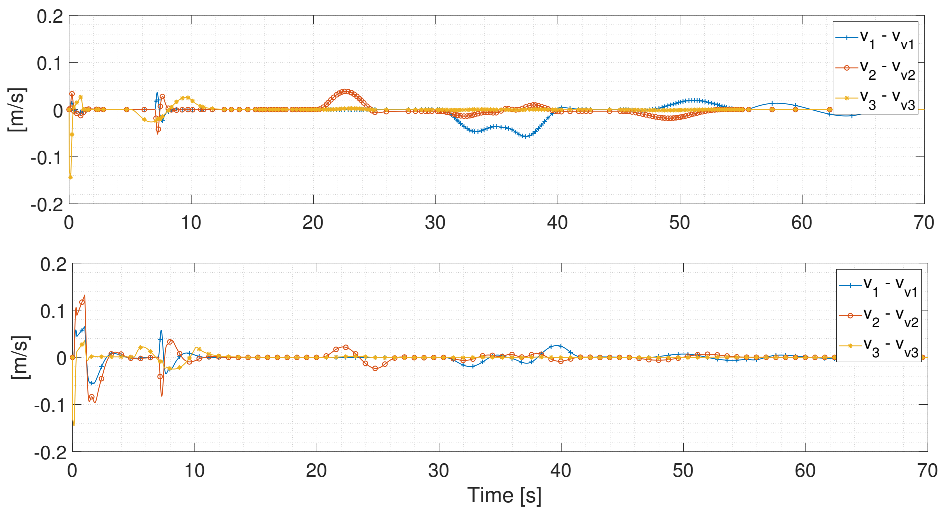

Figure 4.91: Translational velocity error with respect to the outer-loop v_v for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

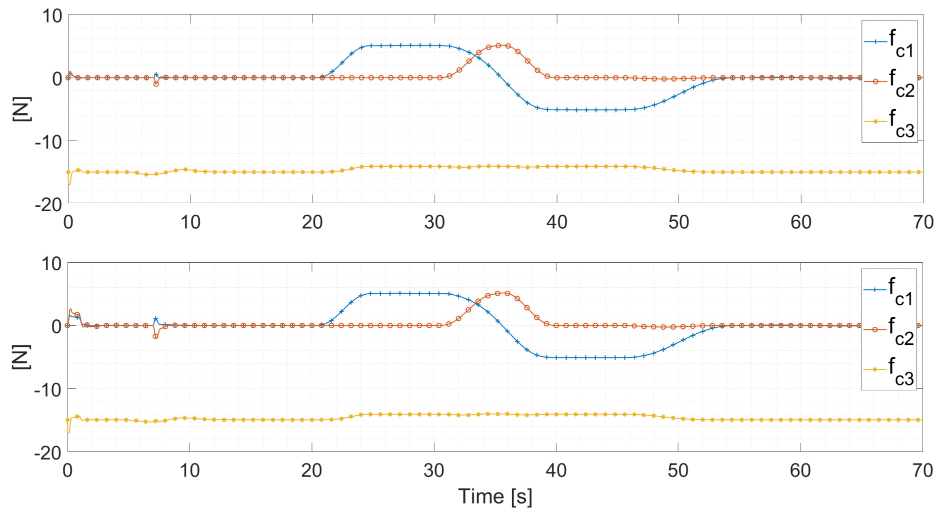

Figure 4.92: Force control input $f_c = f_c^b + f_c^a$ for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

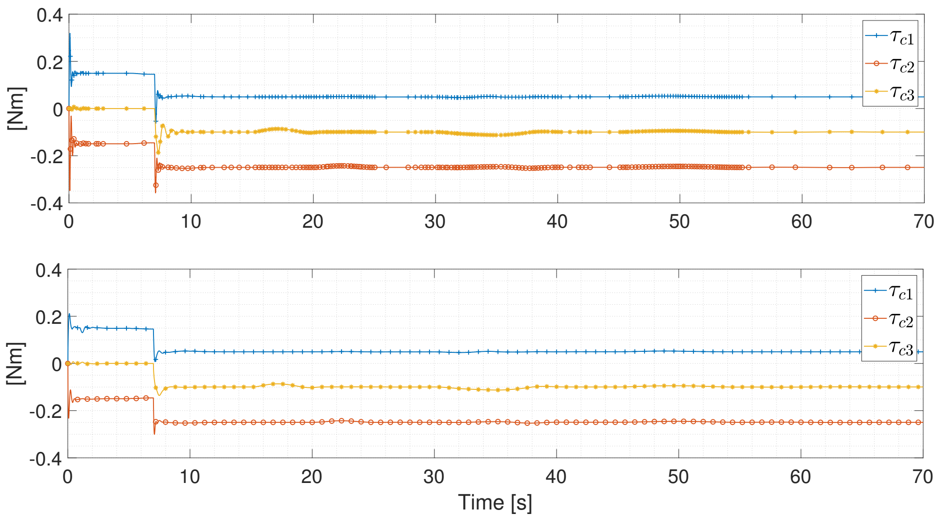

Figure 4.93: Torque control input $\tau_c = \tau_c^b + \tau_c^a$ for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

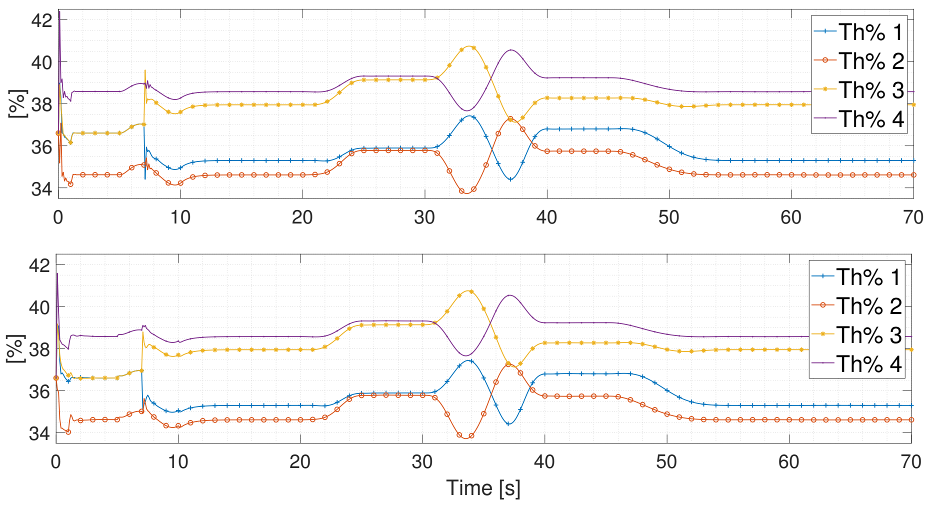

Figure 4.94: $Th\%$ requested by the four UAV's propellers for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

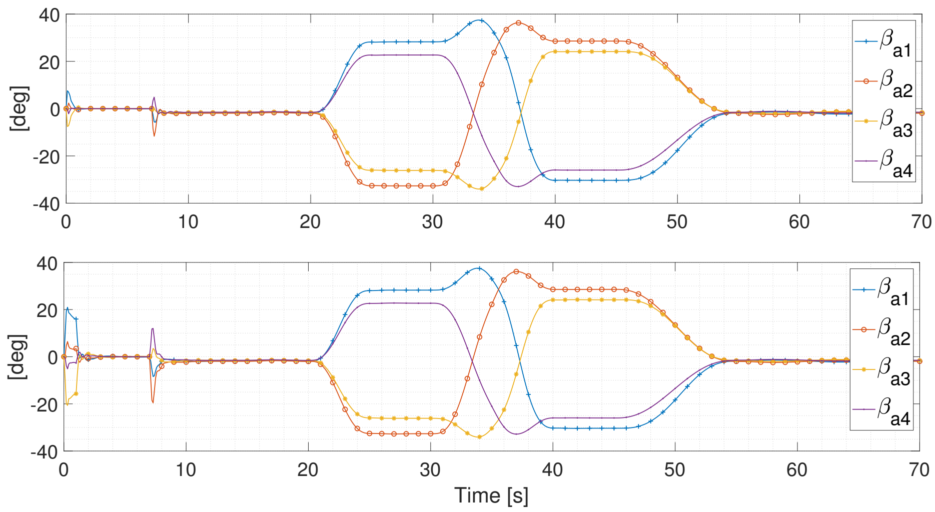

Figure 4.95: Servo-actuators angular position β_a requested by the four UAV's propellers for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom). Simulation with disturbance.

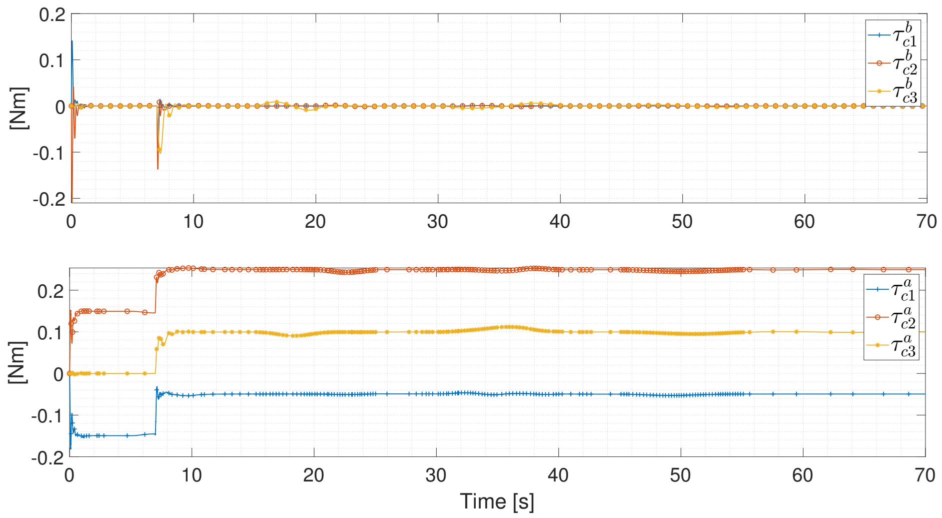

Figure 4.96: Baseline (on top) and adaptive (bottom) control input torque for the model with the CMRAC adaptive controller. Simulation with disturbance.

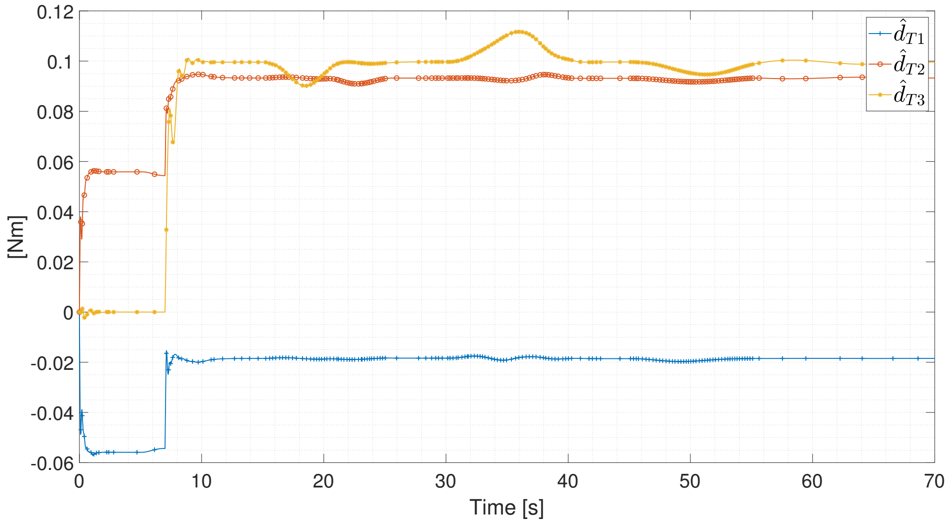

Figure 4.97: On-line estimation of the disturbance $\hat{d}_T$ for the model with the CMRAC adaptive controller. Simulation with disturbance.

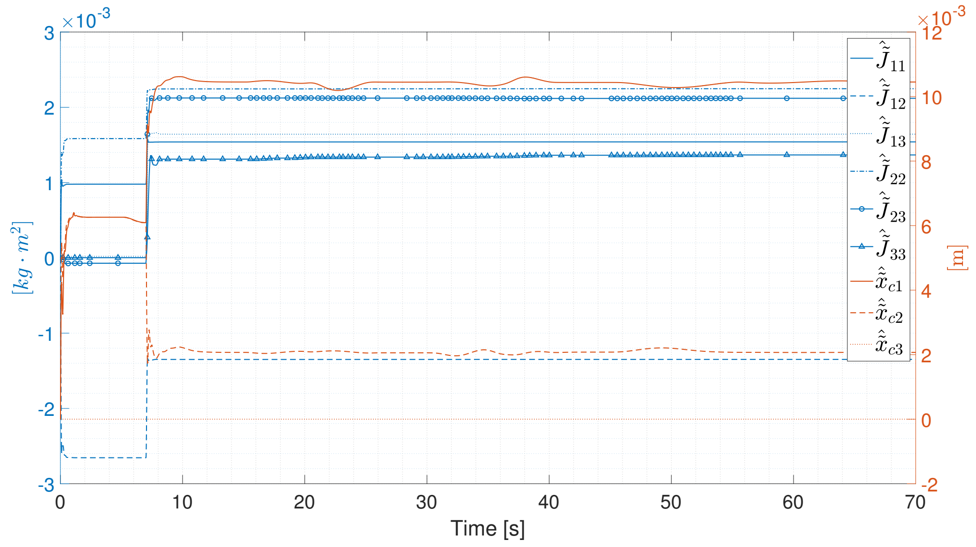

Figure 4.98: On-line estimation of the uncertain parameters contained in θ_a for the model with the CMRAC adaptive controller. Simulation with disturbance.

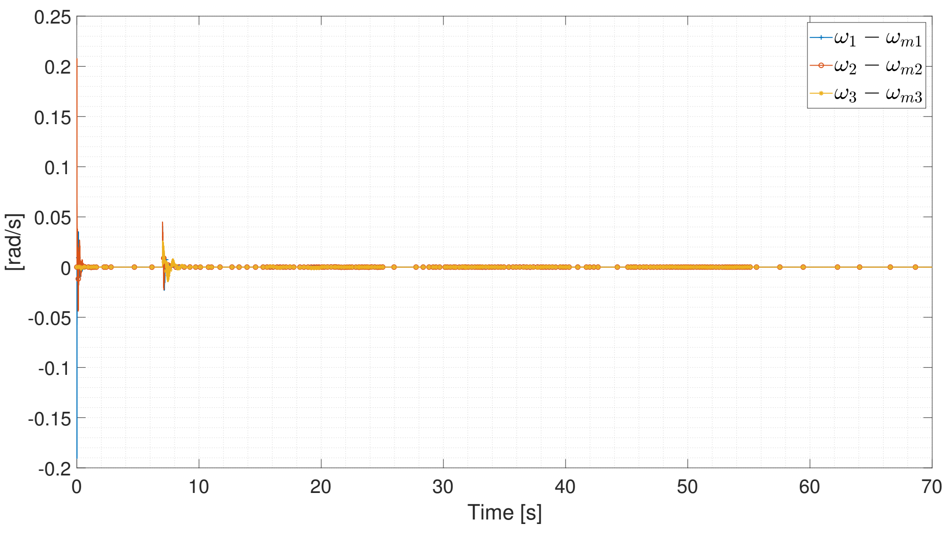

Figure 4.99: Error between the plant and the reference model angular velocity for the model with the CMRAC adaptive controller. Simulation with disturbance.

e_{abs}	**U.M.**	**CPID**	**Adaptive**
$\mathbf{e_{x1}}$	[m]	0.1923	0.1650
$\mathbf{e_{x2}}$	[m]	0.0676	0.0256
$\mathbf{e_{x3}}$	[m]	0.4714	0.4115
$\mathbf{v_{x1}}$	[m/s]	0.0704	0.0493
$\mathbf{v_{x2}}$	[m/s]	0.1332	0.0462
$\mathbf{v_{x3}}$	[m/s]	0.2487	0.2230
$\alpha_{\mathbf{x1}}$	[deg]	6.3290	3.1721
$\alpha_{\mathbf{x2}}$	[deg]	5.0428	4.1072
$\alpha_{\mathbf{x3}}$	[deg]	24.8948	14.1721
$\omega_{\mathbf{x1}}$	[deg/s]	43.6134	36.9546
$\omega_{\mathbf{x2}}$	[deg/s]	38.4120	36.2783
$\omega_{\mathbf{x3}}$	[deg/s]	47.9761	46.3772

Table 4.8: Comparison of the maximum absolute error e_{abs} between the model with the CPID and the CMRAC adaptive controller acting on the attitude. Simulation with disturbance.

many high frequency oscillations. The estimation of the uncertain parameters shows a slight different behaviour with respect to the previous case. In fact, at the beginning they converge to a certain value and then, after the step occurrence, they converge to a different value. The dynamic of the tiltrotor has shown to be very close to the one generated by the reference model. The biggest error is present only at the very beginning of the simulation, when the controller has to adapt itself to the system, and when the disturbance has been activated.

4.9 Conclusion

The analysis developed until now refer to the attitude dynamic only, since as first approach, it has been chosen to see how the adaptive laws, defined by the Projection Operator, work on the main uncertainties of the system. From the first results computed by the simplified simulator, it is possible to conclude that a very simple adaptive architecture like the MRAC one ensures the convergence of the tracking error even if a linear uncertain model and a very simple proportional baseline controller have been considered. As expected, many control variables show very fast oscillations and they increase incrementing the adaptation gain. Since, in general we want to guarantee fast response and good tracking, this drawback cannot be completely avoided. A possible solution has been discovered in the CMRAC controller, which is able to maintain good performances without the presence of vibrations thanks to the new innovation term. Therefore, this last approach has been chosen as the most suitable architecture for our problem and, for this reason, it has been proven on non-linear plant models, on the tilt-arm simulator and finally compared with the CPID controller. The results have shown that the adaptive CMRAC controller ensures better performances with respect to the CPID one and good tracking and disturbance rejection capabilities. Moreover, since the CMRAC has the ability to adapt itself against the uncertainties of the system, it has been discovered that the tuning effort of the adaptive controller parameters has been less intense with respect to the classical PID. As drawback, the computational power is increased together with the overall complexity of the algorithm, meaning that a better hardware is needed.

Chapter 5

Position Adaptive Control System

This chapter will focus on adaptive controllers applied to the position dynamic. The structure is very similar to the one presented before. Firstly, the details about the baseline controller chosen and the derived uncertain state space model for the position dynamic will be shown. Secondly, the reference model selected will be illustrated together with the adaptive control law adopted highlighting the reasons of its choice. Finally, the results obtained with the tilt-arm simulator will be presented, in order to understand if an adaptive controller acting on the position dynamic actually improves the performances of the system.

5.1 Baseline controller

As previously done in the last chapter and as discussed in section 1.5, it has been chosen to design the baseline controller as the CPID architecture but without the derivative and integral terms, since in theory, the adaptive controller should be able to replace the features of these components. Therefore, considering only the last two equations of (1.12), it is possible to write the position baseline controller as:

$$\begin{cases} v_v = -K^P_{po} e_x \\ f^b_c = R^T (K^P_{ff} v_v - K^P_{pi}(v - v_V) + mge_3) \end{cases} \tag{5.1}$$

in which only the proportional term has been taken from the complete CPID controller.

5.2 Uncertain state space model

As previously done for the attitude dynamic, also for the position one it is possible to derive a complete uncertain state space model similar to the one presented in equation (3.5). Referring to the position dynamic of (3.15) only, one should write:

$$
\begin{aligned}
\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} &= \underbrace{\begin{bmatrix} 0_3 & 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & -\frac{K_{pi}^P}{m} I_3 & 0_3 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x \\ v \end{bmatrix}}_{X} + \\
&+ \underbrace{\begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & \frac{K_{ff}^P + K_{pi}^P}{m} I_3 & 0_3 \end{bmatrix}}_{B_r} \underbrace{\begin{bmatrix} 0_{31} \\ v_v \end{bmatrix}}_{r} + \\
&+ \underbrace{\begin{bmatrix} 0_3 \\ \frac{1}{m} I_3 \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} 0_3 \\ I_3 \end{bmatrix}^T}_{\Lambda} \underbrace{\left(\begin{bmatrix} 0_{31} \\ f(X)\theta_{v4,5} \end{bmatrix} + \begin{bmatrix} 0_{31} \\ f_c^a \end{bmatrix} + \begin{bmatrix} 0_{31} \\ d_{off} \end{bmatrix} \right)}_{Uncertain}
\end{aligned}
\tag{5.2}
$$

where, as mentioned in section 3.2.1 and as done in the previous chapter, the position baseline controller (5.1) appears inside the A and the B_r matrix, the r vector collects the desired translational velocity generated by the controller v_v, and the last row collects the uncertain portion of the model. In this case, B is known since the mass is known, and Λ is the identity. Instead, the uncertain contribution is split in a first term that depends on the state, in a second term that represents the adaptive force control input and finally in a third one that collects all the constant disturbances that are independent from the state.

5.3 Reference model

In contrast to the previous chapter, the position dynamic will be directly analysed with the complete tilt-arm simulator, implementing the model with the CMRAC controller. In fact, the previous simulations have shown that the MRAC approach is not a suitable architecture for this problem. With that in mind, the reference model has been derived by taking the ideal version of the third equation of system (3.3). Therefore, we can assume:

- Negligible disturbance: $d_F = 0$
- $x_c \approx [0, 0, 0]$.

So, the CMRAC differential reference model equation describing the behaviour of the quadrotor translational velocities $v_m \in \mathbb{R}^3$ becomes simply:

$$
m\dot{v}_m = -mge_3 + Rf_c^b + \lambda_p e_p \tag{5.3}
$$

Where $\lambda_p \in \mathbb{R}^{3\times 3}$ is the position closed-loop gain and $e_p = v - v_m$ is the adaptation error. As in the attitude case, the above expression takes as input the baseline control force showed in equation (5.1), but in which v_m has been used in place of v.

Following the same idea of the previous chapter, the results obtained with the model with the CPID controller will be compared to the ones with the adaptive architecture. For the same reasons explained before, also in this case it has been decided to tune the reference model gains in such a way that its behaviour gets as close as possible to the CPID one. This will be accomplished by looking directly to the plots and to the tables, where the maximum absolute error 4.5 and root mean square error 4.6 will be reported for all the state variables that refer to the position dynamic. Also here, the trajectory chosen is the Paoll trajectory illustrated in 4.1 and the values of the tuned gains can be found in appendix A table A.2.

Figures 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, and tables 5.1, 5.2 show the results obtained. Since in this chapter we analyse the position dynamic only, they refer to the position, the translational velocities, the control input forces and the thrust percentage $Th\%$.

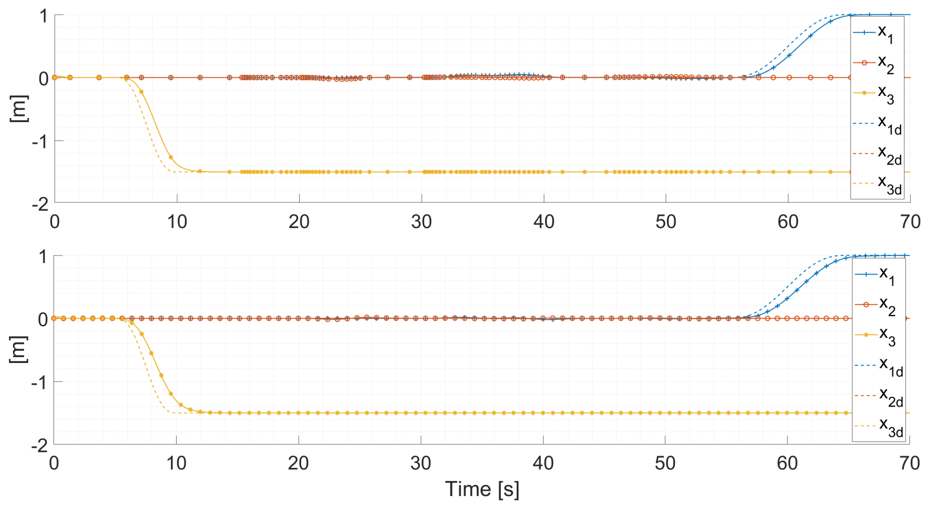

Figure 5.1: Plant and desired position for the reference model (on top) and for the system with the CPID controller (bottom).

5.4 Position adaptive laws

As previously said, the idea followed until now is to decouple the design of the attitude dynamic from the position one. Pursuing the same goal, in this chapter we will assume that the attitude dynamic will not be affected by any sort of uncertainties and, consequently, we will consider it as ideal. This means that for

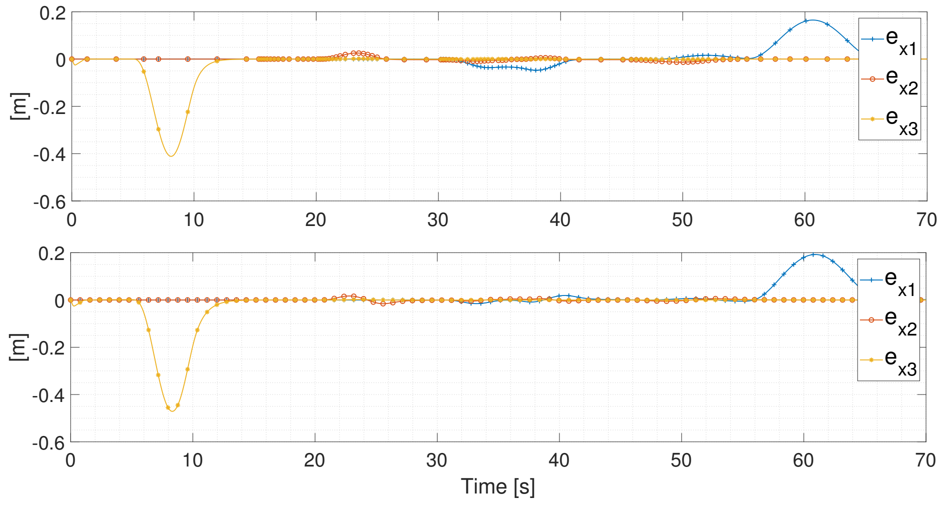

Figure 5.2: Position error e_x for the reference model (on top) and for the system with the CPID controller (bottom).

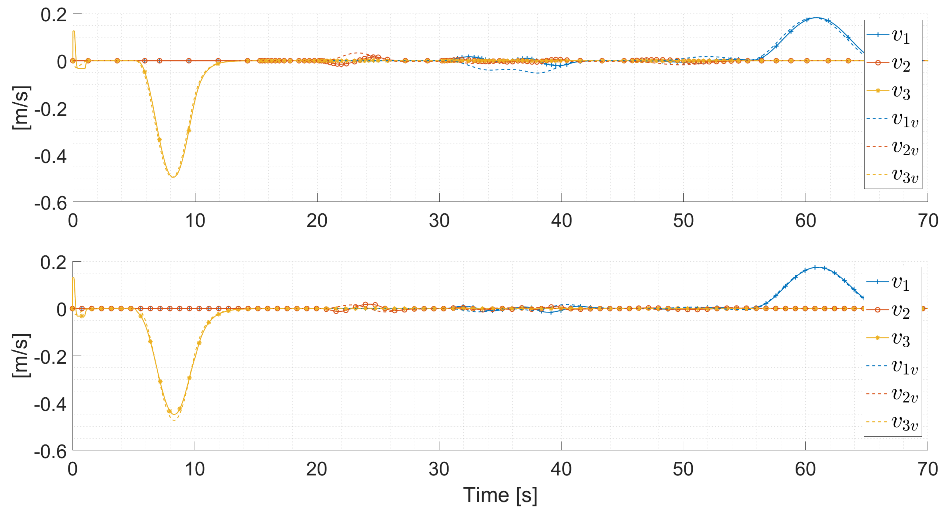

Figure 5.3: Plant and desired translational velocity for the reference model (on top) and for the system with the CPID controller (bottom).

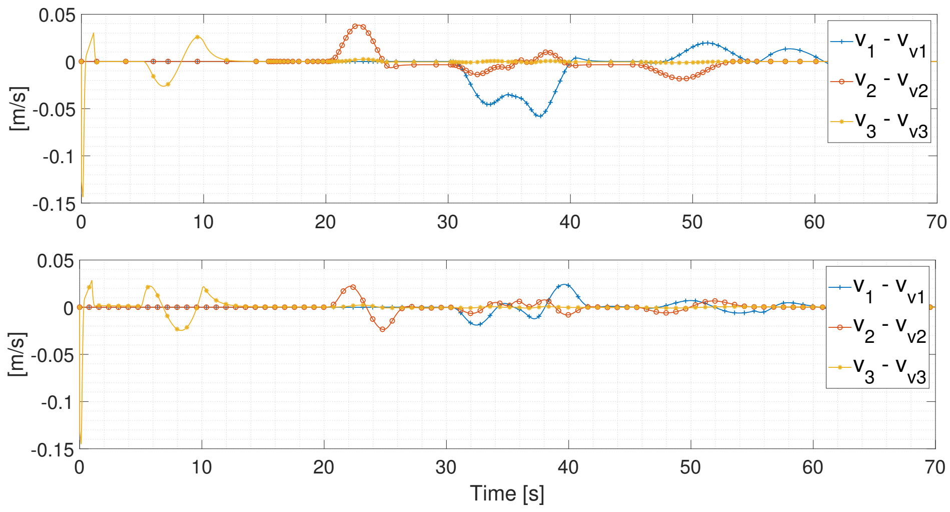

Figure 5.4: Translational velocity error with respect to the outer-loop v_v for the reference model (on top) and for the system with the CPID controller (bottom).

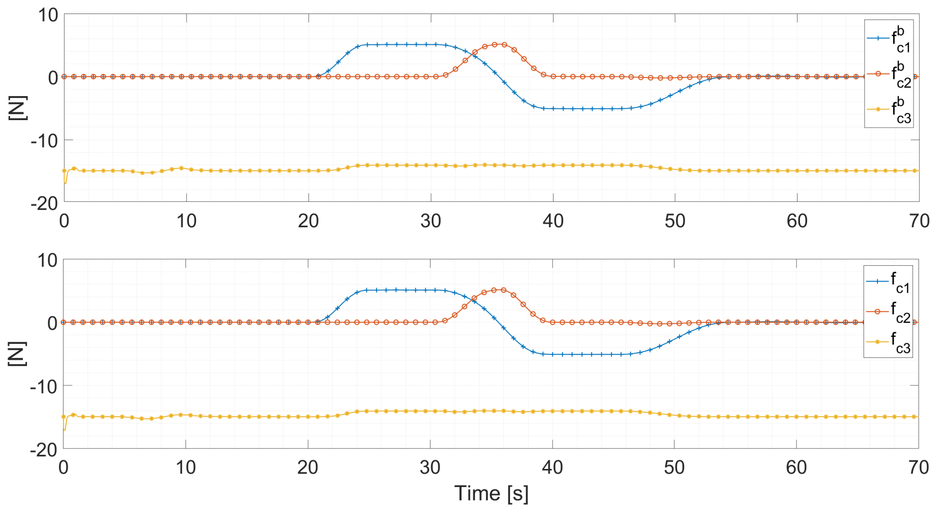

Figure 5.5: Force control input $f_c = f_c^b + f_c^a$ for the reference model (on top) and for the system with the CPID controller (bottom).

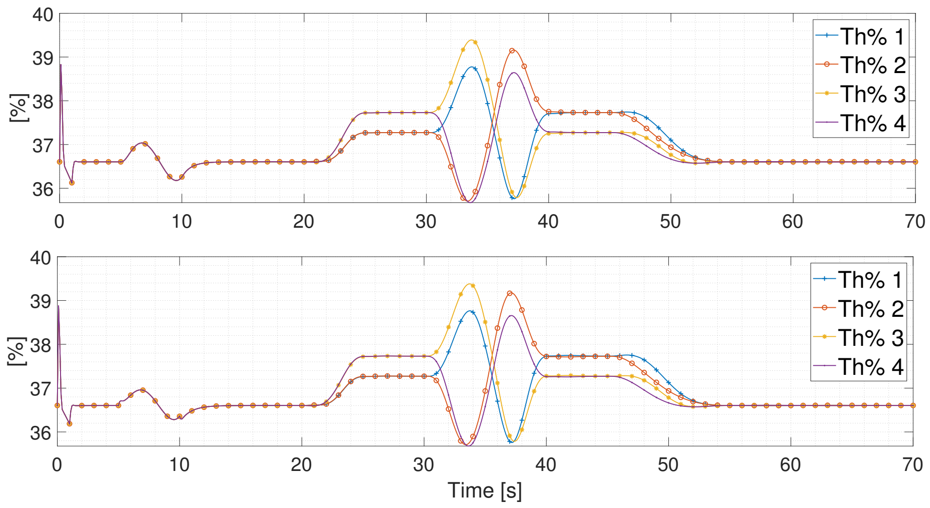

Figure 5.6: $Th\%$ requested by the four UAV's propellers for the reference model (on top) and for the system with the CPID controller (bottom).

e_{abs}	**U.M.**	**CPID**	**Ref. Model**
$\mathbf{e_{x1}}$	[m]	0.1923	0.1650
$\mathbf{e_{x2}}$	[m]	0.0676	0.0256
$\mathbf{e_{x3}}$	[m]	0.4713	0.4114
$\mathbf{v_{x1}}$	[m/s]	0.0704	0.0493
$\mathbf{v_{x2}}$	[m/s]	0.1332	0.0312
$\mathbf{v_{x3}}$	[m/s]	0.2487	0.2230

Table 5.1: Comparison of the maximum absolute error e_{abs} between the system with the CPID and the reference model for the position dynamic.

e_{rms}	**U.M.**	**CPID**	**Ref. Model**
$\mathbf{e_{x1}}$	[m]	0.0476	0.0231
$\mathbf{e_{x2}}$	[m]	0.0086	0.0084
$\mathbf{e_{x3}}$	[m]	0.0847	0.0307
$\mathbf{v_{x1}}$	[m/s]	0.0176	0.0085
$\mathbf{v_{x2}}$	[m/s]	0.0153	0.0056
$\mathbf{v_{x3}}$	[m/s]	0.0495	0.0229

Table 5.2: Comparison of the root mean square error e_{rms} between the system with the CPID and the reference model for the position dynamic.

all the results that will be illustrated and referring to the attitude dynamic only, it has been assumed:

- No disturbances: $d_T = 0$
- Null center of mass position in the fourth equation of (1.11)
- Exact inertia tensor: $J = \bar{J}$ with $\bar{J}$ equals to equation (4.10).

On the other hand, referring to the third equation of (1.11), since the position dynamic is the one under study in this chapter, all its uncertainties have been taken into account. In particular, given that the main uncertain contribution comes only from the disturbance vector d_F, unlike the previous case, it has been modelled in a more complicated way, as showed in section 3.2.2.

As mentioned in section 2.2, in general the adaptive control input has to be chosen in such a way that it eliminates all the uncertain contributions. Regarding our problem they are earlier modelled as in equation (3.14), therefore f_c^a can be written as:

$$f_c^a = -\hat{d}_{off} + f(X)\hat{\theta}_{p4,5} \tag{5.4}$$

where, as always, the hat symbol indicates that the variable under interest has to be estimated.

As in the previous chapter, the estimation has been carried out by an adaptive law that has been chosen as the Projection Operator: it allows to bound all the processes, to avoid *wind up* problems and to prevent the parameters from drifting away. Therefore, the adaptive law takes the form of equation (5.5).

$$\dot{\hat{d}}_{off} = \gamma_{off} Proj\left(\hat{d}_{off}, B_p^T P_p e_p\right) \qquad \dot{\hat{\theta}}_{p4,5} = \gamma_{\theta_p} Proj\left(\hat{\theta}_{p4,5}, f(X)^T B_p^T P_p e_p\right) \tag{5.5}$$

where $\gamma_{off} \in \mathbb{R}^3$ and $\gamma_{\theta_p} \in \mathbb{R}^2$ are the adaptation gains acting respectively on the offset and state dependent term of the d_T estimation. These latter can be collected in a single vector as:

$$\gamma_P = \begin{bmatrix}\gamma_{off}, & \gamma_{\theta_p}\end{bmatrix}^T \quad \in \mathbb{R}^5. \tag{5.6}$$

The initial conditions used to integrate (5.5) are reported in A.3. Furthermore, $e_p = v - v_m \in \mathbb{R}^3$ is the translational velocity error between the plant and the reference model, P_p is the solution of the Lyapunov equation:

$$A_p^T P_p + P_p A_p = -Q$$

A_p, Q and B_p are chosen for simplicity by looking at the second component of the state space model showed in equation (5.2), since in this chapter we are looking at the position dynamic only. Therefore:

$$\begin{cases} B_p = \frac{1}{m} I_3 \\ A_p = -\frac{K_{pi}^P}{m} I_3 \\ Q = I_3 \end{cases}$$

In this case, the position reference model used to compute the above cited errors has been implemented for the CMRAC architecture only as the one showed in equation (5.3).

5.5 Tilt-arm simulator

In the previous chapter, we have seen that, before using the complete tilt-arm simulator, a simplified numerical software has been realized. This was done in order to understand how the adaptive controller behaves on this particular system and to highlight its possible drawbacks. Since from the previous simulations we have already understood the main peculiarities of the control architectures under study, and given that the position dynamic is less affected by uncertainties and consequently it is simpler from the adaptation point of view, it has been choose to implement the position adaptive controller directly on the tilt-arm simulator. As before, the performances will be evaluated against the CPID ones.

The adaptation and the closed-loop gains have been chosen with the same idea of the previous case. We want the performances to be better or at least equal to the ones given by the CPID approach, but taking into account of CPU effort and the possible presence of fast oscillations. For these reasons, the adaptation gain, which affects the speed of adaptation and therefore of the tracking, has been selected as:

$$\gamma_P = [400, \quad 400, \quad 150, \quad 100, \quad 100]^T \tag{5.7}$$

while the closed loop gain, which instead affects the number of oscillations present and how much the reference system is influenced by the innovation error e_p, has been taken as:

$$\lambda_p = \begin{bmatrix} 30 & 0 & 0 \\ 0 & 30 & 0 \\ 0 & 0 & 8 \end{bmatrix}. \tag{5.8}$$

Since we are evaluating the position performances, the trajectory chosen is an "8 shape" trajectory as the one showed in figure 5.7. Instead, the values of the center of mass position and inertia tensor have been left equal to the ones reported in chapter 4. Regarding the d_F coefficients that appear in equation (3.11), it has to be reminded that d_F is composed by an offset plus a term that represents the aerodynamic resistance. In this context, the offset term has been exploited to simulate a sudden gust of wind that the UAV can encounter during an outdoor flight. Therefore, d_{off} in equation 3.14 has been selected as:

$$d_{off} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{cases} 0_{31} & \text{if} \quad 0 \leqslant t < 10 \\ [0.3, \ 0.3, \ 0.1]^T & \text{if} \quad 10 \leqslant t < 50 \\ 0_{31} & \text{elsewhere.} \end{cases} \tag{5.9}$$

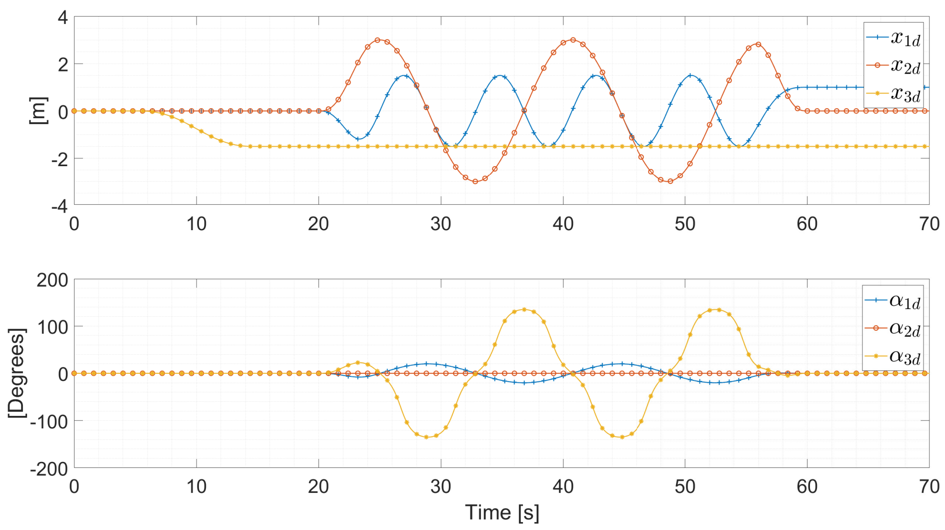

Figure 5.7: 8 Shape trajectory: desired position (on top) and Euler angles (bottom)

Instead, $\theta_{p4,5}$ that refers to the aerodynamic resistance has been chosen as follow:

$$\theta_{p4,5} = [0.1, \quad 0.01]^T. \tag{5.10}$$

As a reminder, for a complete overview of the simulator data one should refer to appendix A. Figures 5.8, 5.9, 5.10, 5.11, 5.12, 5.13, 5.14, 5.15, 5.16, 5.17, 5.18, 5.19, 5.20, 5.21, 5.22 and 5.23 show the results achieved. As before, tables 5.3 and 5.4 report respectively the absolute and the root mean square error between the plant state and the desired dynamic computed by the trajectory generator. Since the disturbance is a quadratic function of the translational velocity, the tracking performances and the disturbance rejection capabilities can be evaluated through the same simulation.

As it is possible to see, given that the attitude is not affected by uncertainties, variables such as the Euler angles and the angular velocities present a behaviour very similar to the ideal one. The simple proportional controller is enough to obtain good performances. Regarding the position dynamic, one should note that the position error with respect to the desired one is lower for the CMRAC approach, especially in the interval of time in which the disturbance magnitude increases. Moreover, the position and the translational velocity show slight faster transients with respect to the CPID, even if they are still inside the correct ranges for small scale UAV. The same can be said for the force and torque control inputs. In particular, f_c is made by the sum of the baseline plus the adaptive control force. The main contribution comes from the first one, which tries to counteract the gravitational force and to make the tracking, while the second one, with minor adjustments, compensates the uncertainties of the UAV dynamic. This

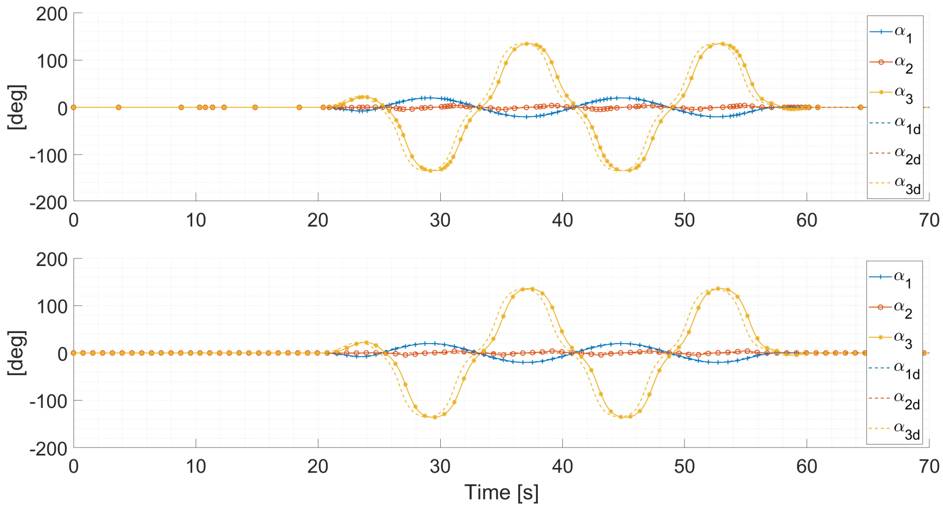

Figure 5.8: Plant and desired Euler angles for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

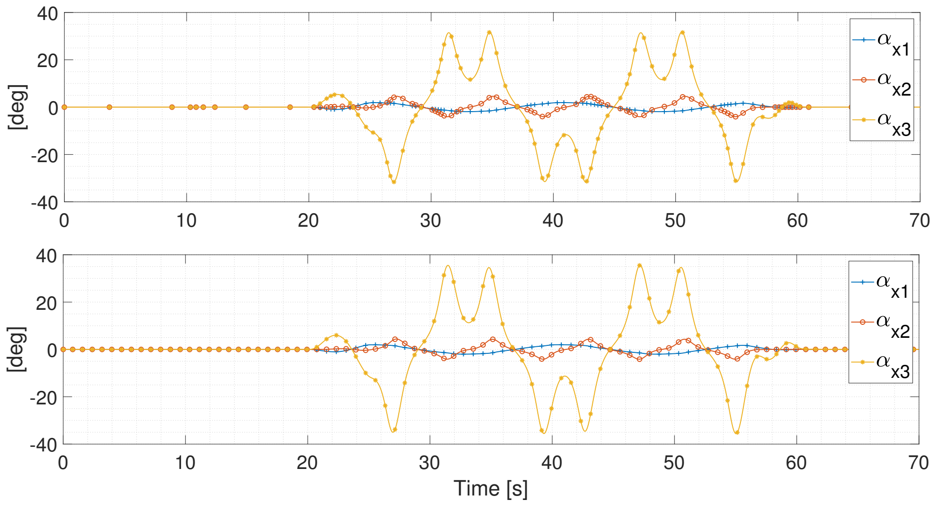

Figure 5.9: Euler angles error α_x for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

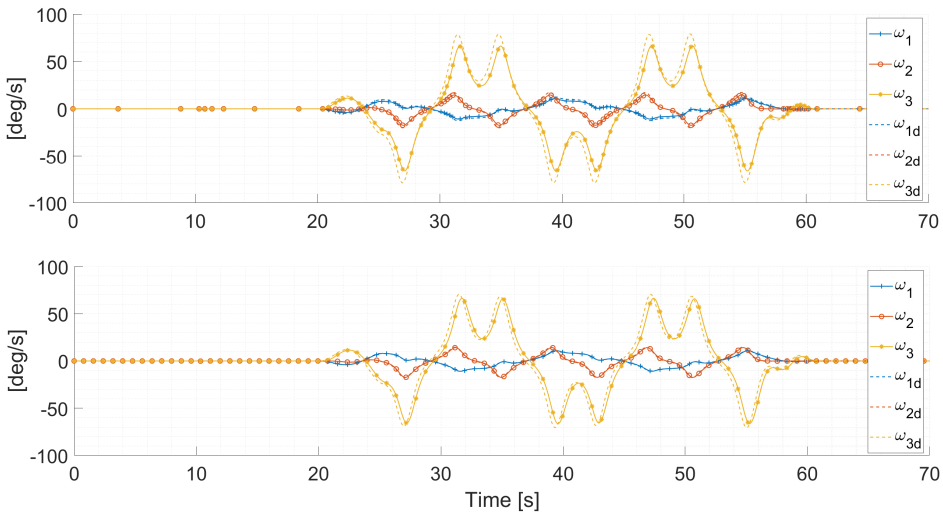

Figure 5.10: Plant and desired angular velocities for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

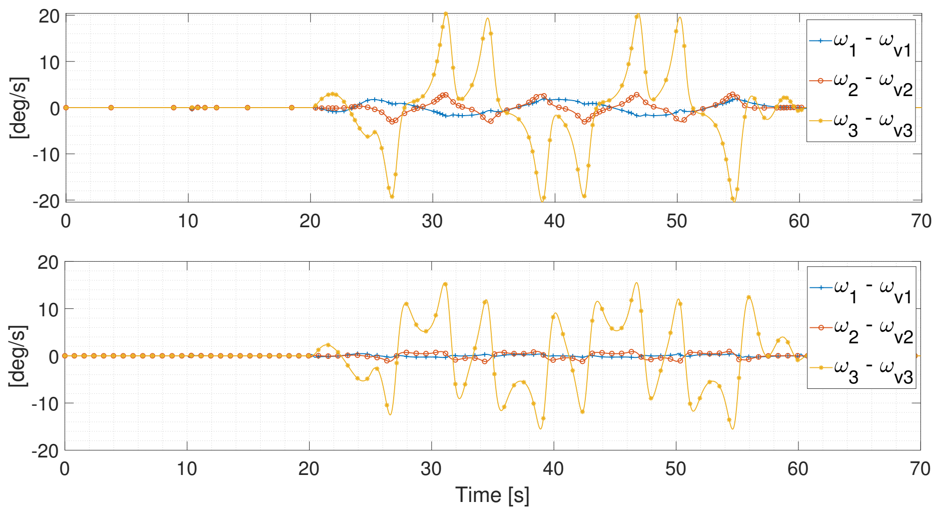

Figure 5.11: Angular velocity error with respect to the outer-loop ω_v for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

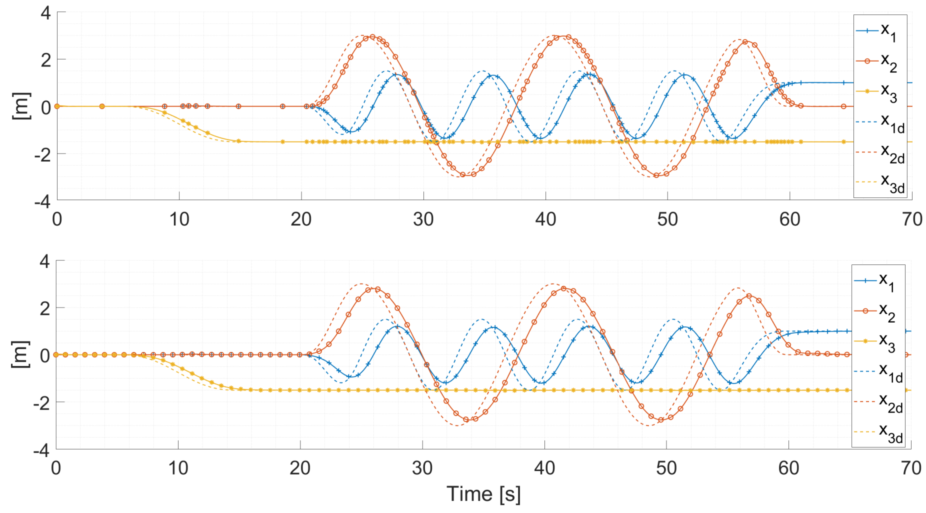

Figure 5.12: Plant and desired position for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

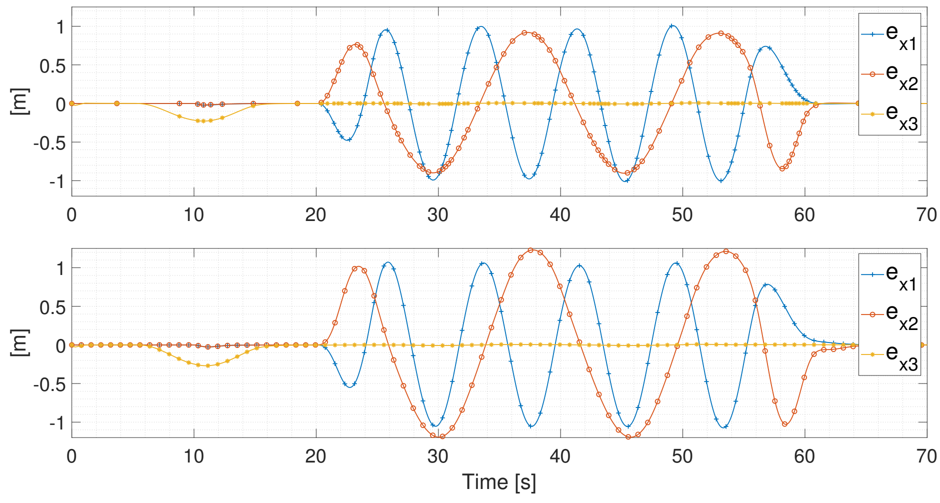

Figure 5.13: Position error e_x for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

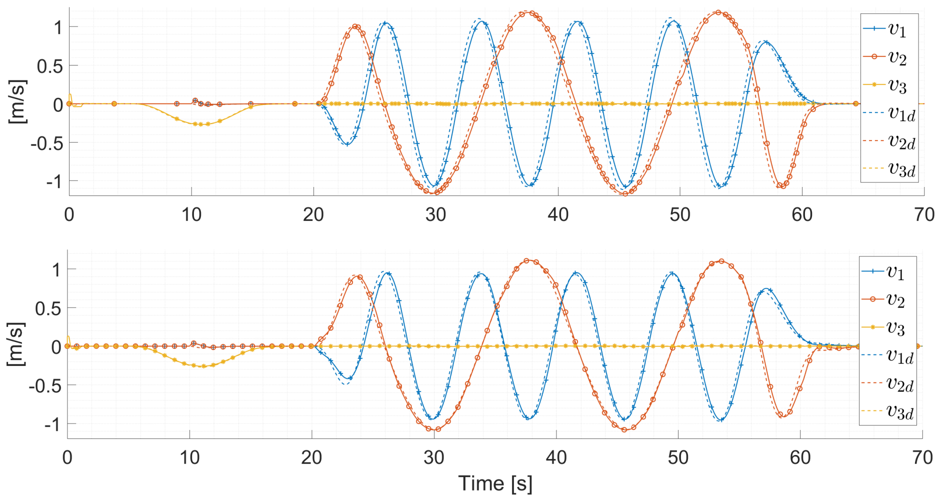

Figure 5.14: Plant and desired translational velocity for the model with the CM-RAC adaptive controller (on top) and with the CPID controller (bottom).

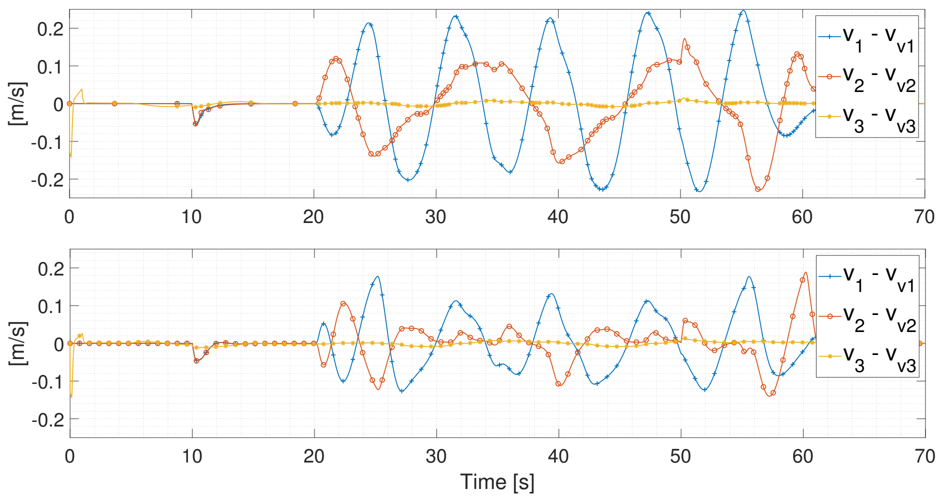

Figure 5.15: Translational velocity error with respect to the outer-loop v_v for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

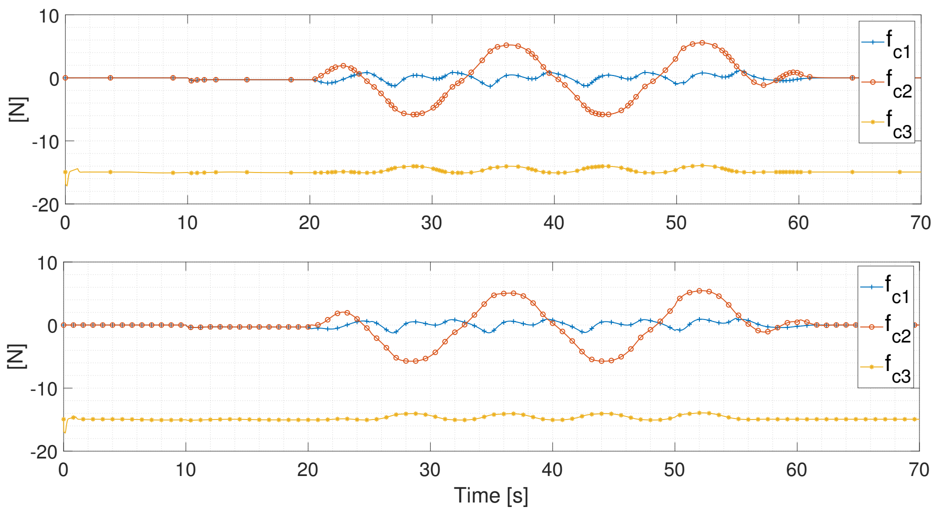

Figure 5.16: Force control input $f_c = f_c^b + f_c^a$ for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

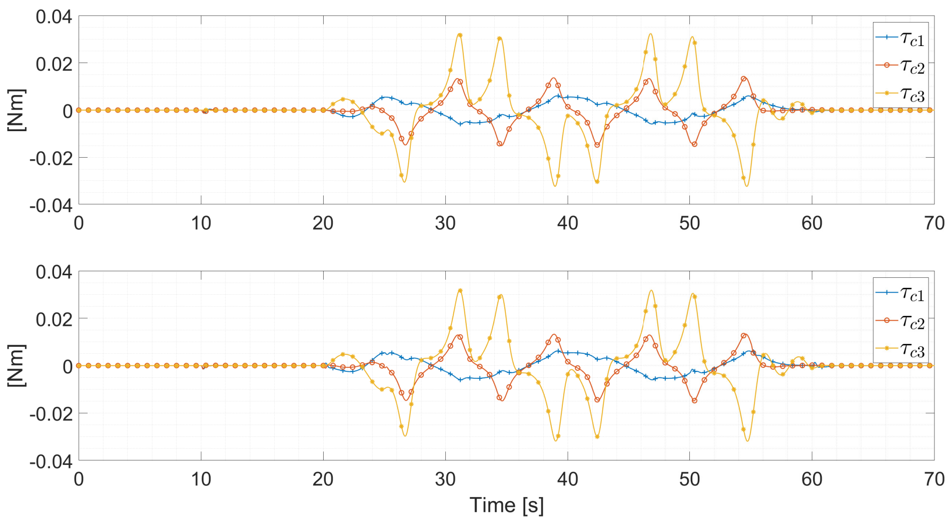

Figure 5.17: Torque control input $\tau_c = \tau_c^b + \tau_c^a$ for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

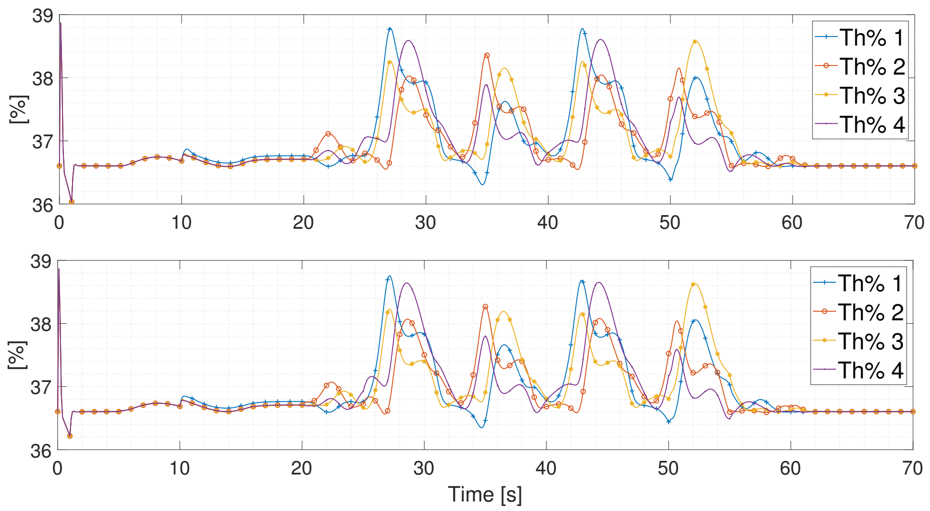

Figure 5.18: $Th\%$ requested by the four UAV's propellers for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

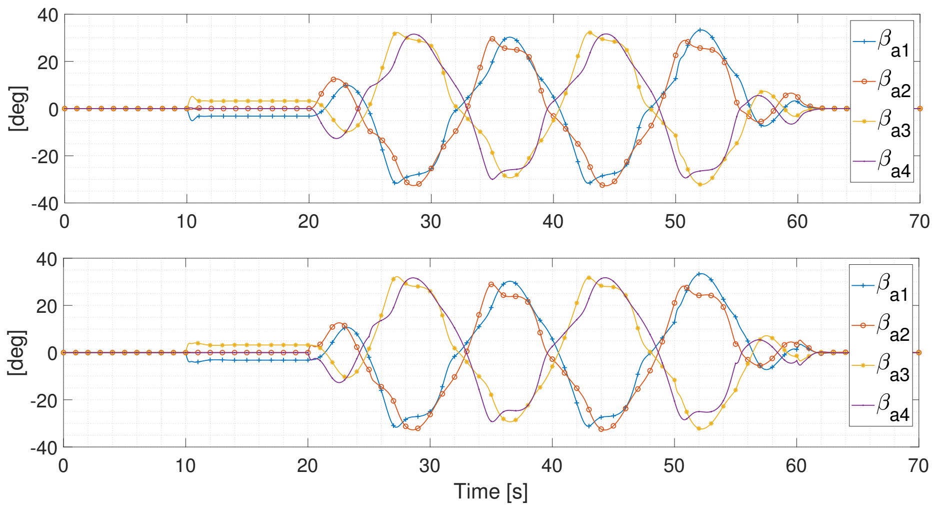

Figure 5.19: Servo-actuators angular position β_a requested by the four UAV's propellers for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

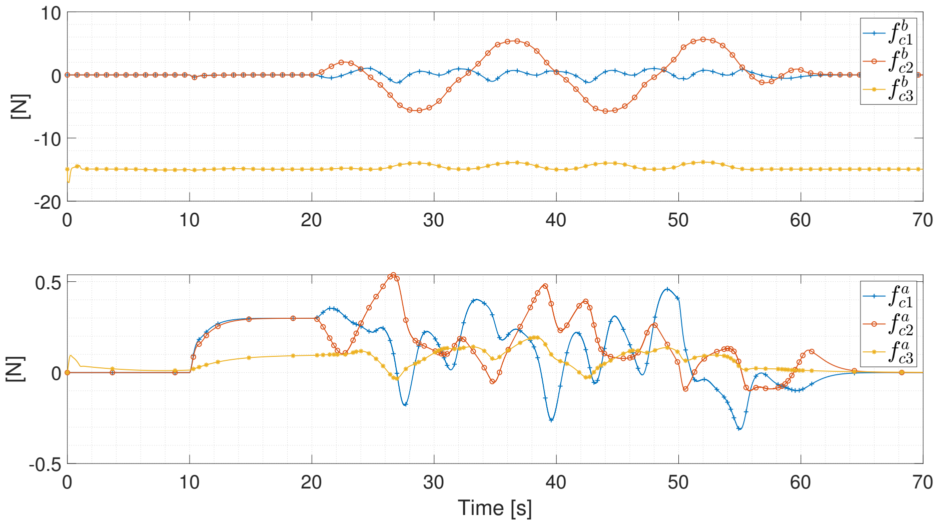

Figure 5.20: Baseline (on top) and adaptive (bottom) control input force for the model with the CMRAC adaptive controller.

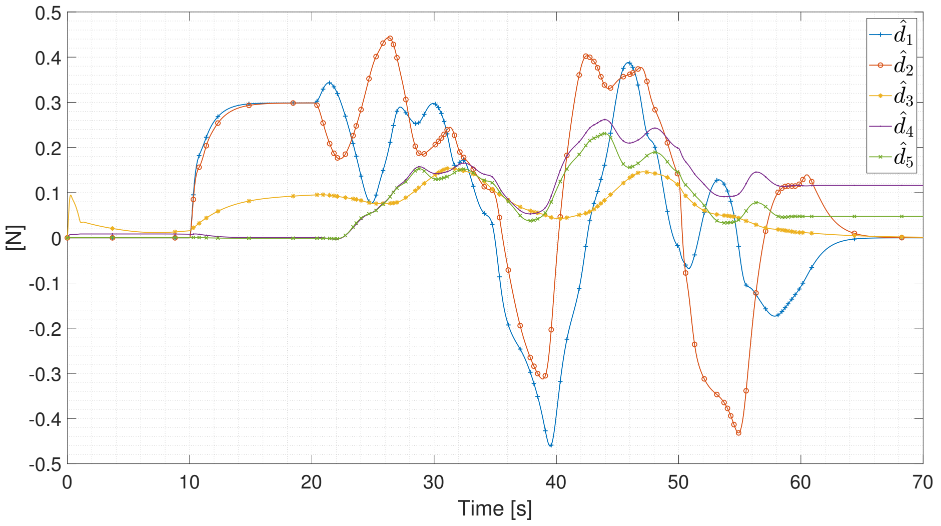

Figure 5.21: On-line estimation of the uncertain components of θ_p in equation (3.12) for the model with the CMRAC adaptive controller.

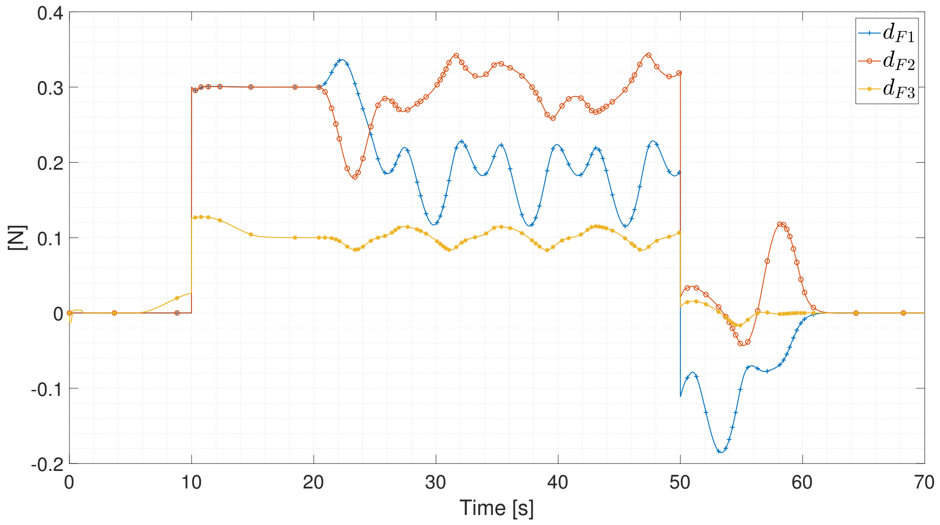

Figure 5.22: Trend of the disturbance vector d_F acting on the plant for the model with the CMRAC adaptive controller.

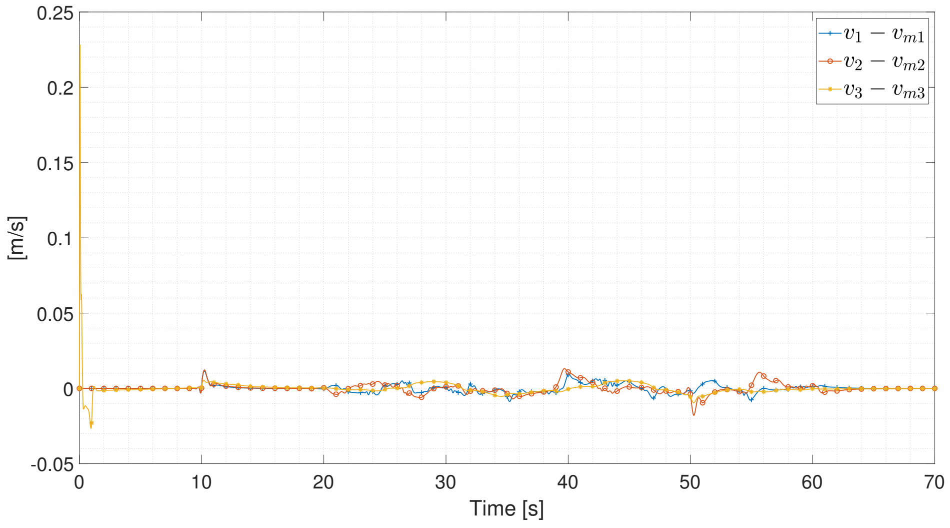

Figure 5.23: Error between the plant and the reference model translational velocity for the model with the CMRAC adaptive controller.

e_{abs}	**U.M.**	**CPID**	**Adaptive**
$\mathbf{e_{x1}}$	[m]	1.0738	1.0117
$\mathbf{e_{x2}}$	[m]	1.2305	0.9203
$\mathbf{e_{x3}}$	[m]	0.2693	0.2281
$\mathbf{v_{x1}}$	[m/s]	0.8620	0.8114
$\mathbf{v_{x2}}$	[m/s]	0.8996	0.7000
$\mathbf{v_{x3}}$	[m/s]	0.1305	0.1265
$\alpha_{\mathbf{x1}}$	[deg]	1.9978	1.9167
$\alpha_{\mathbf{x2}}$	[deg]	4.8906	4.5828
$\alpha_{\mathbf{x3}}$	[deg]	35.5575	31.7488
$\omega_{\mathbf{x1}}$	[deg/s]	4.9769	4.8572
$\omega_{\mathbf{x2}}$	[deg/s]	17.7730	17.4547
$\omega_{\mathbf{x3}}$	[deg/s]	30.4698	28.0924

Table 5.3: Comparison of the maximum absolute error e_{abs} between the model with the CPID and the CMRAC adaptive controller acting on the position.

e_{rms}	**U.M.**	**CPID**	**Adaptive**
$\mathbf{e_{x1}}$	[m]	0.5731	0.5581
$\mathbf{e_{x2}}$	[m]	0.6217	0.5912
$\mathbf{e_{x3}}$	[m]	0.0661	0.0495
$\mathbf{v_{x1}}$	[m/s]	0.5205	0.4966
$\mathbf{v_{x2}}$	[m/s]	0.2921	0.2685
$\mathbf{v_{x3}}$	[m/s]	0.0224	0.0109
$\alpha_{\mathbf{x1}}$	[deg]	1.1310	1.0935
$\alpha_{\mathbf{x2}}$	[deg]	2.5486	2.0335
$\alpha_{\mathbf{x3}}$	[deg]	15.7642	15.5016
$\omega_{\mathbf{x1}}$	[deg/s]	0.0098	0.0049
$\omega_{\mathbf{x2}}$	[deg/s]	0.0436	0.0223
$\omega_{\mathbf{x3}}$	[deg/s]	0.0895	0.0407

Table 5.4: Comparison of the root mean square error e_{rms} between the model with the CPID and the CMRAC adaptive controller acting on the position.

allows the system to correctly converge to the desired set points. Furthermore, the estimated variables contained in θ_p seem to converge to a constant value only when the quadrotor is in hovering conditions. When some tracking tasks are requested, the adaptive law does not seem able to find the θ_p ideal values. For what concern the error between the plant and the reference model translational velocity, it is clear that it is very small. After an initial fast transient at the beginning, where the error is bigger since the adaptive controller has to take some time to learn about the model uncertainties, it becomes very limited, meaning that the UAV behaviour is similar to the ideal one of the reference model. In general, all the variables do not present high frequency oscillations. This confirms the fact that a correct selection of the closed-loop gain can lead to the elimination of the oscillations while maintaining good tracking performances. As final remark, referring to tables 5.3 and 5.4, both the absolute and the root mean square errors for all the components of the state vector show lower values for the model where the CMRAC controller is implemented.

5.6 Conclusion

The analysis carried out until now show that an adaptive controller acting on the position dynamic can improve the performances of the system even if the only source of uncertainty is the disturbance. An adaptive architecture can learn from the measurements how to counteract them and, with a proper tuning, return good performances. In fact, it has also to be highlighted that when the designer has understood how the adaptive and closed-loop gains work, the tuning phase of such controllers has discovered to be easier than the CPID. Indeed, since the baseline controller is a simple proportional controller, the number of tuning gains is lower and consequently less complicated to tune.

Chapter 6

Attitude and Position Adaptive Controllers

Firstly, in this chapter we will focus on the results obtained by the tilt-arm simulator with the adaptive controller active on both the attitude and the position dynamic. In particular, the trends of each variable will be analysed, in order to understand if a complete adaptive architecture can be more suitable and more performing than other current solutions for the problem of controlling a tiltable propellers UAV. Finally, a Monte Carlo simulation will be carried out to highlight the system robustness to uncertainties that can be also present in the mixer matrix.

6.1 Tilt-Arm simulator

In the previous two chapters, the design of the adaptive controller has been split in the attitude and the position dynamic. As previously said, the reason of this choice lies in the fact that the linearised system has shown to be completely decoupled. In reality, the tiltable propellers UAV is a platform affected by important non linearities and perhaps a complete decoupled approach could not be satisfactory. On the contrary, the analysis showed before illustrates that the position dynamic does not affect too much the attitude one and vice-versa, especially when one of the twos is left ideal and the other with all the characteristic uncertainties. Therefore, it is correct to hope that if we activate the adaptive controller on both the attitude and the position dynamic, the results will not show very big discrepancies with respect to the previous ones. For these reasons, the goal of this section will be precisely to analyse the UAV performances, and to try to understand if a complete adaptive architecture will be suitable for futures real world applications.

Going into more detail regarding the numerical simulation, the trajectory chosen is the one illustrated in figure 6.1. More precisely, it is a trajectory where the UAV has to perform an “infinity shape” in the space, while changing its attitude.

This allows to evaluate both the posisition and the attitude dynamic. Given that

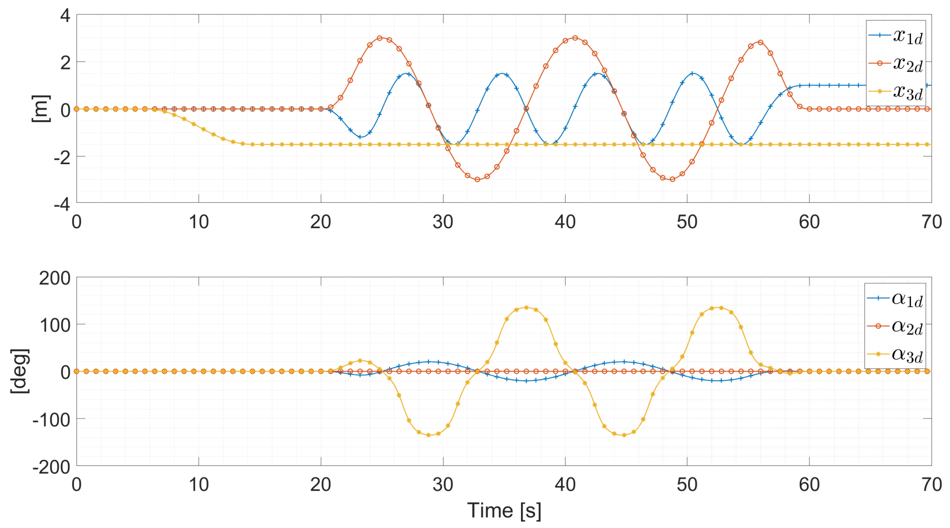

Figure 6.1: Infinity shape trajectory: desired position (on top) and Euler angles (bottom)

the two previous analyses on the attitude and on the position have been made also to correctly tune all the controllers, for this simulation all the adaptation and the closed-loop gains are left equal to the ones chosen before. Consequently, γ_a, γ_p, λ_a and λ_p have been taken respectively as equations (4.15), (5.7), (4.16) and (5.8). Like the gains, also all the other variables have been selected as the ones showed in the previous chapters. In particular, also in this case the disturbance acting on the position dynamic has been modelled as an offset term, which simulates a sudden gust of wind, plus an aerodynamic resistance term as in equation (5.9) and (5.10), while d_T acting on the attitude dynamic has been picked as a step function beginning at $t = 7$ seconds as done in section 4.8.2. For a complete overview of the simulator data, one should refer to appendix A. Figures 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9, 6.10, 6.11, 6.12, 6.13 and 6.14 show the results obtained.

As it is possible to see, in general the UAV performances seem to be satisfactory. More specifically, both the position and the attitude show good tracking and disturbance rejection capabilities. As expected, the errors increase each time some tasks are requested by the trajectory generator, but after a short transient they become very small. Under this perspective and considering the presence of system uncertainties and disturbances, the convergence time is fast enough to allow the tracking. Indeed, for what concerns the disturbance, it should be noted that when the d_T step occurs, all the variables present a pick, including the position and the translational velocity, demonstrating the fact that some couplings are present in the system. The sudden step is seen by the controller as an unexpected event that has to be counteracted. From this point of view, the adaptation seems to perform

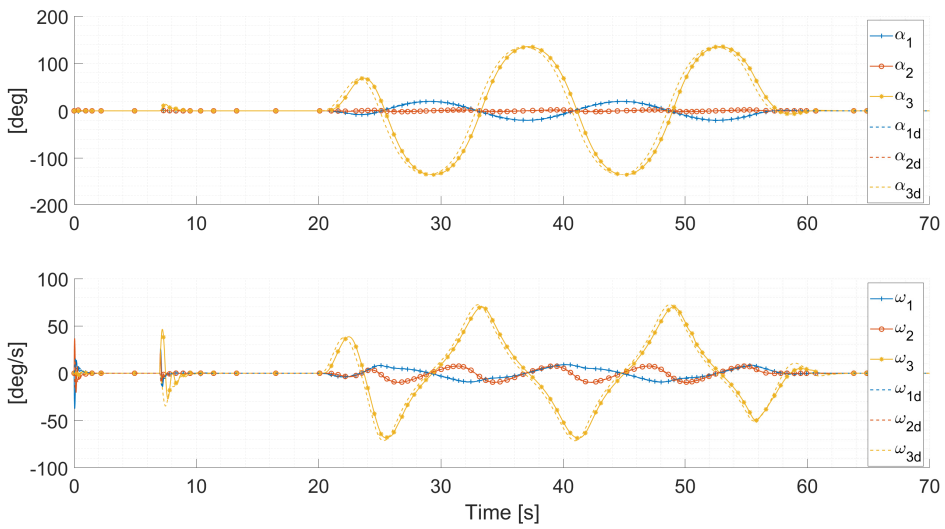

Figure 6.2: Plant and desired Euler angles (on top), and plant and desired angular velocities (bottom).

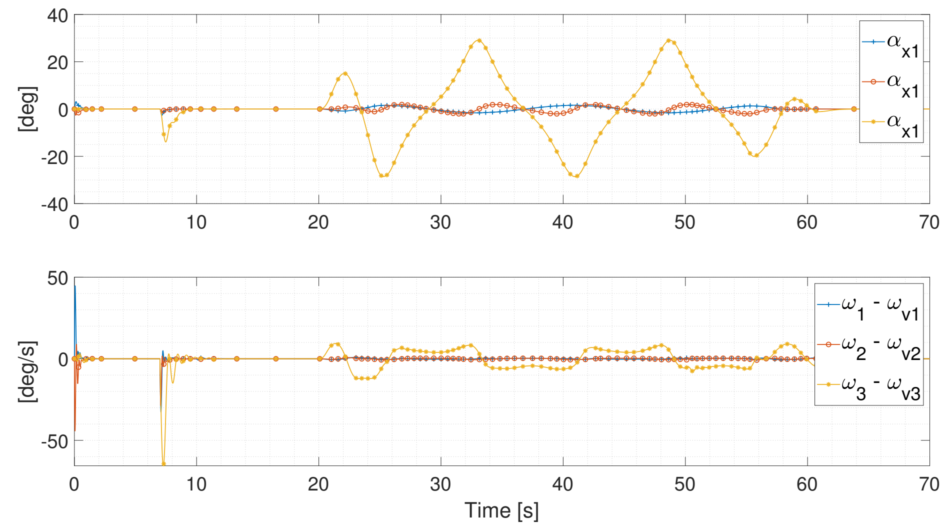

Figure 6.3: Euler angles error α_x (on top) and angular velocity error with respect to the outer-loop ω_v (bottom).

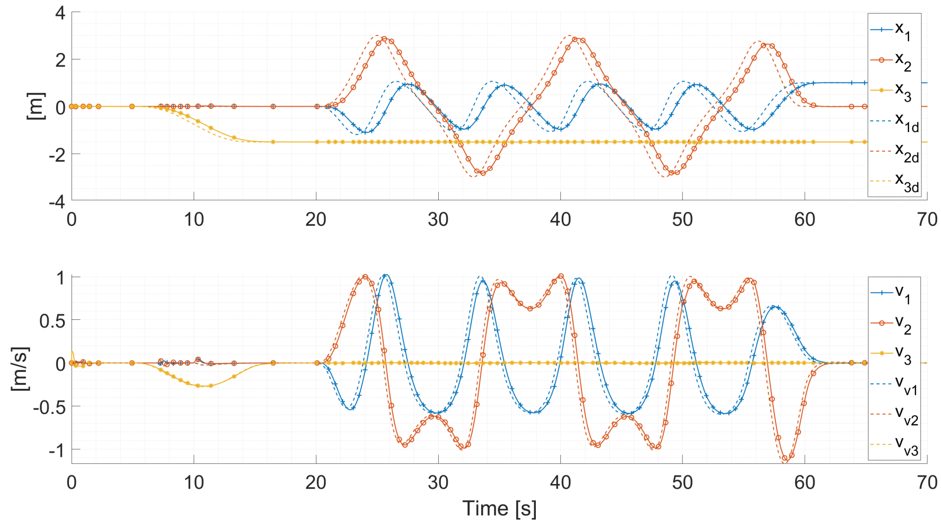

Figure 6.4: Plant and desired position (on top), and plant and desired translational velocity (bottom).

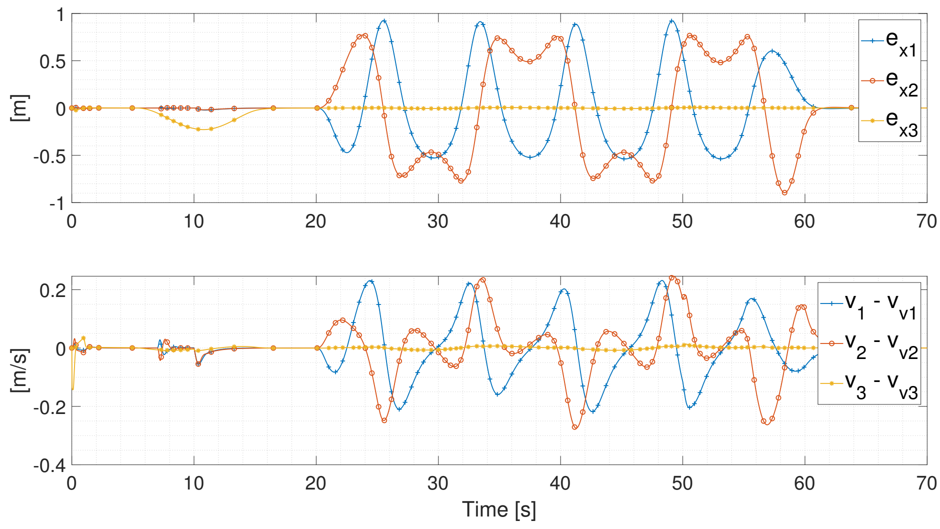

Figure 6.5: Position error e_x (on top) and translational velocity error with respect to the outer-loop v_v (bottom).

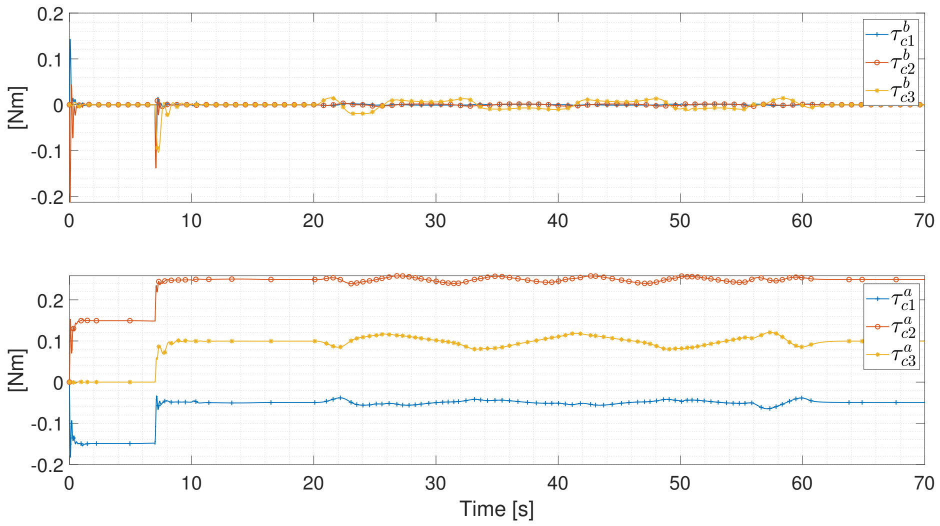

Figure 6.6: Baseline (on top) and adaptive (bottom) control input torque.

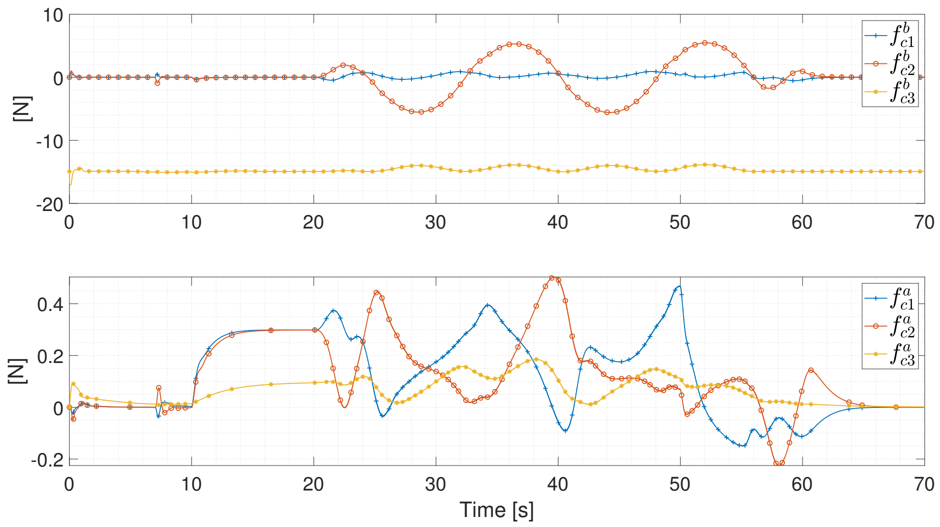

Figure 6.7: Baseline (on top) and adaptive (bottom) control input force.

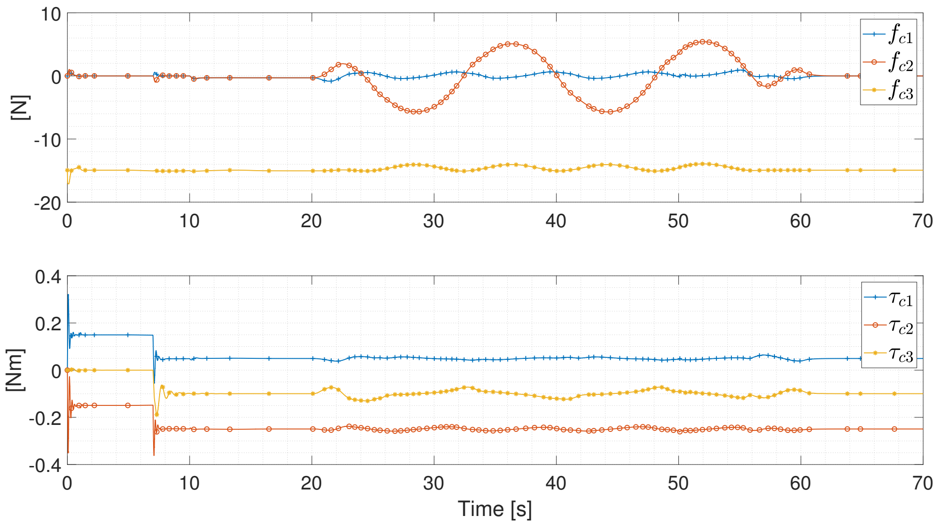

Figure 6.8: Force control input $f_c = f_c^b + f_c^a$ (on top) and torque control input $\tau_c = \tau_c^b + \tau_c^a$ (bottom).

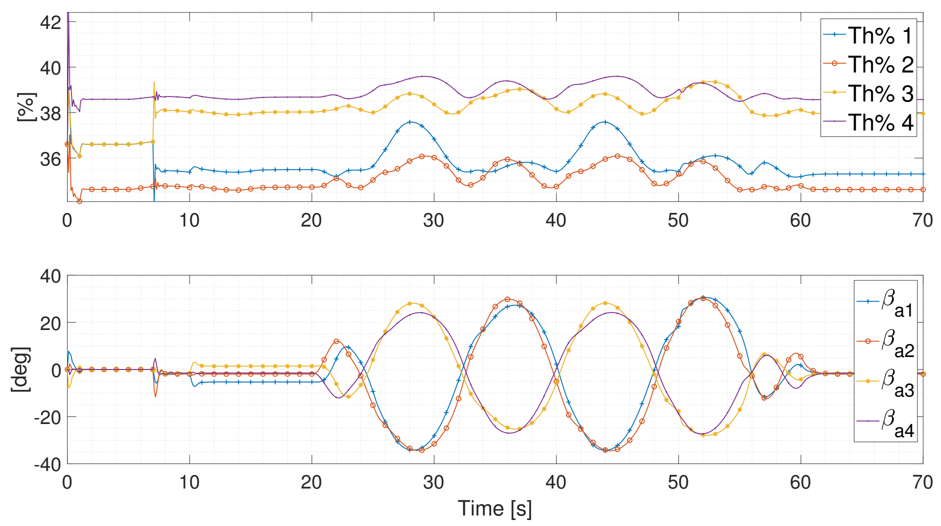

Figure 6.9: *Th*% (on top) and servo-actuators angular position β_a (bottom) requested by the four UAV's propellers.

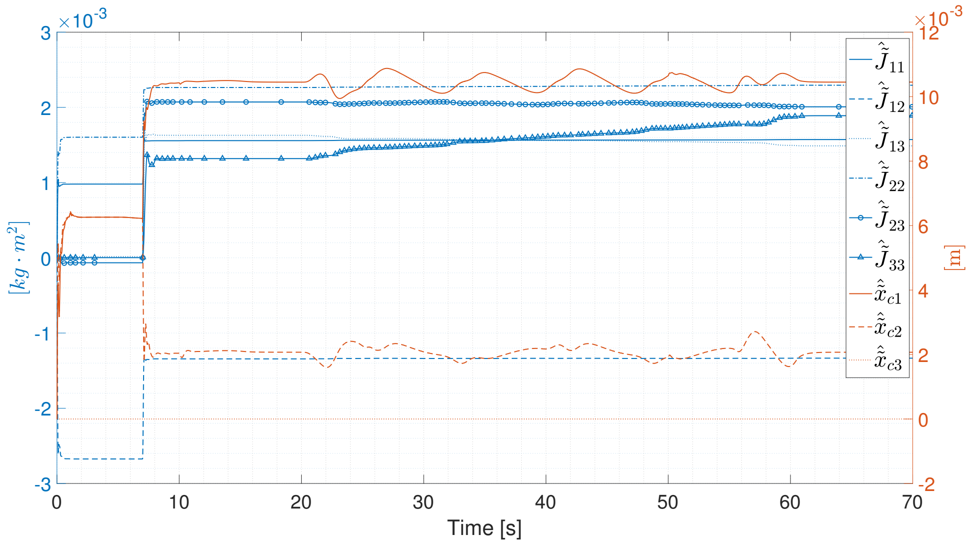

Figure 6.10: On-line estimation of the uncertain parameters contained in θ_a

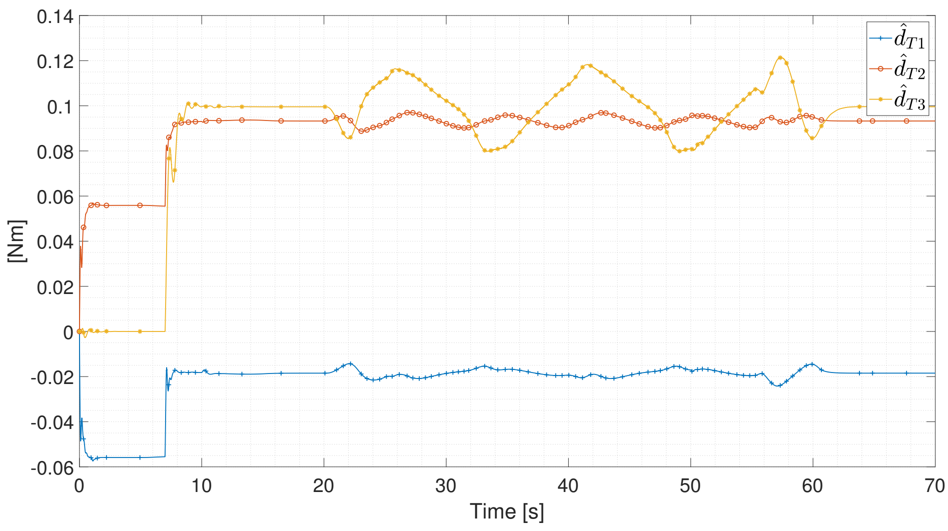

Figure 6.11: On-line estimation of the disturbance $\hat{d}_T$

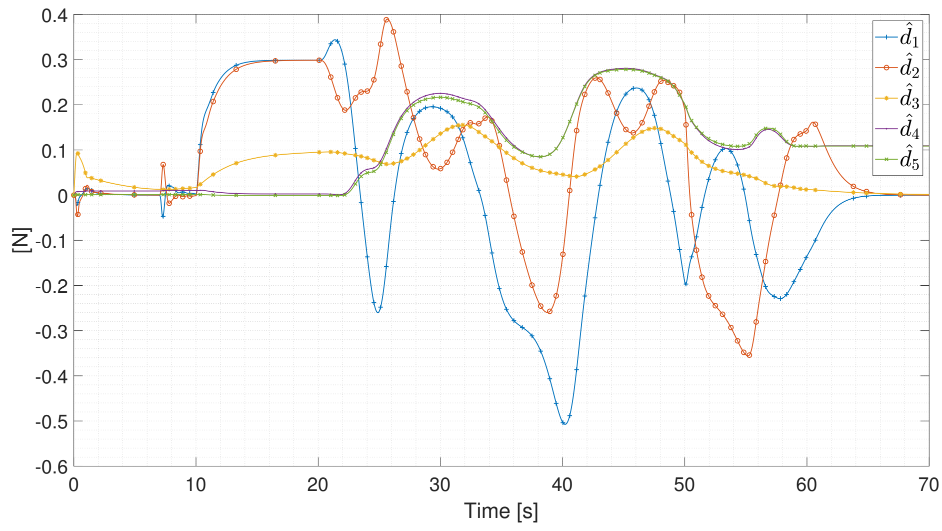

Figure 6.12: On-line estimation of the uncertain components of θ_p in equation (3.12)

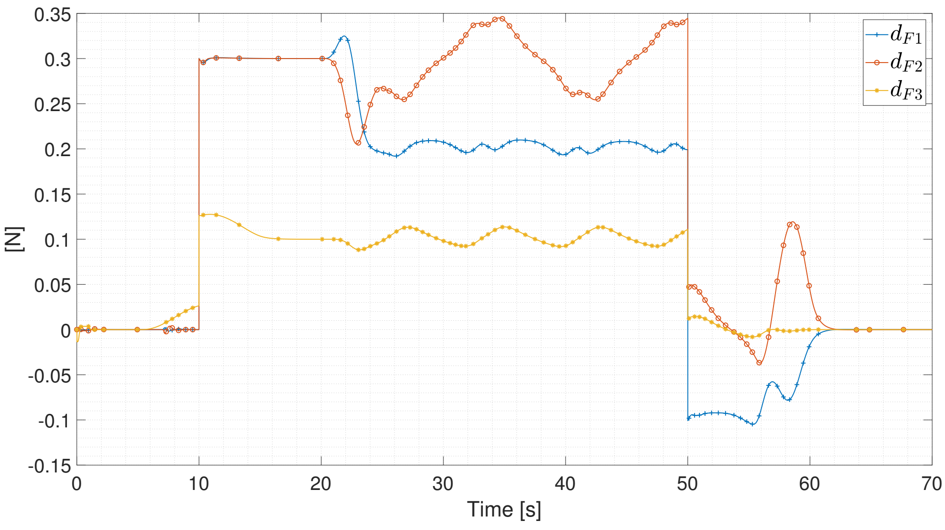

Figure 6.13: Trend of the disturbance vector d_F acting on the plant

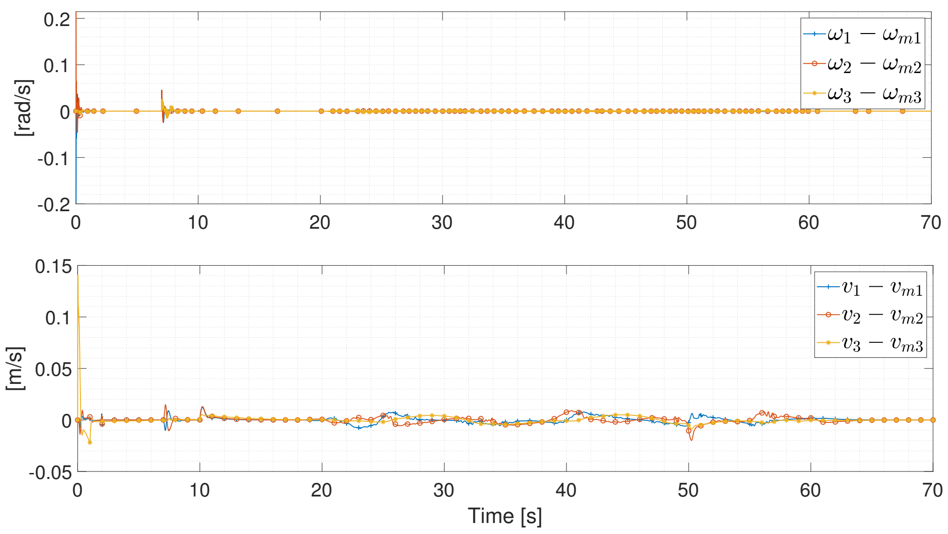

Figure 6.14: Error between the plant and the reference model angular velocity (on top) and the translational velocity (bottom)

very well since after a very short interval of time the pick vanishes. Regarding the control inputs, an analysis very similar to the one illustrated in chapters 4 and 5 can be carried out. The baseline control input tries to bring the system as close as possible to the requested trajectory, while the adaptive one counteracts all the uncertainties and disturbances in the dynamic. This allows the UAV to closely follow the desired set points. Moreover, all the control inputs are in the correct ranges of typical small scale quadrotors. The thrust and the servo actuators angular position requested by the four propellers do not exceed respectively the 40% and the ± 35 degrees. Concerning the estimation of the uncertain variables, as explained before this is done by an adaptive law, and more precisely by the Projection Operator. Similarly to what we have seen in the previous two chapters, both $\hat{\theta}_a$ and $\hat{d}_T$ seem to converge to a constant value, but different from the ideal one. The only exception are d_{T1} and d_{T2} that in reality seem to converge to the correct ideal value: 0.1 Nm. Instead, as before the estimation of θ_p shows greater difficulties. Furthermore, some words have to be spent on the error between the plant and the reference model. As can be seen, also in this simulation the system appears to have a behaviour very similar to the ideal one. After a very short initial transient where the adaptive controller has to "learn" and to "adapt" itself to the system uncertainties, the error shows a value very close to zero. Finally, it has to be highlighted that the activation of the adaptive controllers on both the attitude and position dynamic has increased the computational effort needed for the simulations. This to remind that the more complex the adaptive architecture, the faster and more expensive hardware required.

6.2 Monte Carlo simulation

6.2.1 Overview

Today, the Monte Carlo Simulation technique is used extensively for modelling uncertain situations. Although we have a profusion of information at our disposal, it is difficult to predict the future with absolute precision and accuracy. This can be attributed to the dynamic factors that can impact the outcome of a course of action. Monte Carlo Simulation enables us to see the possible outcomes of a decision, which can thereby help us make better decisions under uncertainty. It is a mathematical technique that generates random variables for modelling risk or uncertainty of a certain system. As can be seen in figure 6.15, the random variables or inputs are generated from probability distributions such as normal, logarithmic, uniform etc. Then, after their implementation in the model, different iterations or simulations are run for generating the outcomes. Monte Carlo Simulation is the most tenable method used when a model has uncertain parameters or a dynamic complex system needs to be analysed. It is a probabilistic method for modelling risk in a system and it provides a probabilistic estimate of the uncertainty in a model. As a drawback, this kind of method is very expensive from the computational power point of view, since it requires the simulation of the same model for a large number of times. Moreover, the results fidelity is strongly influenced by the number of iterations. The higher are the number of simulations, the better will be the precision of the outcomes.

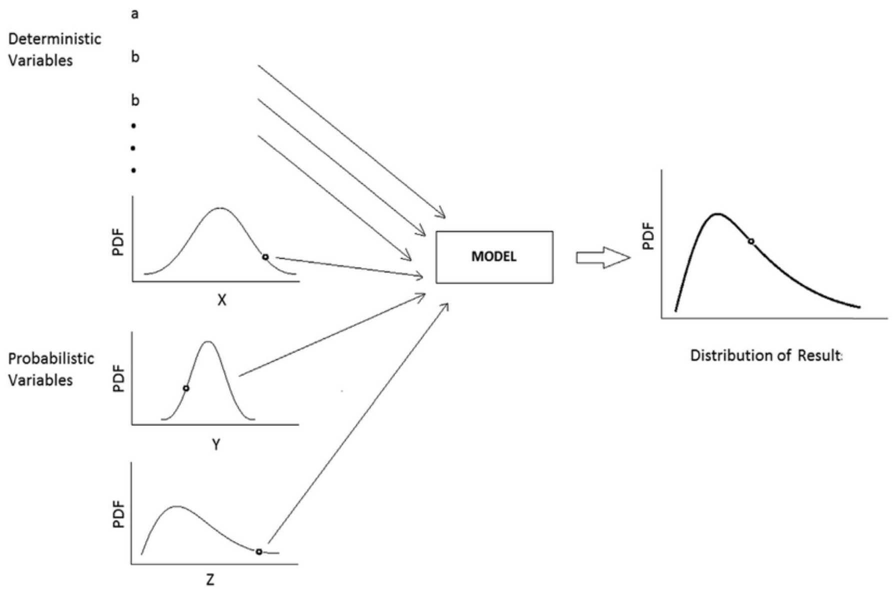

Figure 6.15: Schematic of Monte Carlo simulation [6]

6.2.2 Uncertain allocation algorithm

All the simulations computed until now assume as uncertain parameters only some variables that are directly involved in the system dynamic: for instance the inertia tensor, the center of mass position and the disturbances. In reality, we have seen in section 1.1.1 that an allocation algorithm is also needed to compute the actual physical inputs ω_{ri} and β_{ai} from the control force and torque. The logic of this algorithm is based on the knowledge of the mixer matrix $M(\sigma, b)$, which is a function of the arm length b and of the actuators properties σ. These latter variables are difficult to estimate with high precision, especially σ. Therefore, in real world applications some uncertainties can also enter in the allocation algorithm. As can be seen in figures 1.6 and 1.7, the pseudo-inverse of the mixer matrix is used a first time to compute ω_{ri} and β_{ai} from f_c and τ_c computed by the controllers to simulate the presence of the actuators. Then, M is directly exploited a second time in the plant to make the opposite process: the control input force and torque are again computed from ω_{ri} and β_{ai} since the quadrotor dynamic system accepts only f_c and τ_c as control inputs. To be more precise, in mathematical terms, the first time we compute f_u as seen in equation (1.7), then the second one we calculate f_c and τ_c as showed in equation (1.5). With regards to our problem, the mixer matrix M takes the following form:

$$M = \begin{bmatrix} -\frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{1}{4} & \frac{2^{\frac{1}{2}}\sigma}{4b} & \frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{\sigma}{4(b^2+\sigma^2)} \\ -\frac{2^{\frac{1}{2}}\sigma}{4b} & \frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{1}{4} & -\frac{2^{\frac{1}{2}}\sigma}{4b} & \frac{2^{\frac{1}{2}}\sigma}{4b} & \frac{\sigma}{4(b^2+\sigma^2)} \\ \frac{2^{\frac{1}{2}}\sigma}{4b} & \frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{1}{4} & -\frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{2^{\frac{1}{2}}\sigma}{4b} & -\frac{\sigma}{4(b^2+\sigma^2)} \\ \frac{2^{\frac{1}{2}}}{4} & \frac{2^{\frac{1}{2}}}{4} & 0 & 0 & 0 & \frac{b}{4(b^2+\sigma^2)} \\ -\frac{2^{\frac{1}{2}}}{4} & \frac{2^{\frac{1}{2}}}{4} & 0 & 0 & 0 & \frac{b}{4(b^2+\sigma^2)} \\ -\frac{2^{\frac{1}{2}}}{4} & -\frac{2^{\frac{1}{2}}}{4} & 0 & 0 & 0 & \frac{b}{4(b^2+\sigma^2)} \\ \frac{2^{\frac{1}{2}}}{4} & -\frac{2^{\frac{1}{2}}}{4} & 0 & 0 & 0 & \frac{b}{4(b^2+\sigma^2)} \end{bmatrix} \tag{6.1}$$

while its pseudo-inverse becomes:

$$M^+ = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ 0 & 0 & 0 & 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ \frac{\sqrt{2}}{2}b & -\frac{\sqrt{2}}{2}b & -\frac{\sqrt{2}}{2}b & \frac{\sqrt{2}}{2}b & \frac{\sqrt{2}}{2}\sigma & \frac{\sqrt{2}}{2}\sigma & -\frac{\sqrt{2}}{2}\sigma & -\frac{\sqrt{2}}{2}\sigma \\ \frac{\sqrt{2}}{2}b & \frac{\sqrt{2}}{2}b & -\frac{\sqrt{2}}{2}b & -\frac{\sqrt{2}}{2}b & \frac{\sqrt{2}}{2}\sigma & -\frac{\sqrt{2}}{2}\sigma & -\frac{\sqrt{2}}{2}\sigma & \frac{\sqrt{2}}{2}\sigma \\ -\sigma & \sigma & -\sigma & \sigma & b & b & b & b \end{bmatrix} \tag{6.2}$$

Usually, if we imagine a real world application and if we follow the same idea of the adaptation theory discussed before, the uncertainties are particularly present in the tiltrotor block of figure 1.6, given that the controllers block is the one that would be actually implemented in a real prototype. In fact, the tiltrotor block tries simply to simulate as close as possible the real UAV dynamic. For these reasons, in the simulation that we are going to implement, we will assume that the first transformation that computes ω_{ri} and β_{ai} is exact, while the second one that computes f_c and τ_c is affected by uncertainties.

6.2.3 Simulation set-up

In order to understand the UAV robustness to this new source of uncertainty, a Monte Carlo simulation will be carried out. This choice comes from the fact that the simulator and the dynamic system is very complex. The Monte Carlo method is the only way to introduce uncertainty in the system and to understand how much these uncertainties affect the performance of the quadrotor.

The first thing to do to set up the simulation is to generate the necessary samples. Since we have seen that the second transformation, which computes the control input force and torque, is the one affected by uncertainty, it has been decided to produce a set of 20 samples for both the b and the σ variables that appear in equation (6.2). Since we have 20 elements for each variable, the total number of iterations needed to complete the simulation is $20^2 = 400$. The samples have been generated trough two Gauss distributions with a mean value η equals to the ones adopted in the tilt-arm simulations of chapters 4 and 5, and with a standard deviation μ such that the 99.7% or 3μ of all the created samples do not deviate more than 20% from the b and σ mean values. This has been accomplished by taking:

$$\mu = \frac{20}{100}\frac{1}{3}\eta. \tag{6.3}$$

Reminding that for a better overview of the all used data one should refer to appendix A, table 6.1 shows the parameters used to generate the Gauss distributions illustrated in figure 6.16, where $g(b)$ and $g(\sigma)$ are respectively the probability density of b and σ. Instead, as previously said, given that equation (6.1) has been

	Mean η	**Std.** μ
b	$\frac{0.45}{2}$	0.0150
σ	0.0076	5.06×10^{-4}

Table 6.1: Mean and standard deviation used to create the Gauss distributions for the b and σ variables

used to compute f_u, the parameters inside equation (6.1) have been left equal to the b and σ mean values reported in table 6.1. Finally, the desired trajectory for the Monte Carlo simulation has been chosen as the trajectory showed in figure 4.1.

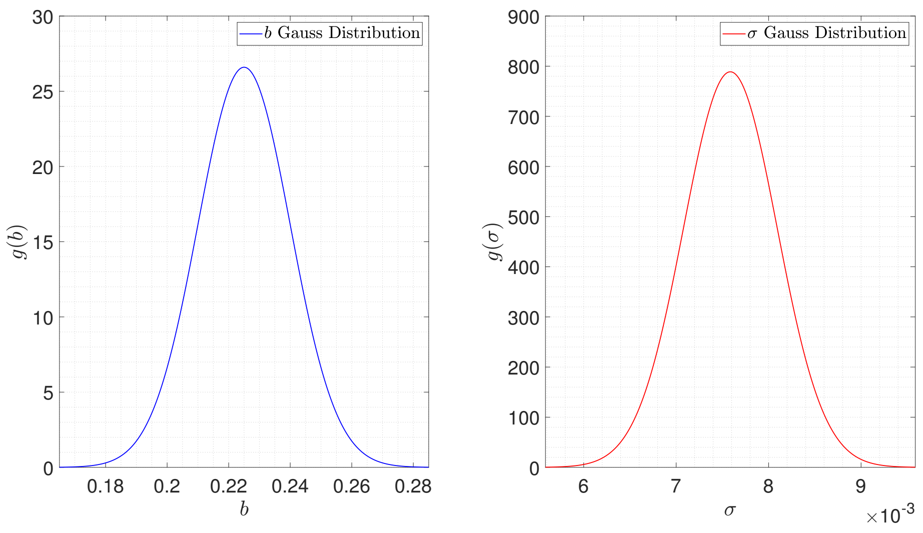

Figure 6.16: Gauss distribution of the variable b (left) and σ (right) from which the sets of samples have been generated

6.2.4 Results

In order to have a better comprehension of the outcomes, similarly to what we have done in chapters 4 and 5, the Monte Carlo method just explained has been applied to both the models with the CPID and CMRAC adaptive controller. More specifically, for comparison reasons, the same σ and b samples created from the Gauss distributions have been used to simulate the model showed in section 1.5 and the model presented in section 6.1.

Figures 6.17, 6.18, 6.19, 6.20, 6.21, 6.22, 6.23 and 6.24 show the results obtained as a function of the number of iterations, while figures 6.25, 6.26, 6.27, 6.28, 6.29, 6.30, 6.31 and 6.32 illustrate the same outcomes but as a function of the generated b and σ samples.

As it is possible to see, in general the CPID architecture shows a flatter behaviour than the CMRAC one, where the plots present a more varying trend. However, the errors computed by the model with the adaptive controller show a value that on average is smaller with respect to the other model. Therefore, this means that overall the CMRAC approach is more robust with respect to the variation of b and σ inside the mixer matrix. Moreover, as can be appreciated from the 3D plots, since along the b axis we have a flatter behaviour compared to the σ one, both the controller architectures have been showed to be more robust with respect to the variation of b than with respect to the variation of σ. The only exceptions are the position and translational velocity root mean square tracking errors of the CMRAC architecture that have been demonstrated to be particularly sensible also along the b axis.

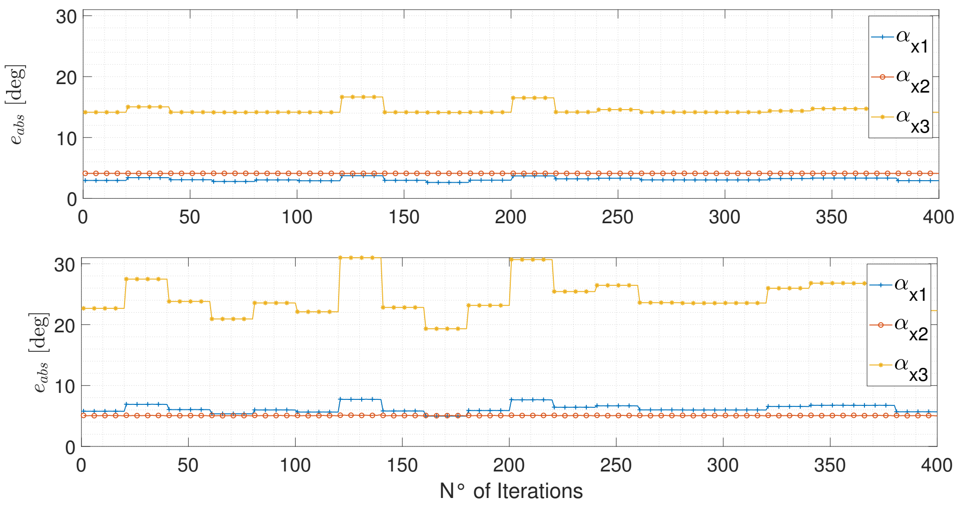

Figure 6.17: Euler angles absolute tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

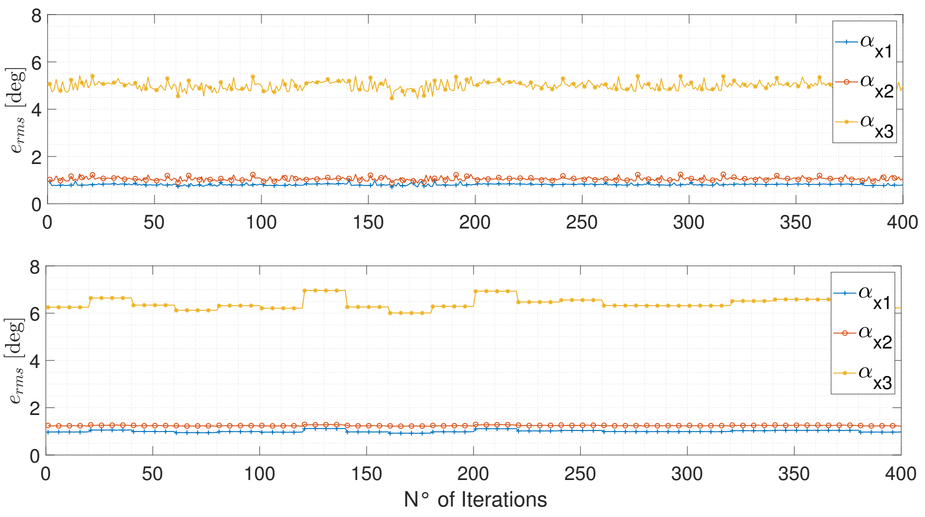

Figure 6.18: Euler angles root mean square tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

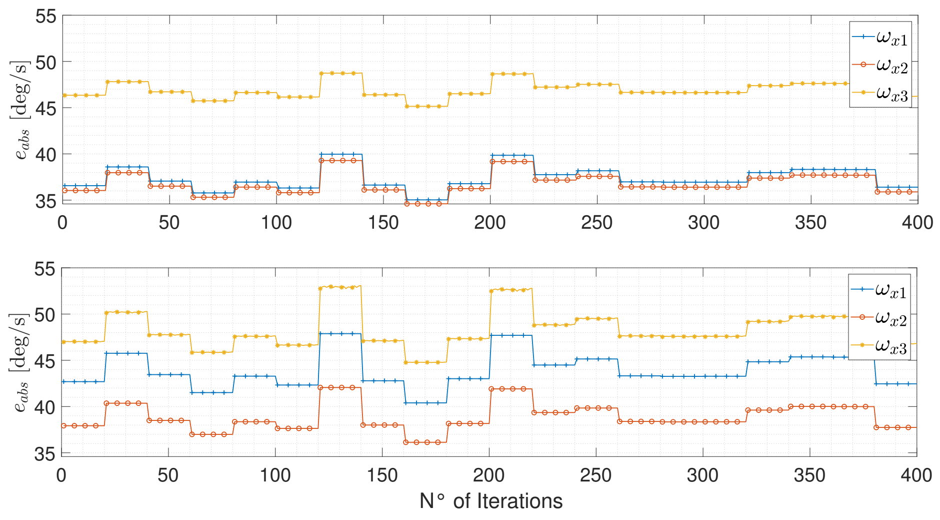

Figure 6.19: Angular velocity absolute tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

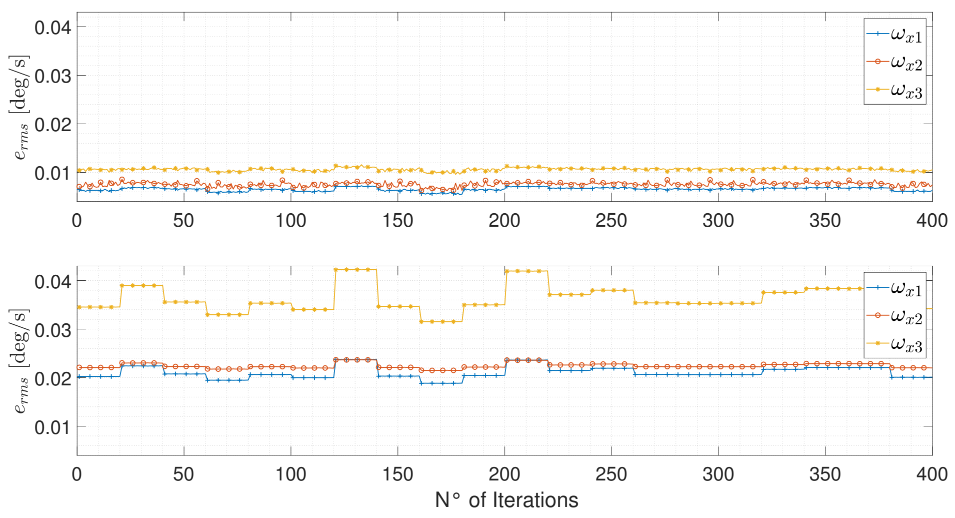

Figure 6.20: Angular velocity root mean square tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

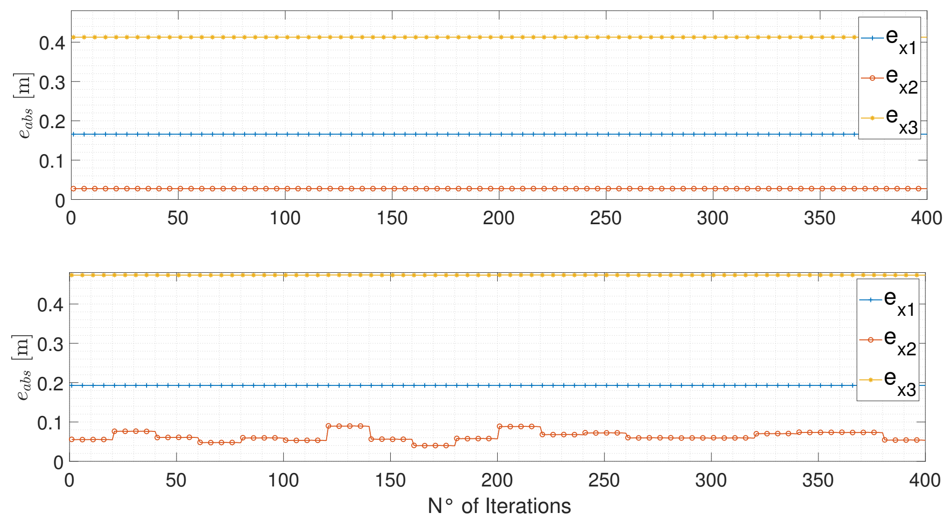

Figure 6.21: Position absolute tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

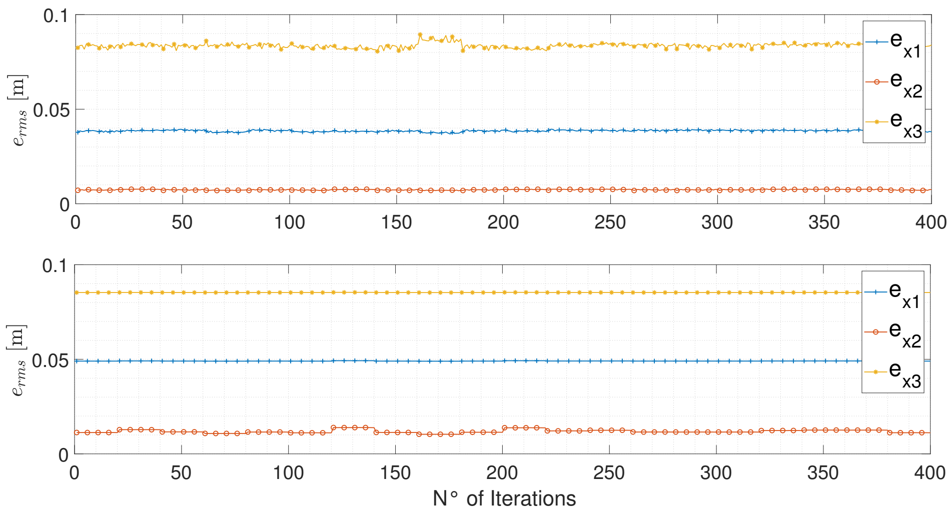

Figure 6.22: Position root mean square tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

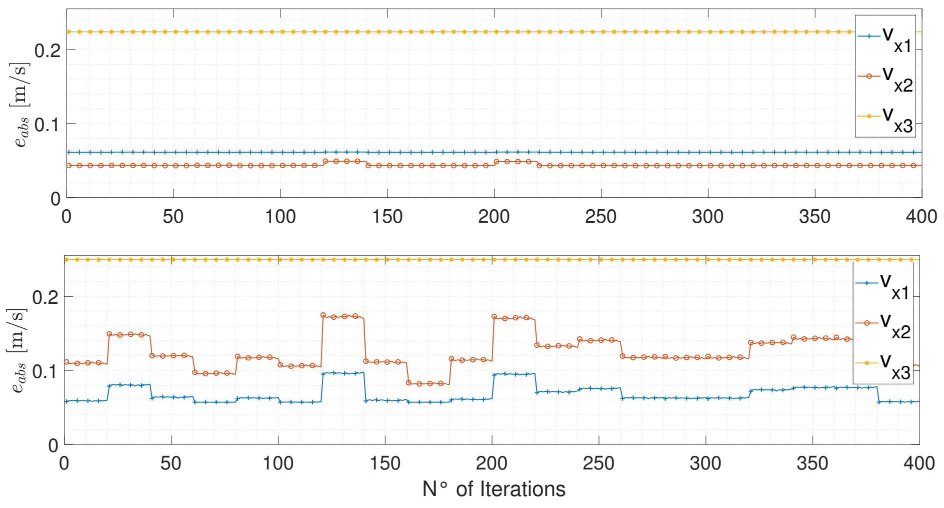

Figure 6.23: Translational velocity absolute tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

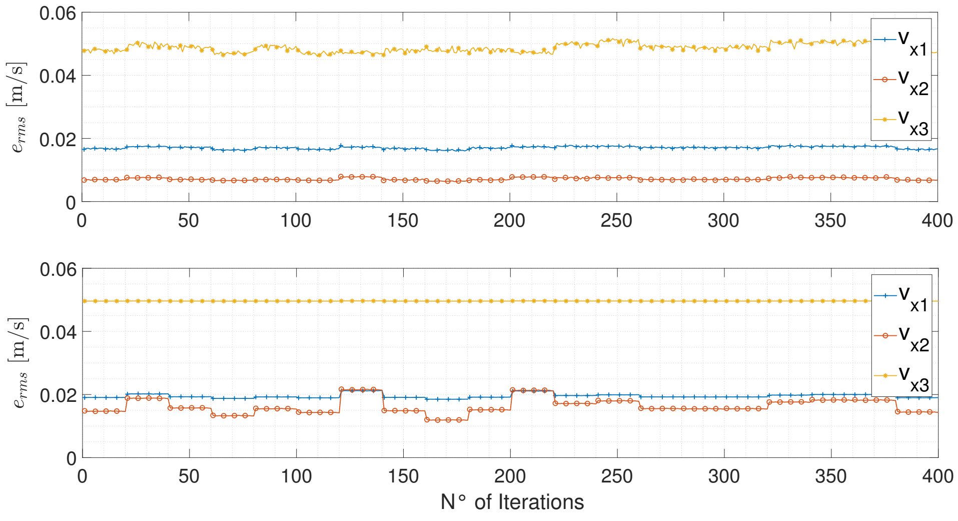

Figure 6.24: Translational velocity root mean square tracking error as a function of the number of iterations for the model with the CMRAC adaptive controller (on top) and with the CPID controller (bottom).

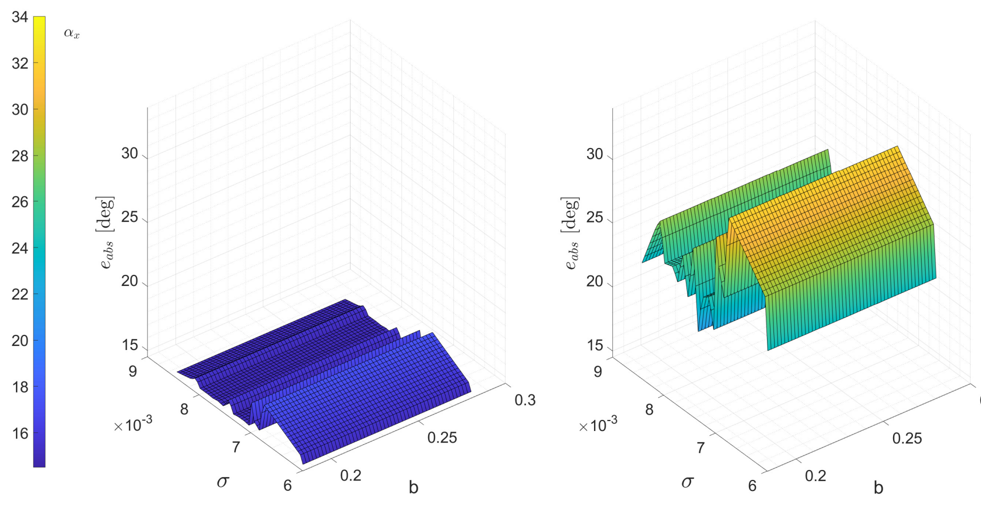

Figure 6.25: Euler angles absolute tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

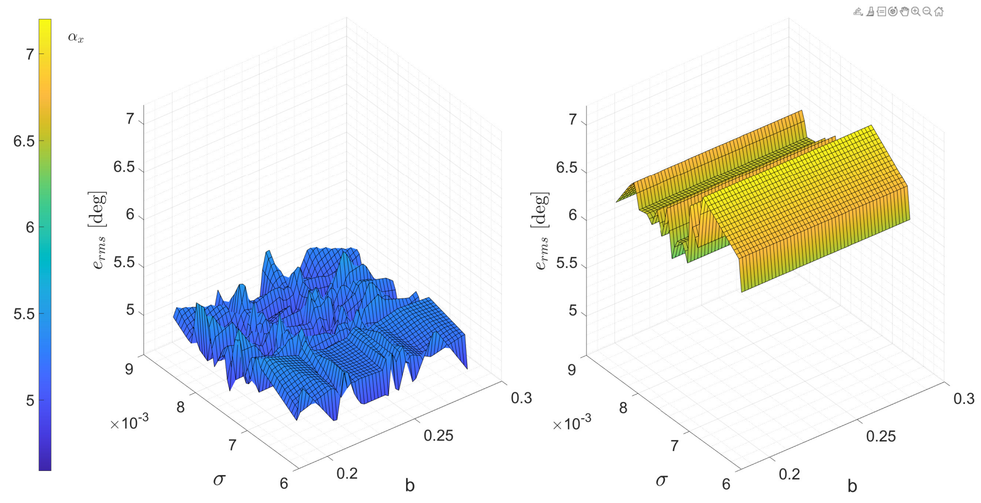

Figure 6.26: Euler angles root mean square tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

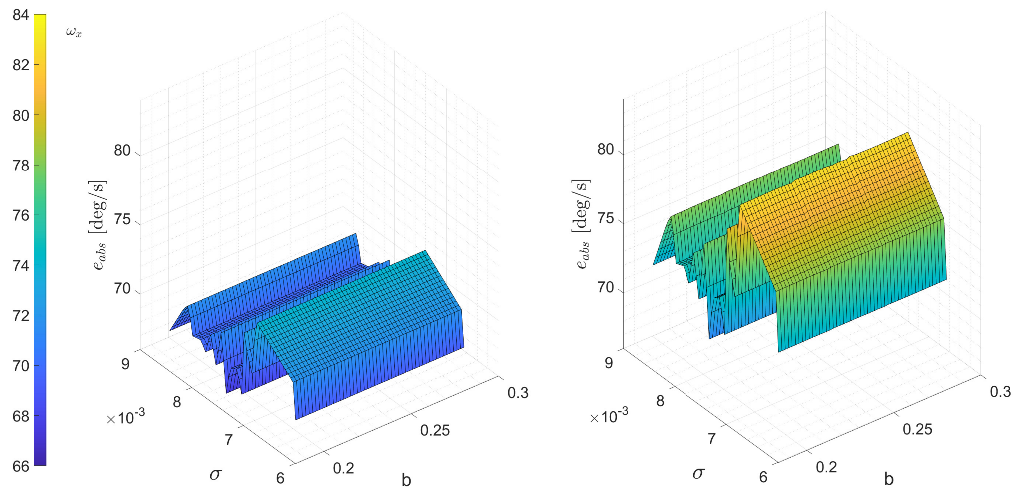

Figure 6.27: Angular velocity absolute tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

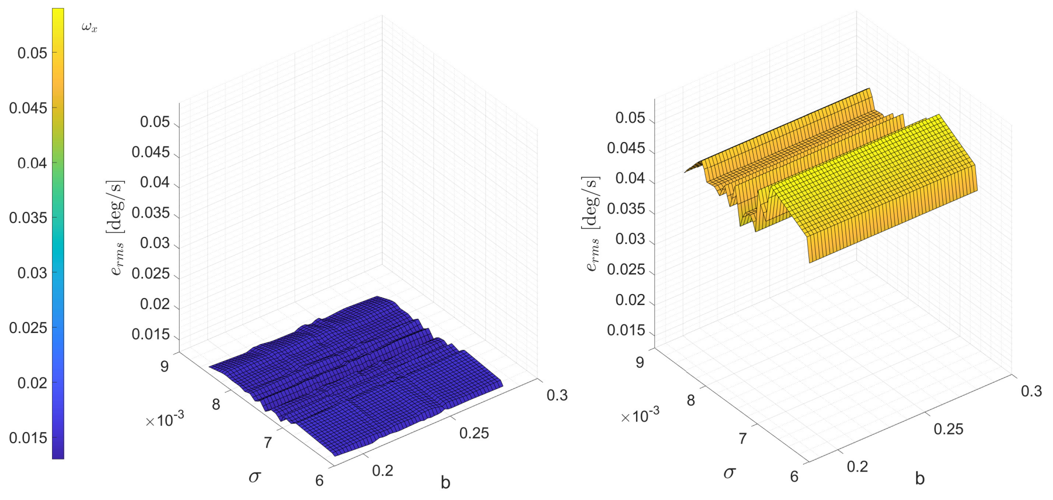

Figure 6.28: Angular velocity root mean square tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

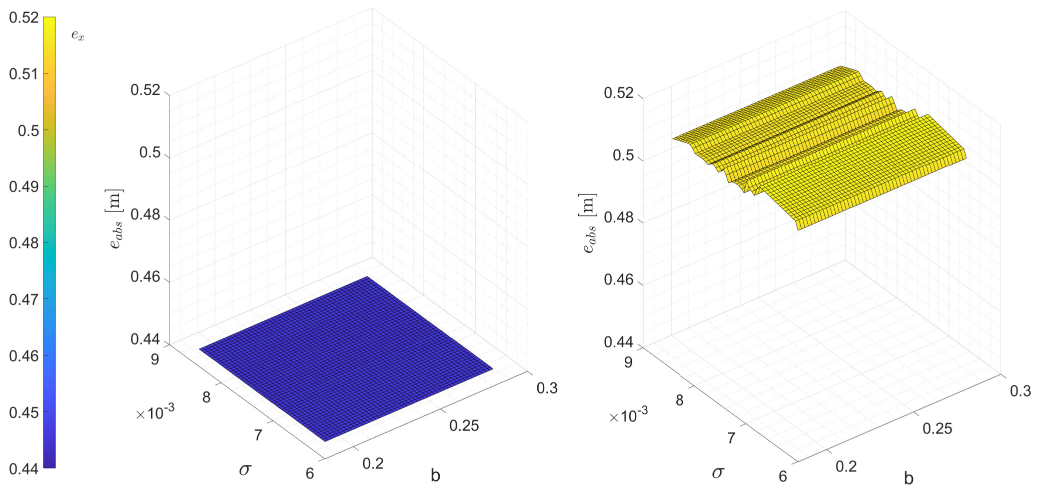

Figure 6.29: Position absolute tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

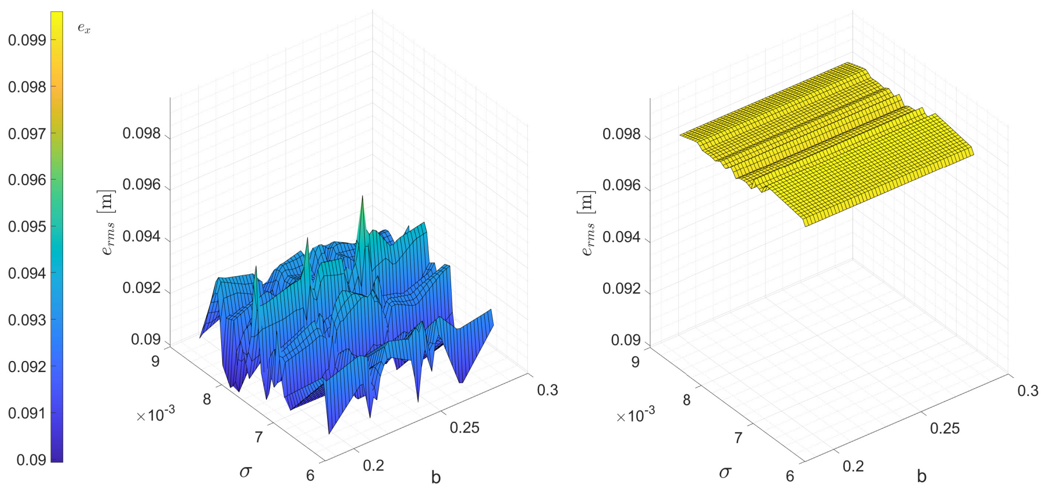

Figure 6.30: Position root mean square tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

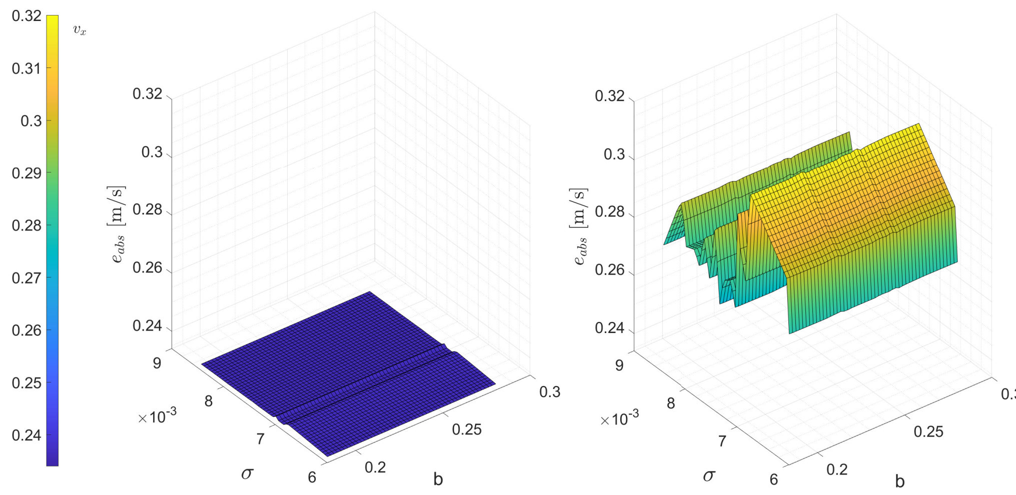

Figure 6.31: Velocity absolute tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

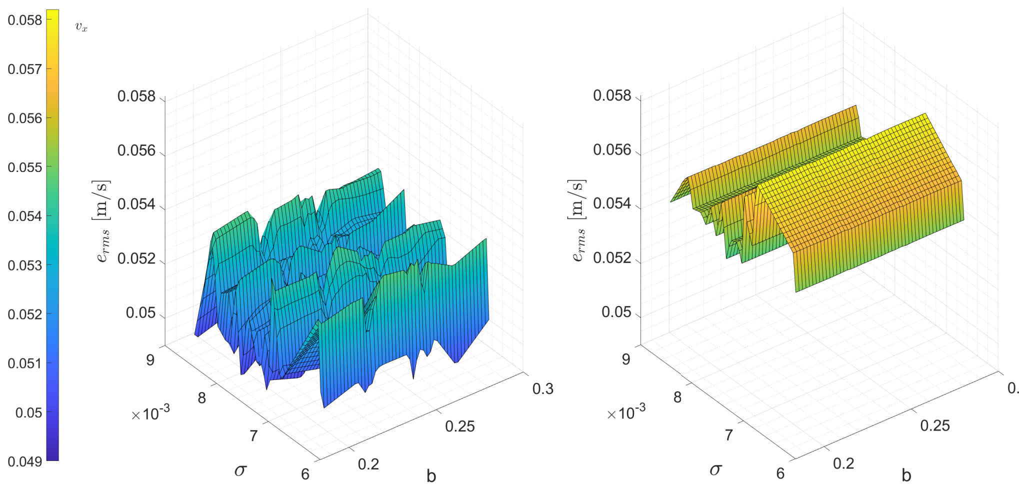

Figure 6.32: Velocity root mean square tracking error as a function of the generated b and σ samples for the model with the CMRAC adaptive controller (left) and with the CPID controller (right).

6.3 Conclusion

The simulations carried out with the adaptive controllers active on both the position and attitude dynamic have highlighted very good tracking and disturbance rejection capability performances. The CMRAC approach chosen has proved to be the correct choice, since it allows the designer to set the speed of convergence and the behaviour of the transients. Finally, a Monte Carlo simulation has been set up to analyse the system robustness to uncertainties that can be present in the mixer matrix. More specifically, the method has been used to transform two deterministic variables to random ones. Since in real world applications there is the possibility that these kinds of uncertainties are present, the results have demonstrated that the CMRAC architecture is more robust and more efficient than the well proven CPID one.

Conclusions

The purpose of this thesis has been to improve the current research in the field of control systems for tiltable propellers UAV. It has been demonstrated that this latter is affected by high non-linear dynamics, making the problem of controlling difficult. Many non linear control algorithms have been implemented, such as the ones reported in [1] and [2], but they have shown some limitations: they have been derived from simplified dynamical models that limit the UAV performance. In this context, a possible solution has been found in adaptive algorithms, where the word "adaptive" refers to their ability to learn from the state measurements and to adapt their-self against the uncertainties of the system. In this work, two different adaptive architectures have been designed and then have been numerically simulated: a Model Reference Adaptive Controller (MRAC) and a Closed-loop Reference Adaptive Controller (CMRAC).

The activities conducted started with an introductory first chapter in which the complete mathematical dynamical model of the quadrotor under study has been shown and analysed, along with an its simplified version more suitable for the control problem. Then, after a brief overview of the current state of the art control systems for tiltable propellers UAV, the most complex numerical simulator, which takes into account for the actuators dynamics and saturations, system non-linearities, external disturbances and not null center of mass position, has been presented. Since the CPID architecture has shown results comparable with other current solutions [2], it has been selected as the reference control architecture with respect to which all the results have been compared. Afterwards, the simulator has been exploited to highlight the limitations of some control designs.

In the second chapter, after a brief introduction about the MRAC and the CMRAC approaches, the control architecture chosen has been described. More specifically, it has been found that the best performances can be obtained by integrating an adaptive controller with a baseline one. For this reason, a simple proportional controller has been coupled with an adaptive controller. The first one has the purpose of bringing the system as close as possible to the desired trajectory, while the second one has been used to counteract the disturbances, uncertainties and non-linearities of the system. Moreover, the instability phenomena that usually affect these kinds of adaptive architectures and their possible solutions have been shown. In particular, the Projection Operator has the capability to increase the system robustness, to avoid the drifting of the uncertain parameters and to bound the overall process.

Taking the complete UAV dynamical model presented in chapter 1, the third chapter began with the derivation of the plant linear model. This latter has been discovered to be completely decoupled, in the sense that all the equations are independent. For this reason, it has been chosen to adopt a decoupled design approach: the control adaptive system has been designed separately for the attitude and for the position. Then, the computed linearised model along with the identified uncertain parameters has been used to derive the complete uncertain model. This latter has the characteristic of separating the uncertain portion of the model and it has been particularly useful for the implementation of the adaptive controllers.

Following a decoupled approach, in the fourth chapter the details regarding the baseline controller, the reference model and the chosen adaptive law have been illustrated. Afterwards, a simplified linear simulator, which differ from the more complex one by the absence of the actuators modelling, has been presented and exploited to analyse the first results obtained using a model in which the MRAC and CMRAC has been implemented. In this way, we have been able to understand the main peculiarities of the two approaches and to select the most suitable control architecture for the problem under study. The simulations have been carried out also to understand how the adaptation and the closed-loop gains affect the UAV performances. Then, the CMRAC has been selected as the most satisfactory control system and its performances have been further proven on more complex numerical simulators such as the non-linear simplified one and the complete tilt-arm one showed in Chapter 1. The outcomes obtained have shown very good tracking performances and disturbance rejection capabilities, comparable or even better than the ones obtained with the classical CPID. Moreover, due to the adaptive nature of the control algorithm, the tuning process has been discovered to be more straightforward than the CPID.

The fifth chapter has a structure very similar to chapter 4, but it refers to the UAV position dynamic. As before, the details about the baseline controller, the reference model and the adaptive law have been analysed. Given that the position has been demonstrated to be less affected by the system uncertainties, the CMRAC controller has been verified directly on the tilt-arm simulator. The results obtained show that an adaptive controller on the position can actually improve the UAV performance, especially if we consider that in real world applications the external disturbances can be very unpredictable. As in the previous case, also for the position the tuning process has not been too much complicated.

In the last chapter the adaptation has been activated on both the attitude and the position dynamics. The outcomes obtained by the tilt-arm simulator have been analysed and discussed. It has been shown that, in simulation environment, the adaptive controller has not been able to bring the estimation of the uncertain parameters to their respective ideal values. This has not avoided the UAV from performing the requested task with satisfactory tracking performances and disturbance rejection capabilities, demonstrating that a decoupled design approach and the use of a linearised uncertain model have been the correct engineering compromise. Moreover, as additional verification, a Monte Carlo simulation has been carried out to assess the quadrotor robustness against the possible presence of uncertainties in the mixer matrix. More specifically, the results have shown that the model with the CMRAC has been more robust with respect to the one with the CPID.

In conclusion, some indications of future developments and improvements for the present thesis work are reported below:

- Prove the results experimentally on a real prototype.
- Evaluate the possibility to improve the adaptive controller implementing an L_1 architecture.

Bibliography

[1] P. Gattazzo. Nonlinear control of a tilt-arm quadrotor uav. Master's thesis, Politecnico di Milano, 2016-2017.

[2] D. Invernizzi, M. Giurato, P. Gattazzo, and M. Lovera. Comparison of control methods for trajectory tracking in fully actuated unmanned aerial vehicles. *IEEE Transactions on Control Systems Technology*, pages 1–14, 2020.

[3] A. Ioannou and Sun Jing. *Robust Adaptive Control.* Prentice-Hall, Inc., USA, 1995.

[4] E. Lavretsky and K. Wise. *Robust and Adaptive Control: With Aerospace Applications.* Advanced Textbooks in Control and Signal Processing. Springer London, 2012.

[5] E. Lavretsky and Travis E. Gibson. Projection operator in adaptive systems. *ArXiv*, abs/1112.4232, 2011.

[6] Thuy Vu, Erik Loehr, and Douglas Smith. Probabilistic analysis and resistance factor calibration for deep foundation design using monte carlo simulation. *Heliyon*, 4:e00727, 08 2018.

[7] M. Kamel, S. Verling, O. Elkhatib, C. Sprecher, P. Wulkop, Z. Taylor, R. Siegwart, and I. Gilitschenski. The voliro omniorientational hexacopter: An agile and maneuverable tiltable-rotor aerial vehicle. *IEEE Robotics Automation Magazine*, 25(4):34–44, 2018.

[8] Px4-community. Available at: `https://docs.px4.io/en/`, 2018.

[9] D. Brescianini, M. Hehn, and R. D'Andrea. Nonlinear quadrocopter attitude control. Technical report, ETH, Zürich, Switzerland, 2013.

[10] MATLAB. *version R2020a.* The MathWorks Inc., Natick, Massachusetts, 2020.

[11] A. Russo. Adaptive control of multirotor UAVs. Master's thesis, Politecnico di Milano, 2015-2016.

[12] T. E. Gibson, A. M. Annaswamy, and E. Lavretsky. On adaptive control with closed-loop reference models: Transients, oscillations, and peaking. *IEEE Access*, 1:703–717, 2013.

[13] B. Peterson and K. Narendra. Bounded error adaptive control. *IEEE Transactions on Automatic Control*, 27(6):1161–1168, 1982.

[14] P.A. Ioannou and P.V. Kokotovic. Instability analysis and improvement of robustness of adaptive control. *Automatica*, 20(5):583–594, 1984.

[15] K. S. Narendra and A. M. Annaswamy. A new adaptive law for robust adaptation without persistent excitation. In *1986 American Control Conference*, pages 1067–1072, 1986.

Appendix A

Tilt-arm simulator data

In table A.1 the main physical parameters used in the tilt-arm simulator are reported. It has to be noted that the aerodynamic damping is a negative torque, which enters in the attitude dynamic, modelled as $c\omega$ where $c \in \mathbb{R}^3$ is the damping coefficient [1]. Moreover, table A.4 reports the values of the nominal inertia components compared to the plant ones. Instead, table A.3 and A.2 show respectively the initial conditions and the baseline controller gains used in the tilt-arm simulator.

Variable	**U.M.**	**Value**
Gravity	$[\frac{\text{m}}{\text{s}^2}]$	9.81
Air Density	$[\frac{\text{kg}}{\text{m}^3}]$	1.225
Bracket length	[m]	0.225
UAV mass	[kg]	1.523
Propeller diameter	[m]	0.3048
Thrust coefficient K_t		2.9×10^-6
Torque coefficient K_q		2.2×10^-8
$\sigma = \frac{K_q}{K_t}$		0.0076
Minimum β_{ai}	[deg]	63.7
Maximum β_{ai}	[deg]	-63.7
Minimum $Th\%$		23
Maximum $Th\%$		100
Roll damping c_1		-0.0463
Pitch damping c_2		-0.0463
Yaw damping c_3		-0.0185
x_{c1}	[m]	0.01
x_{c2}	[m]	0.01
x_{c3}	[m]	0.01

Table A.1: Main physical parameters used in the tilt-arm simulator

Gain	**Value**
Attitude dynamic	
K_{po}^{A}	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}$
K_{pi}^{A}	$\begin{bmatrix} 0.18 & 0 & 0 \\ 0 & 0.27 & 0 \\ 0 & 0 & 0.09 \end{bmatrix}$
K_{ff}^{A}	0_3
Position dynamic	
K_{po}^{P}	$\begin{bmatrix} 1.1 & 0 & 0 \\ 0 & 1.3 & 0 \\ 0 & 0 & 1.2 \end{bmatrix}$
K_{pi}^{P}	$\begin{bmatrix} 6 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 15 \end{bmatrix}$
K_{ff}^{P}	0_3

Table A.2: Baseline controller gains used in the tilt-arm simulator

The four servo-actuators have been modelled trough a generic state space model as the one given below:

$$\dot{x_b} = A_b x_b + B_b u_b \qquad y_b = C_b x_b + D_b u_b \tag{A.1}$$

where A_b and B_b are equals to:

$$A_b = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -4650.23 & -598.46 & -28.36 \end{bmatrix} \qquad B_b = \begin{bmatrix} 0 \\ 0 \\ 4670.25 \end{bmatrix} \tag{A.2}$$

while $C_b = I_3$ and $D_b = 0_{31}$. $u_b \in \mathbb{R}$ and $y_b \in \mathbb{R}$ are respectively the system input and output that, in our case, correspond to the bracket angles β_{ai}. Finally, the four brush-less motors have been modelled as a linear transfer function in the form of equation (A.3):

$$G(s) = \frac{1}{T_b s + 1} \tag{A.3}$$

where $T_b = 0.04$ is the motor time constant.

Variable	**U.M.**	**Value**
Position x_0	[m]	$[0, 0, 0]^T$
Velocity v_0	[m/s]	$[0, 0, 0]^T$
Angular rates ω_0	[deg/s]	$[0, 0, 0]^T$
Attitude x_0	[deg]	$[0, 0, 0]^T$
$\hat{\theta}_{a0}$		0_{91}
$\hat{d}_{T0}$	[Nm]	$[0, 0, 0]^T$
$\hat{\theta}_{p0}$		0_{51}

Table A.3: Initial conditions used in the tilt-arm numerical integrators

Nominal $\bar{J}$	**Value** [kg m^2]	**Plant** J	**Value** [kg m^2]
$\bar{J}_{11}$	0.0114	J_{11}	0.0098
$\bar{J}_{12}$	0	J_{12}	-1.523×10^{-4}
$\bar{J}_{13}$	0	J_{13}	-1.523×10^{-4}
$\bar{J}_{21}$	0	J_{21}	-1.523×10^{-4}
$\bar{J}_{22}$	0.0104	J_{22}	0.0090
$\bar{J}_{23}$	0	J_{23}	-1.523×10^{-4}
$\bar{J}_{31}$	0	J_{31}	-1.523×10^{-4}
$\bar{J}_{32}$	0	J_{32}	-1.523×10^{-4}
$\bar{J}_{33}$	0.0198	J_{33}	0.0168

Table A.4: Nominal and plant inertia components used in the tilt-arm simulator