Treball de Fi de Master

Master's degree in Automatic Control and Robotics

CONTROL OF AN UAV USING LPV TECHNIQUES

MEMÒRIA

Autor: Carlos Trapiello Fernández
Director/s: Dr. Vicenç Puig Cayuela
Dr. Bernardo Morcego Seix

Convocatòria: Juny 2018

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

Small quadcopters have demonstrated to be the perfect testing bench for addressing the autonomous flying problem. Unmanned Air Vehicles (UAVs) represent a very challenging research topic that requires from different disciplines such as electronics, computer vision, geolocalization, control and planning. The dynamics of a quadcopter presents six degrees of freedom: three translations and three angular movements, but only four control commands (the overall thrust force and three moments), leading to an underactuated control problem.

Thus, this project tackles the position and heading, yaw angle, control problem for a quadcopter, making use of a cascade control structure. This cascade controller is composed by an outer position controller which returns the needed thrust force and roll-pitch angles to achieve a desired position, and an inner attitude controller that tracks the previous roll-pitch angles plus an external yaw angle, inducing three possible moments in the quadrotor structure.

After a model derivation stage, the inner attitude controller was designed by means of a Linear Parameter-Varying (LPV) approach, such that minimizes the Linear Quadratic Regulator (LQR) problem, defined as a set of Linear Matrix Inequalities (LMIs) for its vertex systems. Furthermore, using the same technique an Unknown Input Observer (UIO) was designed for decoupling possible disturbances affecting the attitude subsystem. For the position control, a feedback linearization structure was used, analytically deriving the control actions. Also, an outer velocities controller was designed by means of feedback linearization.

All the proposed control variations were validated in simulation, including the response to disturbances of the vulnerable attitude subsystem. Besides, the system response with a velocities path planner was tested. Finally, the inner attitude controller was implemented in a real platform demonstrating its capabilities in flight conditions.

Acknowledgements

I would like to thank Prof. Vicenç Puig Cayuela for his expert advise and his involvement in the project. I would also like to thank Prof. Bernardo Morcego Seix for his inestimable help. Last but not least, I would like to thank the guys in the Advanced Control Systems Group for their help during the implementation stage.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The continuous advances in electronics during the last decades, entail a huge leap in microcomputers performance while the size and weight of the components decreases each day. As the electronic industry widens its range of applications into all the fields of human life and society, the price of reliable electronic components such as sensors, microprocessors, etc. is getting more and more affordable.

As a consequence of the aforementioned development in the electronics field, the drone platforms got out of the military sector, and started to get introduced in the commercial market during the last years. Soon a huge number of commercial applications of drone technology appeared, and the market of drone components as well as the drone services seems to be in a continuous expansion [1]. A clear example of the relevance that drone industry is gaining can be seen in the Eurocontrol agency project: CORUS [2], that aims to develop a concept of operations for managing drone in the European Very Low Level (VLL) airspace; or within the national territory with the latest decree approved by the government, which reviews in a more ambitious way the civil use of remote piloted aircrafts [3].

So far, the more generic term drone was used. Nevertheless, for most commercial applications nowadays, a most accurate term would be RPAS (Remote Piloted Aircraft system) always under the supervision of a certified human pilot. Leaving the completely autonomous systems still in the research stage. It is worthwhile to remark the wide open source community around drones, with a huge number of projects like Ardupilot [4], which provides open hardware and software that allows to test different control algorithms in home-made drones.

Although there is huge variety of drone platforms, multirotors are the ones that got greater impact in commercial applications due to its capacity to hover in a fixed point and fly in limited spaces. Multirotors or multicopters are rotorcrafts with more that two rotors, which, unlike helicopters, uses fixed-pitch blades and the control of the vehicle is achieved varying the relative speed of each motor modifying the thrust and torques produced by each one. Depending on the number of rotors a different name is given to the rotorcraft. The work developed in this thesis is focused in quadrotors or quadcopters.

1.1 Motivation

If it is desired that Unnamed Air Vehicles (UAVs) are integrated as a precision tool not only in the industry, but also in our social lives, a tighter and more reliable control systems must be developed. Until it cannot be proof the resilience of the systems, within strict margins, in all sort of conditions, autonomous UAVs should not be able to flight in real conditions were human lives are at risk.

So this Thesis tries to contribute to the integration of UAVs into our daily lives, designing and validating high performance control algorithms for small quadcopters. These small quadcopters are considered the perfect test-bed for developing proofs of concept when developing new systems. This final integration of UAVs seem to be one of the latest market trends [5], and a wide scope of possibilities will appear that could help the society like substituting potentially dangerous works, providing support to rescue teams, supplying first-aid kit, etc.

1.2 Thesis Objectives

The initial aim of this project was to develop a reliable quadrotor position controller. However, as the position control is achieved tilting the platform certain angles, the design of the attitude controller is also considered. Hence, the objective of this thesis can be defined as: the design of a quadrotor position-attitude controller, testing its performance in simulation and also in a real platform. In order to achieve that goal a set of sub-goals are defined as follows:

- To identify a six degrees of freedom dynamic model of a quadcopter.
- To design a control structure for the underactuated quadcopter system.
- To implement a disturbance rejection mechanism.

- To validate the proposed control algorithms with a more complex model in Simulation.
- To export the simulation results obtained in Matlab®, into Robotic Operating System (ROS) environment in order to implement the controller in a real platform.
- To test the controller design in real flight conditions.

1.3 Thesis Structure

The work is organized as follows:

Chapter 2

This chapter covers the current state of the art on quadrotors and the more relevant control techniques applied in the field.

Chapter 3

In this chapter, a generic quadrotor model is identified. It is divided it two parts, the first one where a description of the considered reference frames and the available control actions is performed. In the second part, a dynamic model is developed with respect to different reference frames.

Chapter 4

In this chapter, the proposed control structure is developed. It is split in three main parts: a first one were the simplifications made for control design are presented, a second one where the inner-loop attitude control (including a variant with disturbace rejection) is developed, and the third one where the position/velocity control is addressed.

Chapter 5

In this chapter, the aforementioned control schemes are validated in simulation using a more realistic model than the design one. Throughout this chapter, an explanation of the validation model differences, the parameter tuning for the target quadrotor, and several simulation tests, are developed.

Chapter 6

In this chapter, the accomplished real implementation stage is explained. Firstly, a detailed analysis of the target quadrotor main characteristics and operation is performed. Next, a description of the algorithm modifications in order to adequate it to the real platform, is detailed. Finally, the results obtained in the testing bench and in real flight, are shown.

Chapter 7

In this chapter, a brief study of the socioeconomic and environmental effects of getting this project off the ground, is carried out.

Chapter 8

In this chapter, it is performed a superficial analysis of the project cost, and the unitary production cost considering to sell the controller as a finished good.

Chapter 9

In this final chapter, the conclusions reached during the development of the project are presented, as well as the main points that could be improved in a future work.

Chapter 2

State of the art on quadrotor control

The state of the art in quadrotor control has suffered a drastic change in the last few years. The number of projects tackling this problem, in different research fields, has considerably increased. Most of these projects are based on commercially available toys like the Draganflyer [6], modified afterwards to have more sensory capabilities. Other researchers prefer instead, to build their own structure from zero as in the case of the X4-Flyer [7] or the mesicopter [8].

Within the quadrotor field of research, there are articles which analyse hybrid structures with non-symmetric rotation directions or with two directional rotors [9]. Other works focus instead on derivations of quadrotor mathematical models [10], or more efficient configurations [11].

Although there are a lot of different topics related with the quadrotor structure, the one that most publications has focused is the control problem. It can be stated that the 85% of the published articles propose a control law, or compare the performance of some of them. The most important techniques developed, and some related publications are presented below:

1. Lyapunov Theory [12]: According to this technique, it is possible to ensure, under certain conditions, the asymptotical stability of the quadcopter.

2. PD^2 feedback, and PID structures [13]: The strength of the PD^2 feedback lays on the exponential convergence that presents. On the other hand, PID structures does not require to determine specific model parameters and the control law is much simpler to implement.

3. Adaptative techniques [14]: These methods provide good performance against parametric uncertainties and unmodeled dynamics.

4. Linear Quadratic Regulator (LQR) [13]: The main advantage of this technique is that the

optimal input signal turns out to be obtainable from full state feedback. However, the analytical solution to the Ricatti equation is difficult to compute.

5. Backstepping control [15]: With this technique, the convergence of the quadrotor internal states is guaranteed, at the expense of a high computational cost.

6. Dynamic feedback [16]: This technique is implemented in a few quadrotor projects in order to transform the closed loop part of the system into a linear, controllable and decoupled subsystem.

7. Visual feedback: The camera used for this purpose can be mounted on-board [17] or off-board [18] (fixed to the ground).

8. Other control algorithms: fuzzy techniques [19], neural networks [20], reinforcement learning [21], as well as the cutting-edge techniques of bioinspired flight controllers [22].

The contribution of this thesis lies mainly in the following fields:

- Accurate and robust control structure
- Reliable simulator
- Proof of concept with a real platform

Chapter 3

Quadrotor model

When facing the quadrotor's model identification problem, most researches consider dynamic models, as the kinematic model of a quadcopter simply represents the movement of a six degrees of freedom (DOF) point in the space. Thus, the main differences in the literature regarding the model correspond with the set of dynamics modelled, or the reference frame considered. During the development of this project, effects like the motor dynamics, or the asymmetry of lift, were not considered. This thesis will make use of the most common quadrotor dynamic model in the research field.

This Chapter is organised as follows: In the first part, a description of the used reference frames and the available control commands is carried out, and in the second part the quadrotor dynamic model is developed in the usual body frame and in a specially chosen hybrid frame.

3.1 Preliminar notions

Quadrotors, also called quadcopters, are multirotor helicopters that are lifted and propelled by four motors. According to [23], a quadrotor is: a Vertical Take Off Landing (VTOL) aircraft having four vertically oriented propellers, and that can be tilted for movement while in flight. Before studying the mathematical model of a quadrotor, it is necessary to introduce the reference coordinate frames in which the variables are described. Here two possible standards can be followed:

1. Aeronautical reference frame standard: Z-axis pointing downwards

2. Robotics reference frame standard: Z-axis pointing upwards

As the development of this work was inside a robotics scope, besides the software used for

implementation comes from the robotics field (ROS), the robotic's reference frame standard was used throughout the project. According to that, the following reference frames are defined:

- **A Fixed Ground reference frame** (E-frame): East, North, Up (ENU). With the frames defined as tangent to the globe lines of coordinates as can be seen in Fig. 3.1.1, that is:
  - East-West tangent to parallels
  - North-South tangent to meridians
  - Up-Down in the direction to the center of the earth

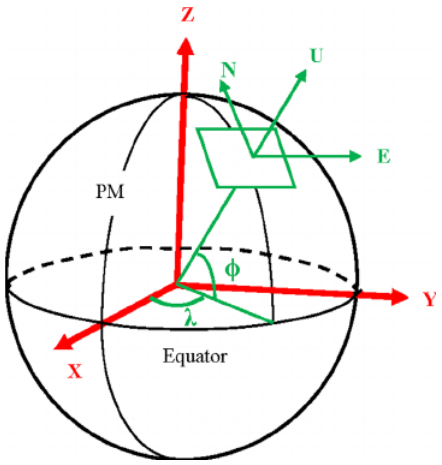


Figure 3.1.1: O_{ENU} fixed reference system

- **A body fixed frame** (B-frame): attached to the quadrotor. With its center fixed to the barycenter of the quadrotor, the X-axis points to the motor 1, the Z-axis pointing upwards and the Y-axis defining a right-handed coordinate system as can be seen in Fig. 3.1.2 . In the scientific literature this frame is called O_{ABC} system, where ABC stands for *Aircraft Body Center*.

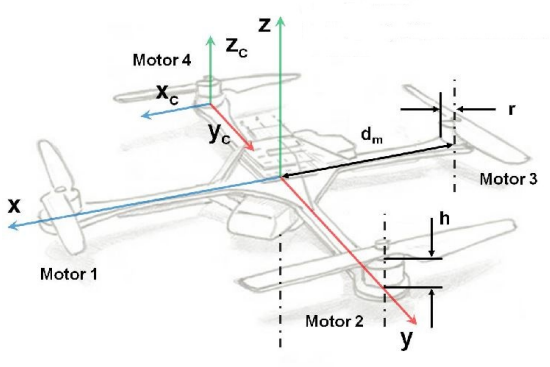


Figure 3.1.2: O_{ABC} mobile reference system

The attitude and position of the quadrotor can be controlled to desired values by modifying the speed of the four motors. The following forces and moments can act in the quadrotor: the thrust caused by rotors rotation, the pitching and rolling moment caused by the difference of the four rotors thrust, the gravity, the gyroscopic effect and the yawing effect. The yawing moment is caused by the unbalanced of the four motors rotational speeds. This moment can be cancelled out when two out of the four rotors rotate in the opposite direction, so the propellers are divided in two groups as follows:

- Front and rear propellers (motors 1 and 3 in Fig. 3.1.3), rotating counter-clockwise.
- Left and right propellers (motors 2 and 4 in Fig. 3.1.3), rotating clockwise.

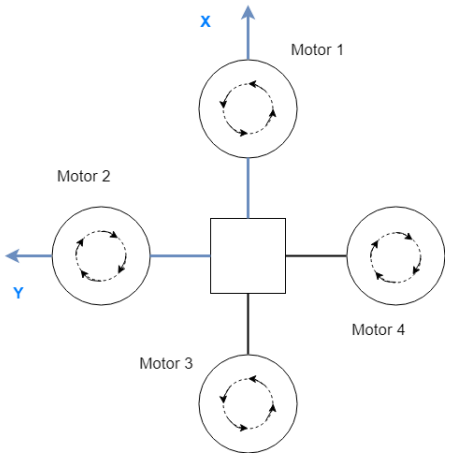


Figure 3.1.3: Direction of propellers rotation

The yaw motion of the quadrotor is generated by the reactive torque produced by each rotor. When the four rotor speeds are the same, the reactive torques will balance each other and the quadrotor will not rotate, whereas if the four rotor speeds are not the same, the reactive torques will not be balanced, and the quadrotor will start to rotate around its Z-axis.

The space motion of the rigid body quadrotor can be divided into two parts:

1. The barycenter movement: three barycenter movements that corresppond with the three translations. This movements defines the position of the quadrotor.
2. Movement around the barycenter: three angular motions that correspond with the three rotation motions along the axes. This movements defines the attitude of the quadrotor.

This leads to six different degrees of freedom, whose control can be implemented by adjusting the rotational speed of the different motors. The motions include forward and backward move-

ments, lateral movement, vertical motion, roll motion, pitch motion and yaw motion.

Depending on the rotation speed of each propeller, it is possible to identify the four basic movements which allow the quadcopter to reach a certain position and attitude:

- Throttle U_1 $[N]$

  This command is provided by increasing, or decreasing, all the propeller speeds by the same amount. It leads to a vertical force with respect to (WRT) the body-fixed frame. If the quadrotor is in the hover position, the vertical direction of both the inertial frame and the body frame coincide. Otherwise, the applied thrust provides both vertical and horizontal accelerations in the inertial frame (E-frame). Figure 3.1.4 shows how an increase $\Delta_A\,[rad/s]$ in all the motor's rotational speeds induces a positive throttle force.

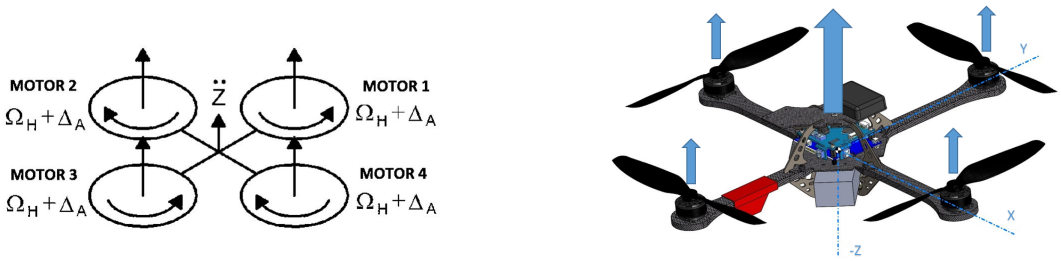


Figure 3.1.4: Throttle movement

- Roll U_2 $[N\,m]$

  This command is provided by increasing (or decreasing) the left propeller speed and by decreasing (or increasing) the right one. It leads to a torque with respect to the X-ais in body frame, which makes the quadrotor turn. Figure 3.1.5 shows the roll command that induces an increment in the roll angle (Δ_A increase in motor 2 and Δ_B decrease in motor 4). The variables Δ_A and Δ_B are chosen to maintain the vertical thrust unchanged. It can be demonstrated that for small values of Δ_A: $\Delta_B \approx \Delta_A$.

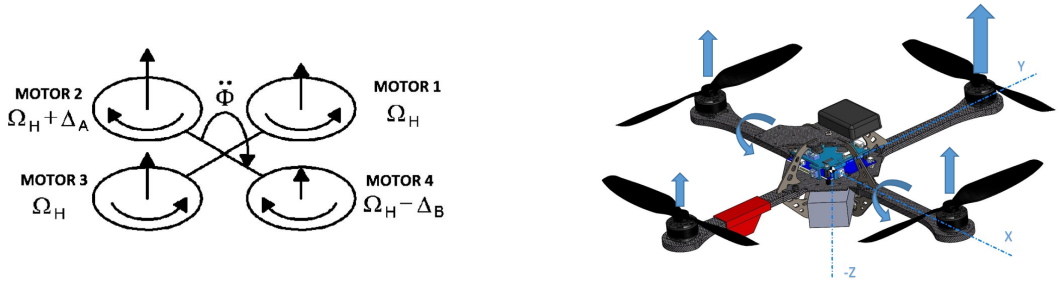


Figure 3.1.5: Roll movement

- Pitch U_3 $[N\,m]$

  This command is similar to the roll one and is provided by increasing (or decreasing) the rear propeller speed and by decreasing (or increasing) the front one. This lead to a torque with respect to the Y-axis in body frame which makes the quadrotor turn. Figure 3.1.5 shows the pitch command that induces an increase in the pitch angle (Δ_A increase in motor 3 and Δ_B decrease in motor 1). As in the previous case, Δ_A and Δ_B are chosen to maintain the vertical thrust unchanged, and for small values of Δ_A: $\Delta_B \approx \Delta_A$.

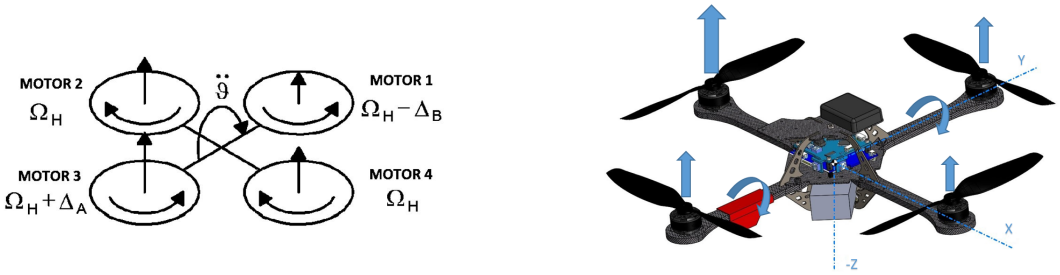


Figure 3.1.6: Pitch movement

- Yaw U_4 $[N\,m]$

  This command is provided by increasing (or decreasing) the front-rear propellers speed and by decreasing (or increasing) that of the left-right couple. It leads to a torque with respect the Z-axis in body frame which makes the quadrotor turn. The yaw movement is generated thanks to the fact that the left-right propellers rotate clockwise while the front-rear ones rotate counterclockwise. Hence, when the overall torque is unbalanced, the helicopter turns on itself around the Z-axis in body frame. Figure 3.1.7 shows the yaw command that induces an increase in the yaw angle. As in the previous cases Δ_A and Δ_B are chosen to maintain the vertical thrust unchanged, and for small values of Δ_A: $\Delta_B \approx \Delta_A$.

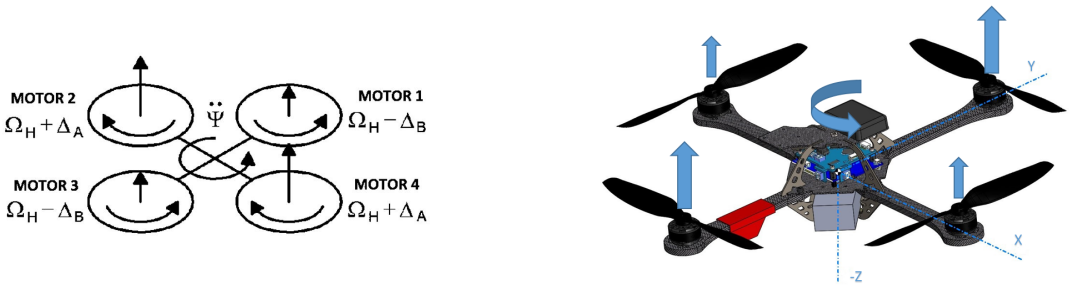


Figure 3.1.7: Yaw movement

The previous basic commands are the control actions available in order to control the quadrotor.

It is important to note that its direct relation with the commands of the remote controller for the commercial drones available nowadays (see Fig. 3.1.8). For the operation of the remote piloted quadrotors, a controller stabilizes the quadrotor over the hover position, and the user through the remote controller can vary the thrust, modifying the height, or induce one of the defined moments, tilting the drone and hence causing a translational displacement.

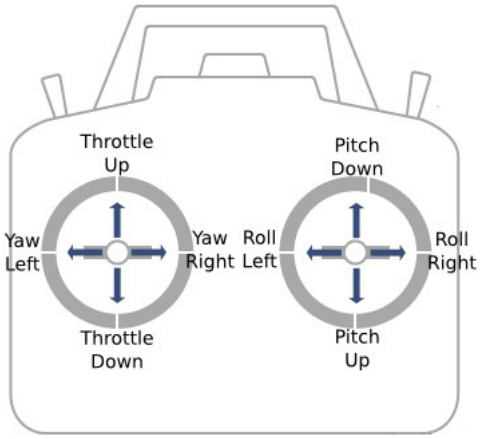


Figure 3.1.8: Remote controller commands

Due to the four inputs and six outputs in a quadrotor system, it is considered an underactuated non-linear complex system. For the controller design procedure several assumptions are made: the quadrotor is a rigid body, the structure is symmetric, the ground effect is ignored. In the following section a detailed description of the quadcopter mathematical model is carried out.

3.2 Quadrotor mathematical model

In this section, a specific description of the quadrotor model is developed based on the generic 6 DOF rigid-body equation derived with the Newton-Euler formalism (see [24] Appendix A).

Previously two different frames were defined:

- Earth inertial frame (E-frame)
- Body-fixed frame (B-frame)

The equations of motion are more conveniently formulated in the body-fixed frame due to the following reasons [25]:

- In B-frame, the inertia matrix is time-invariant.

- It can be taken advantage of the quadrotor symmetry.
- Measurements taken on-board are easily converted to B-frame.
- Control forces are almost always given in B-frame.

3.2.1 Kinematics

The kinematics of a 6 DOF rigid body are described by the following equation

$$\dot{\xi} = J_\Theta \, \nu \tag{3.2.1}$$

where $\dot{\xi}$ is the generalized velocity vector WRT E-frame, ν is the generalized velocity vector WRT B-frame and J_Θ is the generalized matrix.

The term ξ is a composition of the quadrotor linear position $\Gamma^E\,[m]$, and angular position $\Theta^E\,[rad]$ expressed in Euler angles following the convention: Z-Y'-X" ($\Theta^E = [\phi,\, \theta,\, \psi]^T$).

$$\xi = [\Gamma^E \; \Theta^E]^T = [x \; y \; z \; \phi \; \theta \; \psi]^T \tag{3.2.2}$$

Similarly, ν is composed of the quadrotor linear $V^B\,[m/s]$ and angular $\omega^B[rad/s]$ velocity vectors WRT B-frame:

$$\nu = [V^B \; \omega^B]^T = [u \; v \; w \; p \; q \; r]^T \tag{3.2.3}$$

The generalized matrix J_Θ is composed of 4 sub-matrices as shown in Eq. (3.2.4).

$$J_\Theta = \begin{bmatrix} R_\Theta & 0_{3\times3} \\ 0_{3\times3} & T_\Theta \end{bmatrix} \tag{3.2.4}$$

Being R_Θ the rotation matrix that relates the B-frame with the E-frame under the Z-Y'-X" Euler angles convention (see Fig. 3.2.1):

$$R_\Theta = \begin{bmatrix} \cos\theta\,\cos\psi & \sin\phi\,\sin\theta\,\cos\psi - \cos\phi\,\sin\psi & \cos\phi\,\sin\theta\,\cos\psi + \sin\phi\,\sin\psi \\ \cos\theta\,\sin\psi & \sin\phi\,\sin\theta\,\sin\psi + \cos\phi\,\cos\psi & cos\phi\,sin\theta\,sin\psi - sin\phi\,cos\psi \\ -\sin\theta & \sin\phi\,\cos\theta & \cos\phi\,\cos\theta \end{bmatrix} \tag{3.2.5}$$

The matrix T_Θ relates the Euler-angle rates $[\dot{\phi} \; \dot{\theta} \; \dot{\psi}]$, with the angular velocities expressed in B-frame $[p \; q \; r]$. For the Z-Y'-X" Euler angles convention:

- $\dot{\psi}$ is measured in the Inertial frame

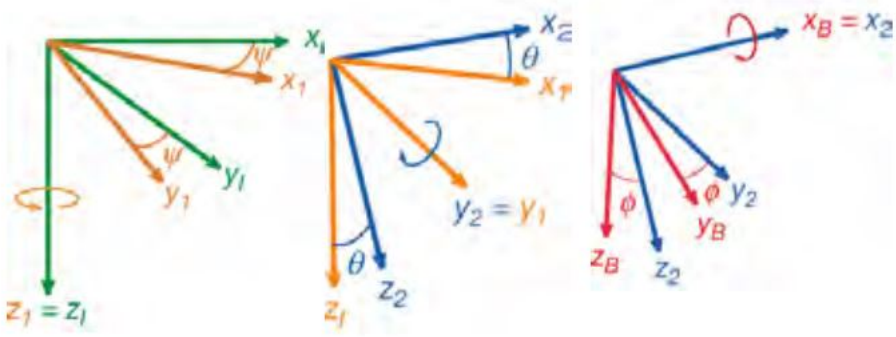


Figure 3.2.1: Euler-angles convention

- $\dot{\theta}$ is measured in the first intermediate frame
- $\dot{\phi}$ is measured in the second intermediate frame

This mathematical relationship is expressed in Eq. (3.2.6).

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = I_3 \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + H_2^B \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + H_2^B H_1^2 \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3.2.6}$$

The inverse transformation of Eq. (3.2.6), returns the T_Θ matrix:

$$T_\Theta = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \tag{3.2.7}$$

3.2.2 Dynamics

3.2.2.1 Body frame (B-frame)

The dynamics of a generic 6 DOF rigid-body takes into account the mass of the body $m\,[kg]$ and its inertia matrix $I\,[N\,m\,s^2]$. A formal derivation of the inertia moments for the simple elements in which the complex quadrotor structure can be decomposed is developed in [24] (see Appendix D). Nevertheless, as explained in Chapter 6, the actual computation of the whole quadrotor inertia moments nowadays, in not done by hand, but by means of models developed in specific design software like SolidWorks ® or CATIA ®.

The dynamics of a 6 DOF rigid body are described by Eq. (3.2.8).

$$\begin{bmatrix} m\,I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I \end{bmatrix} \begin{bmatrix} \dot{V}^B \\ \dot{\omega}^B \end{bmatrix} + \begin{bmatrix} \omega^B \times (m\,V^B) \\ \omega^B \times (I\,\omega^B) \end{bmatrix} = \begin{bmatrix} F^B \\ \tau^B \end{bmatrix} \tag{3.2.8}$$

where $\dot{V}^B$ $[m/s^2]$ is the quadrotor linear acceleration vector WRT B-frame, and $\dot{\omega}^B$ $[rad/s^2]$ is the quadrotor angular acceleration vector also WRT B-frame. Furthermore, F^B $[N]$ is the quadrotor forces vector WRT B-frame and τ^B $[N\,m]$ is the quadrotor torques vector WRT B-frame.

Two assumptions have been considered in order to simplify the dynamics mathematical model development:

- The origin of the body frame (B-frame) is coincident with the center of mass (COM) of the body.
- The axes of the B-frame (see Fig. 3.1.2) coincide with the body principal axes of inertia. In this case, the inertia matrix I is diagonal and the body equations become easier.

If a generalized force vector Λ is defined as follows:

$$\Lambda = [F^B\ \tau^B]^T = [F_x\ F_y\ F_z\ \tau_x\ \tau_y\ \tau_z]^T \tag{3.2.9}$$

then it is possible to rewrite Eq. (3.2.8) in a simpler matrix form:

$$M_B\dot{\nu} + C_B(\nu)\nu = \Lambda \tag{3.2.10}$$

where $\dot{\nu}$ is the generalized acceleration vector WRT B-frame. M_B is the system inertia matrix and $C_B(\nu)$ is the Coriolis-centripetal matrix, both WRT B-frame. It is worthwhile to remark that Eq. (3.2.10) is valid for all the rigid bodies that obey the simplifications previously made.

Thanks to the assumptions aforementioned, the system inertia matrix M_B is diagonal and constant:

$$M_B = \begin{bmatrix} m\,I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{xx} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{yy} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{zz} \end{bmatrix} \tag{3.2.11}$$

Eq. (3.2.12) shows the Coriolis-centripetal matrix:

$$C_B(\nu) = \begin{bmatrix} 0_{3\times3} & -m\,S(V^B) \\ 0_{3\times3} & -S(I\,\omega^B) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & m\,w & -m\,v \\ 0 & 0 & 0 & -m\,w & 0 & m\,u \\ 0 & 0 & 0 & m\,v & -m\,u & 0 \\ 0 & 0 & 0 & 0 & I_{zz}\,r & -I_{yy}\,q \\ 0 & 0 & 0 & -I_{zz}\,r & 0 & I_{xx}\,p \\ 0 & 0 & 0 & I_{yy}\,q & -I_{xx}\,p & 0 \end{bmatrix} \tag{3.2.12}$$

where $S(\cdot)$ refers to the skew-symmetic operator. For generic three dimension vectors, as the ones used, the skew-symmetric matrix $S(k)$ is defined:

$$S(k) = -S^T(k) = \begin{bmatrix} 0 & -k_3 & k_1 \\ k_3 & 0 & -k_1 \\ -k_2 & k_1 & 0 \end{bmatrix} \qquad k = \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \tag{3.2.13}$$

The aforementioned generalized force vector Λ can be divided in three components according to the nature of the quadrotor contributions:

1. Gravitational vector $G_B(\xi)$: given from the acceleration due to gravity g $[m/s^2]$. As it is modelized as a force applied in the COM, it only affects the linear equations. $G_B(\xi)$ in B-frame is:

$$G_B(\xi) = \begin{bmatrix} F_G^B \\ 0_{3\times1} \end{bmatrix} = \begin{bmatrix} R_\Theta^{-1}\,F_G^E \\ 0_{3\times1} \end{bmatrix} = \begin{bmatrix} R_\Theta^T \begin{bmatrix} 0 \\ 0 \\ -m\,g \end{bmatrix} \\ 0_{3\times1} \end{bmatrix} = \begin{bmatrix} m\,g\,\cos\theta \\ -m\,g\,\cos\theta\,\sin\phi \\ -m\,g\,\cos\theta\,\cos\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.2.14}$$

   where $F_G^B\,[N]$ is the graviational force WRT B-frame and $F_G^E\,[N]$ is WRT E-frame.

2. Gyroscopic effects $O_B(\nu)$: The second contribution takes into account the gyroscopic effects produced by the propeller rotation. Since two of them are rotating clockwise and the other two counterclockwise, there is an overall imbalance when the algebraic sum of the rotor speeds is not equal to zero. If, in addition, the roll or pitch rates are also different than zero, the quadrotor experiences a gyroscopic torque according to:

$$O_B(\nu) = \begin{bmatrix} 0_{3\times1} \\ -\sum_{k=1}^{4} J_{TP} \left(\omega^B \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) (-1)^k \Omega_k \end{bmatrix} = \begin{bmatrix} 0_{3\times1} \\ J_{TP} \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \Omega \end{bmatrix} =$$

$$= J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{\Omega} \tag{3.2.15}$$

where $J_{TP}\,[N\,m\,s^2]$ is the total rotational moment of inertia around the propeller axis. It is obvious that the gyroscopic effects produced by the propeller rotation are just related to the angular and not to the linear equations.

In Eq. (3.2.15) the overall propellers speed $\Omega\,[rad/s]$ and the propellers speed vector $\mathbf{\Omega}\,[rad/s]$, are defined as follows:

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \qquad \mathbf{\Omega} = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \tag{3.2.16}$$

where $[\Omega_1,\, \Omega_2,\, \Omega_3,\, \Omega_4]$ correspond to the angular velocity in $[rad/s]$ of motor 1, motor 2, motor 3 and motor 4, respectively.

3. Main movement inputs $U_B(\Omega)$: The third contribution takes into account the forces and torques directly produced by the main movement inputs (see Section 3.1). From aerodynamics considerations, it follows that both forces and torques are proportional to the squared propellers speed. Therefore, the movement matrix E_B is multiplied by Ω^2 to get the movement vector $U_B(\Omega)$.

Eq. (3.2.17) shows the relation between the considered control commands (U_1, U_2, U_3, U_4) and the motor speeds. The aerodynamic terms c_T and c_Q can be analytically derived [23], although in practice identification tests are carried out for estimating their values. Thanks to this relation the required motor speeds for controlling the quadrotor can be obtained.

$$U_B(\Omega) = E_B\,\Omega^2 = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ c_T\,(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ c_T\,l\,(\Omega_4^2 - \Omega_2^2) \\ c_T\,l\,(\Omega_3^2 - \Omega_1^2) \\ c_Q\,(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{bmatrix} \tag{3.2.17}$$

where $l\,[m]$ is the distance between the center of the quadrotor and the center of a propeller.

According to the aforementioned, it is possible to identify a constant matrix E_B which multiplied by the squared propellers speed Ω^2, produces the movement vector $U_B(\Omega)$. The resulting E_B matrix is:

$$E_B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ c_T & c_T & c_T & c_T \\ 0 & -c_T\,l & 0 & c_T\,l \\ -c_T\,l & 0 & c_T\,l & 0 \\ -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \tag{3.2.18}$$

Taking into consideration previously described forces that conform the generalized force vector Λ, the quadrotor dynamic equation (Eq. (3.2.10)) can be transformed into:

$$M_B\dot{\nu} + C_B(\nu)\nu = G_B(\xi) + O_B(\nu)\,\Omega + E_B\,\Omega^2 \tag{3.2.19}$$

Rearranging Eq. (3.2.19), it is possible to isolate the derivate of the generalized velocity vector WRT B-frame $\dot{\nu}$ as follows:

$$\dot{\nu} = M_B^{-1}\,(-C_B(\nu)\nu + G_B(\xi) + O_B(\nu)\,\Omega + E_B\,\Omega^2) \tag{3.2.20}$$

Expressing the previous Eq. (3.2.20) as a system of equations, conforms the mathematical model

of a quadrotor in the B-frame:

$$\dot{u} = (vr - wq) + g\sin\theta \tag{3.2.21}$$

$$\dot{v} = (wp - ur) - g\cos\theta\sin\phi \tag{3.2.22}$$

$$\dot{w} = (uq - vp) - g\cos\theta\cos\phi + \frac{U_1}{m} \tag{3.2.23}$$

$$\dot{p} = qr\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}q\Omega + \frac{U_2}{I_x} \tag{3.2.24}$$

$$\dot{q} = pr\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}p\Omega + \frac{U_3}{I_y} \tag{3.2.25}$$

$$\dot{r} = pq\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} \tag{3.2.26}$$

where the propellers speed inputs are given through the expression:

$$U_1 = c_T\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \tag{3.2.27}$$

$$U_2 = c_T\, l\left(\Omega_4^2 - \Omega_2^2\right) \tag{3.2.28}$$

$$U_3 = c_T\, l\left(\Omega_3^2 - \Omega_1^2\right) \tag{3.2.29}$$

$$U_4 = c_Q\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \tag{3.2.30}$$

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{3.2.31}$$

3.2.2.2 Hyrbrid frame (H-frame)

The quadrotor dynamic system shown in Eqs. (3.2.22) to (3.2.26) is derived for a body fixed frame (B-frame). Nevertheless, when it comes to position control, specially height control, it is more useful to express it WRT E-frame. Due to that a new "hybrid" frame called H-frame is used such that: the linear equations are expressed in E-frame and the angular equations in the B-frame. Eq. (3.2.32) shows the quadrotor generalized velocity vector WRT the new H-frame

$$\zeta = [\dot{\Gamma}^E\, \omega^B]^T = [\dot{x}\ \dot{y}\ \dot{z}\ p\ q\ r]^T \tag{3.2.32}$$

The dynamics equation in matrix form WRT the H-frame is rewritten as follows

$$M_H\dot{\zeta} + C_H(\zeta)\zeta = G_H(\zeta) + O_H(\zeta)\,\Omega + E_H(\xi)\,\Omega^2 \tag{3.2.33}$$

Below, the matrices that conform the dynamics model in Eq. (3.2.33) are rewritten WRT the H-frame:

- The system inertia matrix WRT the H-frame M_H is exactly the same as expressed WRT

the B-frame:

$$M_H = M_B = \begin{bmatrix} m\,I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_x & 0 & 0 \\ 0 & 0 & 0 & 0 & I_y & 0 \\ 0 & 0 & 0 & 0 & 0 & I_z \end{bmatrix} \tag{3.2.34}$$

- The Coriolis-centripetal matrix WRT the H-frame $C_H(\zeta)$ is different than the one expressed WRT B-frame, and it is defined as follows:

$$C_H(\zeta) = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -S(I\,\omega^B) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_z\,r & -I_y\,q \\ 0 & 0 & 0 & -I_z\,r & 0 & I_x\,p \\ 0 & 0 & 0 & I_y\,q & -I_x\,p & 0 \end{bmatrix} \tag{3.2.35}$$

- The gravitational vector WRT H-frame G_H is much simpler, as now the linear Z-axis corresponds to the Z-axis E-frame:

$$G_H(\zeta) = \begin{bmatrix} F_G^E \\ 0_{3\times1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -m\,g \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.2.36}$$

- The gyroscopic effect produced by the propeller rotation is unvaried as it only affects the angular equations that remain referred to the B-frame. Then, the gyroscopic propeller matrix WRT H-frame $O_H(\zeta)$ is:

$$O_H(\zeta)\mathbf{\Omega} = O_B(\nu)\mathbf{\Omega} = \begin{bmatrix} 0_{3\times1} \\ J_{TP} \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \Omega \end{bmatrix} = J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{\Omega} \tag{3.2.37}$$

- The movement matrix WRT the H-frame $E_H(\xi)$ is different from the one expressed in B-frame, as now the input U_1 affects all the three linear equations through the rotation matrix R_Θ (just the opposite than with the gravitational vector $G_H(\zeta)$).

$$E_H(\xi)\Omega^2 = \begin{bmatrix} R_\Theta & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} E_B\Omega^2 = \begin{bmatrix} (\cos\phi\ \sin\theta\ \cos\psi + \sin\phi\ \sin\psi)\, U_1 \\ (cos\phi\, sin\theta\, sin\psi - sin\phi\, cos\psi)\, U_1 \\ (\cos\phi\ \cos\theta)\, U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{3.2.38}$$

Rearranging Eq. (3.2.33), the derivate of the generalized velocity vector WRT H-frame is obtained as follows:

$$\dot{\zeta} = M_H^{-1}\left(-C_H(\zeta)\zeta + G_H + O_H(\zeta)\,\Omega + E_H\,\Omega^2\right) \tag{3.2.39}$$

Expressing the aforementioned expression as a system of equations:

$$\ddot{x} = (cos\phi sin\theta cos\psi + sin\phi sin\psi)\frac{U_1}{m} \tag{3.2.40}$$

$$\ddot{y} = (cos\phi sin\theta sin\psi - sin\phi cos\psi)\frac{U_1}{m} \tag{3.2.41}$$

$$\ddot{z} = -g + cos\phi cos\theta\frac{U_1}{m} \tag{3.2.42}$$

$$\dot{p} = qr\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}q\Omega + \frac{U_2}{I_x} \tag{3.2.43}$$

$$\dot{q} = pr\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}p\Omega + \frac{U_3}{I_y} \tag{3.2.44}$$

$$\dot{r} = pq\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} \tag{3.2.45}$$

where the system inputs (U_1, U_2, U_3, U_4) are related with the motor speeds in the same manner than in the system WRT B-frame, as expressed in Eqs. (3.2.27) to (3.2.30).

Chapter 4

Control strategies

In this chapter, a cascade control scheme is presented in order to deal with the underactuated quadrotor system. Through the cascade structure the translational and rotational dynamics are decoupled, facing two non-linear problems in different loops. In the inner attitude loop, the angular dynamics are stabilized using the Linear Parameter-Varying (LPV) approach. The basic idea of the LPV technique is to embed the system non-linearities in some time variant parameters, allowing to face the non-linear problem as an affine composition of linear systems. To deal with possible disturbances, an LPV Unknown Input Observer is also designed for disturbance rejection in the attitude subsystem.

For the outer position loop, the translational dynamics are controlled by means of the feedback linearization approach. This method consists in the transformation of the non-linear system into an equivalent linear system through a change of variables an a suitable control input. Also a velocities controller is designed for using instead of the position one, if required.

This Chapter is divided in three parts: in the first one the simplifications made in the mathematical model are presented, in the second one the design of the LPV attitude controller is developed, and in the third one the design of the position/velocities control is addressed.

4.1 Model for control design

In Chapter 3, a quadrotor mathematical model was developed WRT B-frame and WRT a hybrid H-frame. As one of the goals of this chapter is the design of a control algorithm that allows to control the platform position in E-frame, the H-frame model will be used. However, for the sake of simplicity in the control design stage some simplifications are made over the obtained model.

The H-frame model obtained in the previous Section is written as:

$$\ddot{x} = (cos\phi sin\theta cos\psi + sin\phi sin\psi)\frac{U_1}{m} \tag{4.1.1}$$

$$\ddot{y} = (cos\phi sin\theta sin\psi - sin\phi cos\psi)\frac{U_1}{m} \tag{4.1.2}$$

$$\ddot{z} = -g + cos\phi cos\theta\frac{U_1}{m} \tag{4.1.3}$$

$$\dot{p} = qr\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}q\Omega + \frac{U_2}{I_x} \tag{4.1.4}$$

$$\dot{q} = pr\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}p\Omega + \frac{U_3}{I_y} \tag{4.1.5}$$

$$\dot{r} = pq\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} \tag{4.1.6}$$

where $[p\ q\ r]$ are the angular rates in B-frame:

$$\omega_B = \begin{bmatrix}\omega_x \\ \omega_y \\ \omega_z\end{bmatrix}_B = \begin{bmatrix}p \\ q \\ r\end{bmatrix} \tag{4.1.7}$$

While the translational equations (Eqs. (4.1.1) to (4.1.3)) are second order differential equations, that relate forces with positions, the rotational equations (Eqs. (4.1.4) to (4.1.6)) are just first order differential equations, that relate torques with angular velocities. So, in order to extend the rotational equations to get Euler-angles, the set of first order differential equations that relate the angular velocities in B-frame (p, q, r) and the Euler-angles rates $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ is needed. Those equations are derived in Section 3.2 through Eqs. (3.2.6) to (3.2.7) resulting in the following expression

$$\begin{bmatrix}\dot{\phi} \\ \dot{\theta} \\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta\end{bmatrix}\begin{bmatrix}p \\ q \\ r\end{bmatrix} = T(\Theta)_B^E\omega_B \tag{4.1.8}$$

This set of equations increases the complexity of the considered quadcopter model Eqs. (4.1.1) to (4.1.6). However, it is worthwhile to note, that close to the hover position ($\phi = 0$, $\theta = 0$) the matrix $T(\Theta)$ becomes the identity, $T(\Theta) = I$. Assuming that the aim is to design a controller that stabilizes the quadcopter close to the hovering position, then the quadrotor model can be simplified as expressed in Eqs. (4.1.9) to (4.1.14). Actually, this is not an important simplification if the main objective is controlling the quadrotor position over a particular location, as the maintenance of that position can only be achieved in hover, and meanwhile not high translational

speed are required, the tilt of the quadrotor (roll ϕ, pitch θ) for achieving a desired position will not result in high values

$$\ddot{x} = (cos\phi sin\theta cos\psi + sin\phi sin\psi)\frac{U_1}{m} \tag{4.1.9}$$

$$\ddot{y} = (cos\phi sin\theta sin\psi - sin\phi cos\psi)\frac{U_1}{m} \tag{4.1.10}$$

$$\ddot{z} = -g + cos\phi cos\theta\frac{U_1}{m} \tag{4.1.11}$$

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}\dot{\theta}\Omega + \frac{U_2}{I_x} \tag{4.1.12}$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}\dot{\phi}\Omega + \frac{U_3}{I_y} \tag{4.1.13}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} \tag{4.1.14}$$

where control actions (U_1, U_2, U_3, U_4) are related with the motor rotation speeds $(\Omega_1, \Omega_2, \Omega_3, \Omega_4)$ as explained in Section 3.2.2

$$U_1 = c_T\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \tag{4.1.15}$$

$$U_2 = c_T \cdot l\left(\Omega_4^2 - \Omega_2^2\right) \tag{4.1.16}$$

$$U_3 = c_T \cdot l\left(\Omega_3^2 - \Omega_1^2\right) \tag{4.1.17}$$

$$U_4 = c_Q\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \tag{4.1.18}$$

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{4.1.19}$$

It is important to note that in the system described by Eqs. (4.1.9) to (4.1.14), the angles and their time derivatives do not depend on translation components. However, on the other hand, translations depend on the angles. Following that fact, the whole system depicted in Eqs. (4.1.9) to (4.1.14) can be understood as constituted by two subsystems:

1. Force subsystem which comprises the translational equations, usually called translation subsystem.

2. Moment subsystem which comprises the angular equations, referred as the angle subsystem in the literature.

in such a way that the translation subsystem depends on the angle subsystem, but the angle subsystem is independent from the translation one. This particular dependence is represented in Fig. 4.1.1.

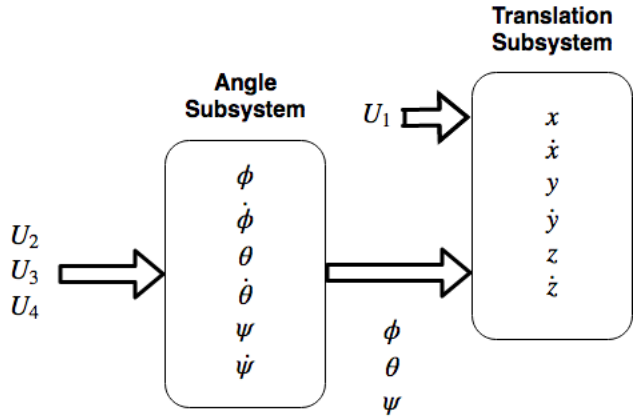


Figure 4.1.1: Quadrotor system structure

Aiming to deal with the control problem of the two subsystems, with the attitude system independent of the position one, a cascade control scheme is proposed. An inner control loop with fast dynamics will be in charge of the attitude control, tracking reference angles $(\phi_{ref}, \theta_{ref}, \psi_{ref})$. An outer control loop, with slower dynamics, assumes that the (ϕ, θ, ψ) angles are perfectly tracked and stabilizes the position $(x,\ y,\ z)$ of the quadrotor computing the required inputs: U_1 and (ϕ, θ) which together with (ψ_{ref}) are the reference for the inner control loop. The proposed control scheme for controlling position and yaw angle (ψ_{ref}) is shown in Fig. 4.1.2.

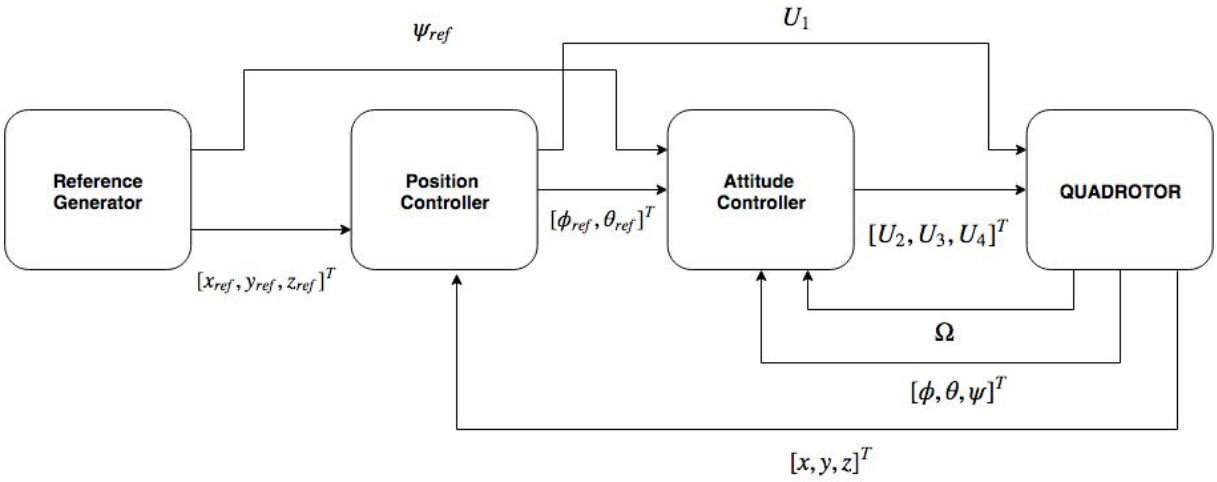


Figure 4.1.2: Proposed Cascade Control scheme

It is worthwhile to note that the simulation of the quadrotor's dynamics follows the opposite order than the controllers. That is: first the attitude control actions (U_2, U_3, U_4) determine the changes in attitude of the quadrotor (ϕ, θ, ψ), and with the new attitude and the overall thrust

force (U_1) the changes in position are computed (see Fig. 4.1.1).

4.2 Attitude Control

As stated in the previous Section, the attitude subsystem model is:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}\dot{\theta}\Omega + \frac{U_2}{I_x} \tag{4.2.1}$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}\dot{\phi}\Omega + \frac{U_3}{I_y} \tag{4.2.2}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} \tag{4.2.3}$$

where U_2, U_3, U_4 are the control actions in $N\,m$.

4.2.1 Quasi-LPV representation of the Attitude system

Naming [ϕ, $\dot{\phi}$, θ, $\dot{\theta}$, ψ, $\dot{\psi}$] as [x_1, x_2, x_3, x_4, x_5, x_6], the non-linear model Eqs. (4.2.1) to (4.2.3) can be expressed in the quasi-LPV absolute form, following the non-linear embedding approach, as shown in Eq. (4.2.4)

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{\Gamma_3(t)\cdot J}{I_x} & 0 & \Gamma_2(t)\cdot\frac{I_y - I_z}{I_x} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{\Gamma_3(t)\cdot J}{I_y} & 0 & 0 & 0 & \Gamma_1(t)\cdot\frac{I_z - I_x}{I_y} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & \frac{\Gamma_2(t)}{2}\cdot\frac{I_x - I_y}{I_z} & 0 & \frac{\Gamma_1(t)}{2}\cdot\frac{I_x - I_y}{I_z} & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{I_x} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{I_z} \end{bmatrix} \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{4.2.4}$$

where $\Gamma(t) = [\Gamma_1(t), \Gamma_2(t), \Gamma_3(t)]$ is the vector of varying parameters. This corresponds to a simple case where the varying parameters $\Gamma(t)$ are scheduled through scheduling variables $p(t)$, that in this case: $\Gamma(t) = p(t) = [x_2, x_4, \Omega] = [\dot{\phi}, \dot{\theta}, \Omega]$.

Starting with the basic assumption that measurements of Γ are available in real-time and the range of the elements of Γ is known a priori such that:

$$\Gamma_{jl} \leq \Gamma_j \leq \Gamma_{ju} \tag{4.2.5}$$

where Γ_{jl} and Γ_{jl} are lower and upper bounds of each element.

Then, vectors ω_i can be formed by taking each possible permutation of upper and lower bounds of the elements in Γ. There will be then $N = 2^{n_\Gamma}$ vectors ω_i such that the parameter vector $\Gamma(t)$ can be expressed into polytopic form:

$$\Gamma \in Co\ \{\omega_1, \omega_2, \cdots, \omega_N\} := \{\sum_{i=1}^{N} \pi_i \omega_i : \pi_i \geq 0, \sum_{i=1}^{N} \pi_i = 1\} \tag{4.2.6}$$

meaning that the vector Γ belongs to the convex hull formed by the vertices ω_i.

This modeling approach is referred to as *bounding box method*, because the convex hull generated by such approach has the shape of a rectangular bounding box, that is, a hyperrectangle.

If an affine function $\delta(\Gamma)$ is applied to the vector Γ, the result can be expressed in a polytopic form:

$$\delta(\Gamma) \in Co\ \{\delta(\omega_1), \delta(\omega_2), \cdots, \delta(\omega_N)\} := \{\sum_{i=1}^{N} \pi_i \delta(\omega_i) : \pi_i \geq 0, \sum_{i=1}^{N} \pi_i = 1\} \tag{4.2.7}$$

So, for the LPV system generated in Eq. (4.2.4), as the matrix $A(\Gamma)$ depends affinely on Γ, it will range in a polytope of matrices whose vertices are the images of the vertices in ω_i:

$$A(\Gamma(k)) \in Co\ \{A_i,\ i = 1, \cdots, N\} := \sum_{i=1}^{N} \pi_i(\Gamma(k)) A_i \tag{4.2.8}$$

with $\pi_i(\Gamma(k)) \geq 0$ and $\sum_{i=1}^{N} \pi_i(\Gamma(k)) = 1$, where each i-th model is called a vertex system. Due to that property this kind of LPV systems are referred to as *polytopic*.

Each element of the vector $p(t) = [x_2,\ x_4,\ \Omega] = [\dot{\phi},\ \dot{\theta},\ \Omega]$ is assumed to take values in an interval known a priori. The selected intervals used to address the attitude control of the quadrotor are presented in Section 5.5.1.1.

4.2.2 Attitude controller design

The continuous state-space representation of the LPV system presented in Section 4.2.1 has the form expressed below:

$$\dot{x}(t) = A(\Gamma(t))x(t) + Bu(t) \tag{4.2.9}$$

$$y(t) = Cx(t) + Du(t) \tag{4.2.10}$$

where $A(\Gamma(t))$ and B are described in Eq. (4.2.4). For the controller design stage, it is assumed that only the Euler angles can be obtained from the Inertial Measurement Unit (IMU). Although this is not necessarily true, and most IMUs provide measurements of the angular velocities in B-frame, is always a good decision to filter the data obtained with a proper state estimator. According with the previously stated, matrix C is assumed to have the following form:

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.2.11}$$

The quasi-LPV attitude subsystem is controlled by a state-feedback controller with reference tracking. This control law can be expressed as shown in Eq. (4.2.12) as follows

$$u(t) = K(\Gamma(t))\hat{x}(t) + v(t) \tag{4.2.12}$$

with $v(t) = F(t)r(t)$ being $r(t)$ the reference that it is desired to track. For the attitude subsystem the reference is such that: $r(t) = [\phi_{ref}, \theta_{ref}, \psi_{ref}]$.

The matrix $K(\Gamma(t)) \in \mathbb{R}^{n_u \times n_x}$ is the gain of the LPV controller that is scheduled according to:

$$K(\Gamma(t)) = \sum_{i=1}^{2^{n_\Gamma}} \pi_i(\Gamma(k))K_i \tag{4.2.13}$$

where K_i are the controller gains computed for the extremes of LPV system, that is for each possible permuation of upper and lower bound of the elements in Γ.

Note that the estimated state $\hat{x}(t)$ is used because $x(t)$ it is assumed not to be available. Con-

sequently, an LPV state observer is used to provide such state estimation:

$$\dot{\hat{x}}(t) = A(\Gamma(t))\hat{x}(t) + Bu(t) + L(\Gamma(t))(y - \hat{y}) \tag{4.2.14}$$

$$\hat{y}(t) = C\hat{x}(t) \tag{4.2.15}$$

where $\hat{x}(t) \in \mathbb{R}^{n_x}$ and $\hat{y}(t) \in \mathbb{R}^{n_y}$ are the estimated state and output respectively. The matrix $L(\Gamma(t)) \in \mathbb{R}^{n_x \times n_y}$ is the gain of the LPV state observer, and is given by:

$$L(\Gamma) = \sum_{i=1}^{N} \pi_i(\Gamma(k))L_i \tag{4.2.16}$$

LQR optimization

References like [26] and [27] address the study of the LPV stability criteria under the Linear Matrix Inequality (LMI) based formulation. A LMI is a convex constraint. Thus, optimization problems with convex objective functions and LMI constraints are relatively efficiently solvable. Various constraints from the control theory such as Lyapunov and Riccati inequalities can all be written as LMIs [28].

The approach followed in order to design the LPV controller matrix $K(\Gamma(t))$ and the LPV state observer matrix $L(\Gamma(t))$, is such that the Linear Quadratic Regulation (LQR) problem is minimized. This can be done designing the vertex controller/observer gains described in the following

4.2.2.1 Observer design LQR-LMIs

In [29] is demonstrated how to derive a set of LMIs that provides an optimal design for the extreme observer gains in Eq. (4.2.16), based on the Riccati equations of the Kalman filter.

For an observer tuning parameters $Q = Q^T \geq 0, R = R^T > 0$, the optimal performance bound $\gamma \geq 0$, the decay rate $\lambda \geq 0$, the output matrix C in Eq. (4.2.11) and the matrices A_i in Eq. (4.2.4). Then, the polytopic observer gains in Eq. (4.2.16) are obtained by finding Y and W_i satisfying the following LMIs:

$$\begin{bmatrix} YA_i + A_i^T Y - W_i C - C^T W_i^T + Y2\lambda & Y(Q^{\frac{1}{2}})^T & W_i \\ Q^{\frac{1}{2}} Y & -I & 0 \\ W_i^T & 0 & -R^{-1} \end{bmatrix} \leq 0 \tag{4.2.17}$$

$$\begin{bmatrix} \gamma I & I \\ I & Y \end{bmatrix} \geq 0 \tag{4.2.18}$$

considering $Y = Y^T > 0$ and applying the transformation $L_i = Y^{-1} W_i$.

4.2.2.2 Controller design LQR-LMIs

In [30] it is demonstrated how the LQR problem for a linear system, can be re-formulated into an H_2 performance problem, and hence can be solved via LMI techniques, with good numerical reliability.

According to the aforementioned reference: given the LQR parameters $Q = Q^T \geq 0, R = R^T > 0$, the optimal performance bound γ (such that $J(x,u) < \gamma$), the imposed decay rate η, and the matrices A_i obtained in Eq. (4.2.8). Then, the polytopic control gains in Section 4.2.2 are obtained by finding $P \in \mathbb{S}^{n_x}, W_i \in \mathbb{S}^{n_u \times n_x}\, and\, Y \in \mathbb{S}^{n_u}$ satisfying the following LMIs:

$$(A_i P + B W_i) + (A_i P + B W_i)^T + 2\eta P \leq 0 \tag{4.2.19}$$

$$trace(Q^{\frac{1}{2}} P (Q^{\frac{1}{2}})^T) + trace(Y) \leq \gamma \tag{4.2.20}$$

$$\begin{bmatrix} -Y & R^{\frac{1}{2}} W_i \\ (R^{\frac{1}{2}} W_i)^T & -P \end{bmatrix} \leq 0 \tag{4.2.21}$$

$$i = 1, \cdots, 2^{n_\Psi} \tag{4.2.22}$$

and applying the transformation $K_i = W_i P^{-1}$.

It must be taken into account than an extra performance term, the decay rate η, is intentionally introduced for ensuring a fast dynamic response of the controller. This fast response plays a fundamental role in the cascade control scheme, as the outer loop (position control) assumes that the reference angles that generates as control commands are perfectly tracked.

4.2.2.3 Reference Tracking

It is known that the LPV controller computed previously can bring the output of the system to zero[26]. Nevertheless, if it is desired that the output tracks a generic reference $r(t)$, a feedforward control action $v(t) = F(t)r(t)$ as shown in Eq. (4.2.12) must be applied.

To have the output $y(t) \to r(t)$, it is needed a unit DC-gain from r to y, that is, the matrix $F(t)$ must be for every moment, the inverse of the DC-gain of the whole attitude subsystem including the designed controller and observer.

Then, the matrix $F(t)$, is the inverse of the DC-gain of the following extended LPV system

$$A_{ext}(\Gamma) = \begin{bmatrix} A(\Gamma) & -BK(\Gamma) \\ L(\Gamma)C & -A(\Gamma)L(\Gamma)C - BK(\Gamma) - L(\Gamma)DK(\Gamma) \end{bmatrix} \quad B_{ext}(\Gamma) = \begin{bmatrix} B \\ B + L(\Gamma)D \end{bmatrix}$$
$$C_{ext}(\Gamma) = \begin{bmatrix} C \\ -DK(\Gamma) \end{bmatrix} \qquad D_{ext} = D \tag{4.2.23}$$

The proposed LPV control scheme for the attitude subsystem can be seen in Fig. 4.2.1.

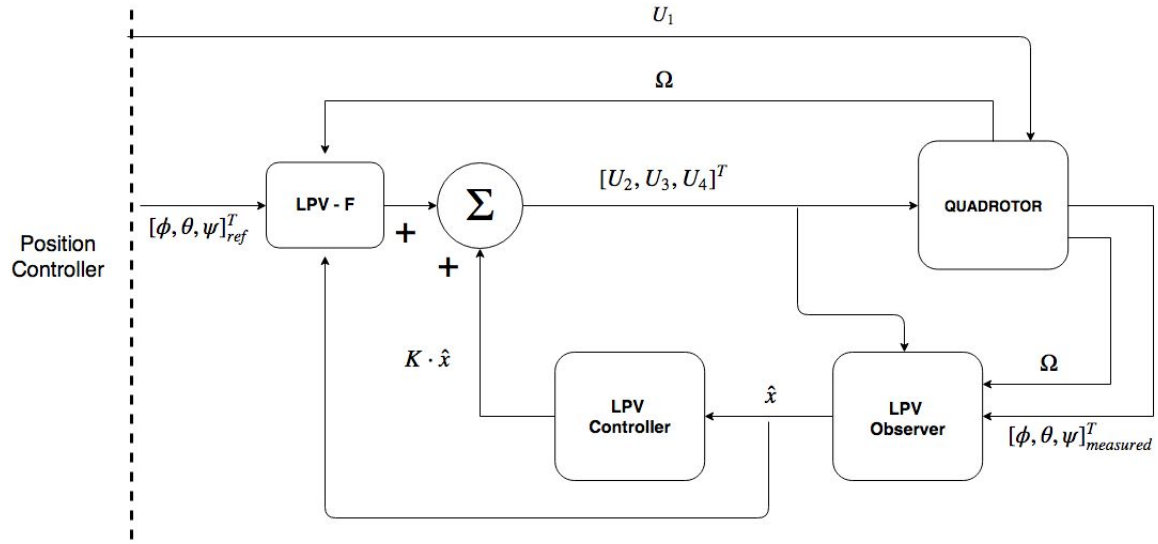


Figure 4.2.1: Attitude controller scheme

4.2.3 Attitude controller design with disturbance rejection

As an extension to the control scheme proposed in Section 4.2.2, here an LPV-LQR control structure is presented but with the addition of an Unknown Input Observer (UIO) that allows to estimate possible disturbances affecting the system. As previously mentioned, majority of the IMUs provide measures of the angular velocities in B-frame $(p,\, q\,, r)$. So simply applying the relationship expressed in Eq. (4.1.8) the Euler-angle rate could be obtained and all the attitude subsystem states available.

According the aforementioned, the system matrix C has now the form of a 6×6 identity matrix as shown in Eq. (4.2.24).

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2.24}$$

The first step in order to implement the Unknown Input Observer is to model the possible disturbances actuating in the system. As a hypothesis, it will be assumed that the attitude subsystem can be disturbed by the presence of unknown torques affecting in the quadrotor B-frame. With the "close to hovering" assumption expressed in Section 4.1, the unknown moments (M_x, M_y, M_z) are added to the attitude sub-model as follows:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}\dot{\theta}\Omega + \frac{U_2}{I_x} + \frac{M_x}{I_x} \tag{4.2.25}$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}\dot{\phi}\Omega + \frac{U_3}{I_y} + \frac{M_y}{I_y} \tag{4.2.26}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} + \frac{M_z}{I_z} \tag{4.2.27}$$

The resulting LPV model with the addition of the unknowns is expressed below:

$$\dot{x}(t) = A(\Gamma(t))x(t) + Bu(t) + EM_{dist}(t) \tag{4.2.28}$$

$$y(t) = Cx(t) + Du(t) \tag{4.2.29}$$

where M_{dist} is the unknown input vector, and the matrix E is:

$$E = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{I_x} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{I_z} \end{bmatrix} \tag{4.2.30}$$

According to the aforementioned, a Linear Parameter Varying Unknown Input Observer (LPV UIO) control structure is designed, aiming to estimate the unknown disturbances affecting the system, and also for filtering the data against noise in the sensors.

4.2.3.1 Unknown Input Observer design

The proposed UIO estimation scheme is developed for LPV systems affected by external disturbances. The disturbance estimation is based on computing the difference between the real system and the estimated outputs used for observation:

$$CEM_{dist} = \dot{y} - C(A\hat{x} + Bu) \tag{4.2.31}$$

Thus, naming the matrix Θ as the pseudo-inverse: $\Theta = (CE)^{+}$, the momentum disturbance estimation can be obtained as:

$$M_{dist} = \Theta(\dot{y} - C(A\hat{x} + Bu)) \tag{4.2.32}$$

Consequently, decoupling the considered disturbance, the attitude model of the system can be rewritten as follows:

$$\dot{\hat{x}} = A_o\hat{x} + B_o u + E\Theta\dot{y} \tag{4.2.33}$$

where:

$$A_o = (I - E\Theta C)A$$
$$B_o = (I - E\Theta C)B$$

Then, the state estimation will depend on the observer gain L (Eq. (4.2.16)), and presents the

form:

$$\dot{\hat{x}} = (A_o - LC)\hat{x} + B_o u + E\Theta\dot{y} + Ly \tag{4.2.34}$$

The computation of the L_i vertex of the observer, is done following the same procedure than in Eqs. (4.2.17) to (4.2.18), but making use of the new: $A_{oi} = (I - E\Theta C)A_i$ and $B_{oi} = (I - E\Theta C)B_i$.

4.2.3.2 Controller design

As the observer decouples the disturbance from the states, there is no need for modifying the controller designed for the first scenario. However, a compensation of the estimated unknown input Eq. (4.2.32) must be performed, and the applied control action has the following form:

$$u(t) = K(\Gamma(t))\hat{x}(t) + v(t) - M_{dist} \tag{4.2.35}$$

4.2.3.3 Reference tracking

The same feedforward control action $v(t) = F(t)\, r(t)$ for a generic reference $r(t)$ of the form (ϕ_{ref}, θ_{ref}, ψ_{ref}) is applied. Matrix $F(t)$ is computed as the inverse of the DC-gain of the extended system shown in Eq. (4.2.23).

Figure 4.2.2 shows the general scheme for the LPV-LQR attitude controller with the implementation of an Unknown Input Observer.

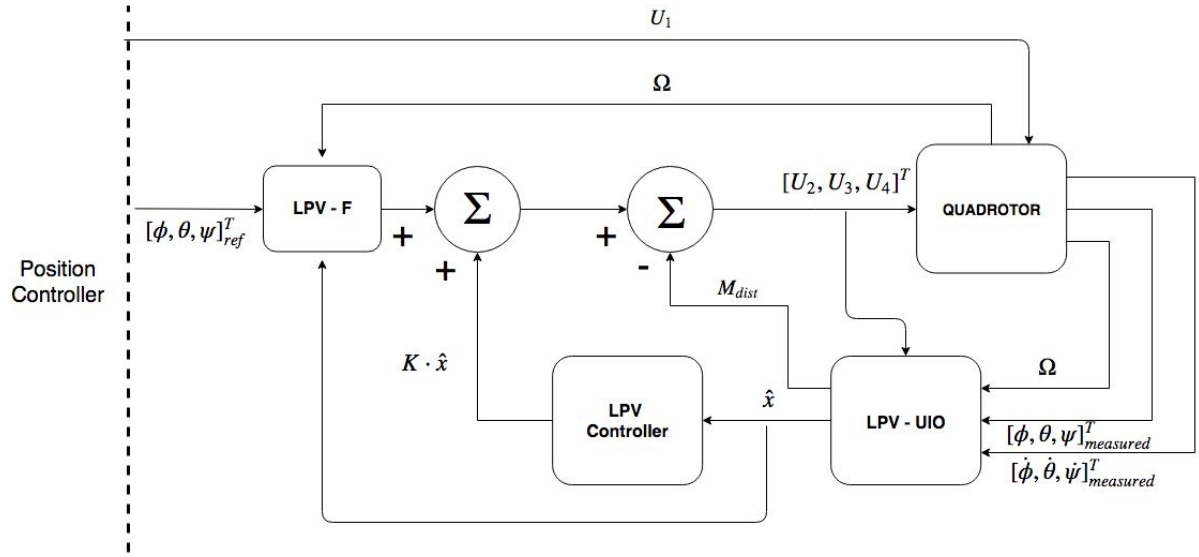


Figure 4.2.2: Attitude controller with disturbance rejection scheme

4.3 Translational control

In the development of the quadrotor mathematical model carried out in Section 3.2, two different reference frames were considered for referencing the equations:

1. **B-frame**: with all the dynamics equations of the quadrotor (translational and rotational) expressed WRT a reference frame fixed in the quadrotor center, Eqs. (3.2.22) to (3.2.26).

2. **H-frame**: with the translational equations developed in the inertial E-frame, and the rotational equations developed WRT the B-frame, Eqs. (3.2.40) to (3.2.45).

Mostly all of the path planners generate the desired references in position WRT an inertial Earth frame (x, y, z), or in velocity WRT the quadrotor B-frame (u, v, w). On the contrary, it does not make any sense to refer to velocities in E-frame or position in B-frame.

Depending if it is desired to stabilize in position or velocity, one of the two aforementioned equation sets is used for control design. Below a detailed analysis of the design procedure for the position control WRT E-frame and velocity control WRT B-frame is developed.

4.3.1 Position control (E-frame)

In the cascade control scheme explained in Section 4.1, the outer loop (with slower dynamics) is in charge of the position control, feeding the inner loop (faster dynamics) with the reference angles needed for reaching a desired position.

Given the position system, in E-frame, shown in Eqs. (4.3.1) to (4.3.3). The aim is to stabilize it, while following a desired reference yaw angle ψ_{ref} imposed by the path planner. Hence, for the position subsystem, the considered control actions are: (ϕ, θ, U_1). While the throttle U_1 needed is directly applied to the quadrotor, the obtained (ϕ, θ) together with the imposed ψ_{ref} are passed as the references for the attitude controller.

The translational subsystem equations WRT E-frame are:

$$\ddot{x} = (cos\phi\, sin\theta\, cos\psi + sin\phi\, sin\psi)\,\frac{U_1}{m} \tag{4.3.1}$$

$$\ddot{y} = (cos\phi\, sin\theta\, sin\psi - sin\phi\, cos\psi)\,\frac{U_1}{m} \tag{4.3.2}$$

$$\ddot{z} = -g + cos\phi\, cos\theta\,\frac{U_1}{m} \tag{4.3.3}$$

Expressed in the state-space form they look as follows:

$$\dot{x}_1 = x_2 \tag{4.3.4}$$

$$\dot{x}_2 = (cos\phi\, sin\theta\, cos\psi + sin\phi\, sin\psi)\,\frac{U_1}{m} \tag{4.3.5}$$

$$\dot{x}_3 = x_4 \tag{4.3.6}$$

$$\dot{x}_4 = (cos\phi\, sin\theta\, sin\psi - sin\phi\, cos\psi)\,\frac{U_1}{m} \tag{4.3.7}$$

$$\dot{x}_5 = x_6 \tag{4.3.8}$$

$$\dot{x}_6 = -g + cos\phi\, cos\theta\,\frac{U_1}{m} \tag{4.3.9}$$

where $x_1 = x,\ x_2 = \dot{x},\ x_3 = y,\ x_4 = \dot{y},\ x_5 = z,\ x_6 = \dot{z}$

4.3.1.1 Controller design

Aiming to deal with the strong non-linearities that the translational subsystem presents, a feedback-linearization control strategy is developed. Firstly, the error between the desired reference states and the measured/estimated states is defined:

$$\bar{x}_1 = x_1^{ref} - \hat{x}_1 \qquad \bar{x}_2 = x_2^{ref} - \hat{x}_2 \tag{4.3.10}$$

$$\bar{x}_3 = x_3^{ref} - \hat{x}_3 \qquad \bar{x}_4 = x_4^{ref} - \hat{x}_4 \tag{4.3.11}$$

$$\bar{x}_5 = x_5^{ref} - \hat{x}_5 \qquad \bar{x}_6 = x_6^{ref} - \hat{x}_6 \tag{4.3.12}$$

Making use of a feedback-linearization scheme, the translational subsystem, Eqs. (4.3.5) to (4.3.9), is reduced to a set of 3 identical 2nd order systems:

$$\begin{cases} \dot{\bar{x}}_1 = \bar{x}_2 \\ \dot{\bar{x}}_2 = v_x \end{cases} \tag{4.3.13}$$

$$\begin{cases} \dot{\bar{x}}_3 = \bar{x}_4 \\ \dot{\bar{x}}_4 = v_y \end{cases} \tag{4.3.14}$$

$$\begin{cases} \dot{\bar{x}}_5 = \bar{x}_6 \\ \dot{\bar{x}}_6 = v_z \end{cases} \tag{4.3.15}$$

Where the linearization terms (v_x, v_y, v_z), are computed in such a way that they stabilize each

of the 2nd order systems through a simple state feedback:

$$v_x = k_1^x \bar{x}_1 + k_2^x \bar{x}_2 \tag{4.3.16}$$

$$v_y = k_1^y \bar{x}_3 + k_2^y \bar{x}_4 \tag{4.3.17}$$

$$v_z = k_1^z \bar{x}_5 + k_2^z \bar{x}_6 \tag{4.3.18}$$

Once the values of (v_x, v_y, v_z) that stabilize each subsystem are computed, the required (ϕ, θ, U_1) that yields those values for an externally imposed ψ_{ref}, must be computed. This leads to a set of complex non-linear equations:

$$v_x = (cos\phi\, sin\theta\, cos\psi_{ref} + sin\phi\, sin\psi_{ref})\,\frac{U_1}{m} \tag{4.3.19}$$

$$v_y = (cos\phi\, sin\theta\, sin\psi_{ref} - sin\phi\, cos\psi_{ref})\,\frac{U_1}{m} \tag{4.3.20}$$

$$v_z = -g + cos\phi\, cos\theta\,\frac{U_1}{m} \tag{4.3.21}$$

In order to solve the previous equations, several mathematical transformations must be performed:

$$\frac{v_x}{v_z + g} = tan\theta\, cos\psi_{ref} + \frac{tan\phi\, sin\psi_{ref}}{cos\theta} \tag{4.3.22}$$

$$\frac{v_y}{v_z + g} = tan\theta\, sin\psi_{ref} - \frac{tan\phi\, cos\psi_{ref}}{cos\theta} \tag{4.3.23}$$

$$\tag{4.3.24}$$

Hereinafter, the following abbreviations will be used: $a = \frac{v_x}{v_z+g}$, $b = \frac{v_y}{v_z+g}$, $c = cos\psi_{ref}$, $d = sin\psi_{ref}$.

Hence, the previous set of equations can be rewritten as follows:

$$a = tan\theta\, c + \frac{\tan\phi\, d}{\cos\theta} \tag{4.3.25}$$

$$b = tan\theta\, d - \frac{\tan\phi\, c}{\cos\theta} \tag{4.3.26}$$

Carrying out several operations the value of pitch angle (θ) can be obtained in the following manner:

$$\tan\phi = \frac{cos\theta(a - tan\theta\, c)}{d} \tag{4.3.27}$$

$$\tan\phi = \frac{cos\theta(tan\theta\, d - b)}{c} \tag{4.3.28}$$

$$\cancel{cos\theta}(\frac{a - tan\theta\, c}{d}) = \cancel{cos\theta}(\frac{tan\theta\, d - b}{c}) \tag{4.3.29}$$

$$\tan\theta = \frac{\frac{a}{d} + \frac{b}{c}}{\frac{c}{d} + \frac{d}{c}} = \frac{(ac + bd)}{(\cancel{c^2 + d^2}^{\,1})} = ac + bd \tag{4.3.30}$$

Checking Eqs. (4.3.27) to (4.3.28), it can be seen that they keep equal but for those values of ψ_{ref} that make $c = 0$ ($\psi_{ref} = \pm\frac{\pi}{2}$) or $d = 0$ ($\psi_{ref} = 0,\ \psi_{ref} = \pi$) where the equations become indeterminate. In order to deal with these singularities, a threshold is introduced in order to use one equation or the other and avoid those singularities. So the computation of the roll angle (ϕ) follows the following rule:

$$\textbf{if}\ \ (|\psi_{ref}| < \frac{\pi}{4}\ or\ |\psi_{ref}| > \frac{3\pi}{4}) \tag{4.3.31}$$

$$\tan\phi = \frac{\cos\theta(\tan\theta\, d - b)}{c} \tag{4.3.32}$$

$$\textbf{else} \tag{4.3.33}$$

$$\tan\phi = \frac{\cos\theta(a - \tan\theta\, c)}{d} \tag{4.3.34}$$

Finally, the total lifting force U_1 is computed as:

$$U_1 = \frac{(v_z + g)m}{\cos\phi\cos\theta} \tag{4.3.35}$$

It can be seen that the thrust force U_1 also presents a singularity if the roll angle is ($\phi = \pm\frac{\pi}{2}$) or if the pitch angle is ($\theta = \pm\frac{\pi}{2}$), that is, when the quadrotor is perpendicular to the ground.

From Eqs. (4.3.30), (4.3.32), (4.3.34) and (4.3.35), the (ϕ, θ, U_1) commands that stabilizes the drone position around a (x, y, z) references are obtained. As explained in Section 4.1, U_1 is applied directly to the quadrotor, meanwhile the obtained (ϕ, θ) plus ψ_{ref} are the reference angles that the attitude control must track.

4.3.1.2 Position observer

For the previous position controller design it is assumed that all the states: $x, \dot{x}, y, \dot{y}, z, \dot{z}$ are known. However, for the development of the position controller it will be considered that the

quadrotor has an on-board GNSS receiver that only provides accurate information of the current position WRT E-frame (x, y, z), discarding possible information on the speed.

In order to obtain a velocities estimation, three different state observer are designed for each of the linearized subsystems previously developed (see Eqs. (4.3.13) to (4.3.15)). According the aforementioned equations, each one of those subsystems presents a matrix A and a matrix C as follows:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.3.36}$$

Taking advantage of the duality between the control problem $K \rightarrow (A, B)$ and the observation problem $L^T \rightarrow (A^T, C^T)$, the observer gain L can be easily determined by pole placement. As general rule, the dynamics of the observer will be designed to be faster than the controller dynamics, that is, its poles will be allocated several times to the left than the controller ones.

It must be taken into account that the procedure developed is only for determining the gain of one of the subsystems. Actually, in order to simplify things, the three subsystem can be gathered in a general one, whose matrices would take the following form:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.3.37}$$

As seen in Fig. 4.3.1, it must be taken into account that in the observer implementation, Eq. (4.3.38), the inputs of the model are the feedback-linearization terms (v_x, v_y, v_z).

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + L(y - \hat{y}) \tag{4.3.38}$$

$$\hat{y}(t) = C\hat{x}(t) \tag{4.3.39}$$

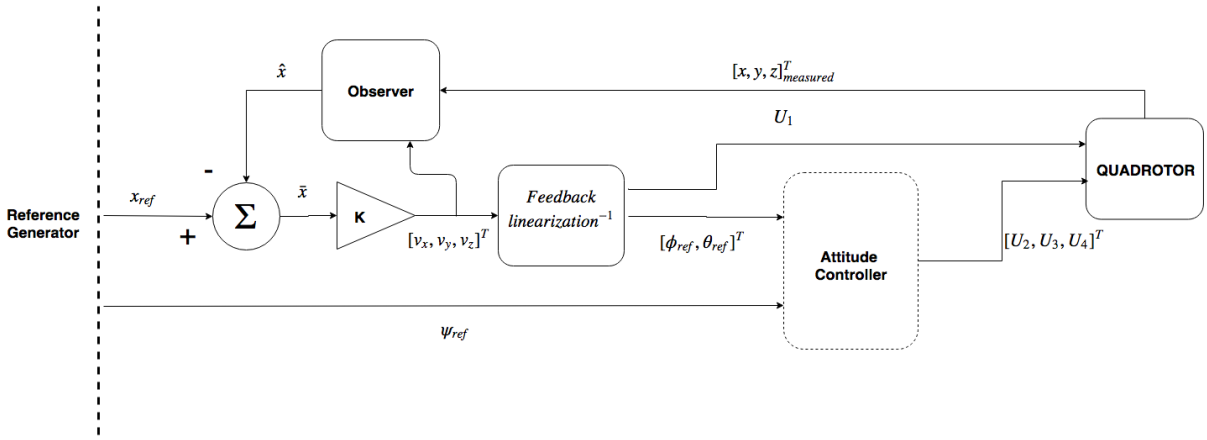


Figure 4.3.1: Position control scheme

4.3.2 Velocity control (B-frame)

Considering the cascade control structure proposed, the outer loop can be modified in order to track velocities in the quadrotor's body frame, instead of positions WRT E-frame. This design proposal is developed in order to meet some trajectory planner commands, that generate the trajectory reference modifying the direction of the velocity vector expressed on the B-frame. This is possible due to the current IMUs are able to integrate their accelerometer measures, and provide, with a reasonable accuracy, measures of the translational speed of the device.

Aiming to design an effective velocity control, the assumption that the drone is close to the hover position is extended considering that no fast rotational speeds are demanded, which means that the Coriolis effects for that flight profile are almost null. According to that hypothesis, the quadrotor translational model WRT B- frame, Eqs. (3.2.22) to (3.2.26), is now simplified as:

$$\dot{u} = \sin\theta\, g \tag{4.3.40}$$

$$\dot{v} = -\sin\phi\cos\theta\, g \tag{4.3.41}$$

$$\dot{w} = -\cos\phi\cos\theta\, g - \frac{U_1}{m} \tag{4.3.42}$$

In order to deal with the non-linearities of the equations, also a feedback linearization strategy is used (same than Section 4.3.1). Thus, the first step is to compute the error between the desired velocity reference, and the measured one:

$$\bar{u} = u_{ref} - u \tag{4.3.43}$$

$$\bar{v} = v_{ref} - v \tag{4.3.44}$$

$$\bar{w} = w_{ref} - w \tag{4.3.45}$$

After that, feedback linearization variables are defined comprising all the non-linearities, and reducing the system to 3 identical 1st order lineal systems:

$$\dot{\bar{u}} = v_x \tag{4.3.46}$$

$$\dot{\bar{v}} = v_y \tag{4.3.47}$$

$$\dot{\bar{w}} = v_z \tag{4.3.48}$$

where the linearization terms are computed as simple proportional controllers:

$$v_x = k_1^x \, \bar{u} \tag{4.3.49}$$

$$v_y = k_1^y \, \bar{v} \tag{4.3.50}$$

$$v_z = k_1^z \, \bar{w} \tag{4.3.51}$$

For the obtained values (v_x, v_y, v_z) that stabilizes the system, the required (ϕ, θ, U_1) is obtained solving the following set of equations:

$$v_x = \sin\theta \cdot g \tag{4.3.52}$$

$$v_y = -\sin\phi\cos\theta \cdot g \tag{4.3.53}$$

$$v_z = -\cos\phi\cos\theta \cdot g - \frac{U_1}{m} \tag{4.3.54}$$

The previous equations can be easily solved as expressed in Eqs. (4.3.55) to (4.3.57).

$$\sin\theta = \frac{v_x}{g} \tag{4.3.55}$$

$$\sin\phi = \frac{-v_y}{\cos\theta g} \tag{4.3.56}$$

$$U_1 = (v_z + \cos\phi\cos\theta g) \cdot m \tag{4.3.57}$$

It can be seen that the obtained solution presents a singularity in $\theta = \pm\frac{\pi}{2}$, that is when the system is completely tilted. One possible solution to this problem is to implement a saturation

in the achievable angles, in order to avoid that the system gets close to that non-safety extreme angles. A scheme of the simple velocity control proposal can be seen in Fig. 4.3.2.

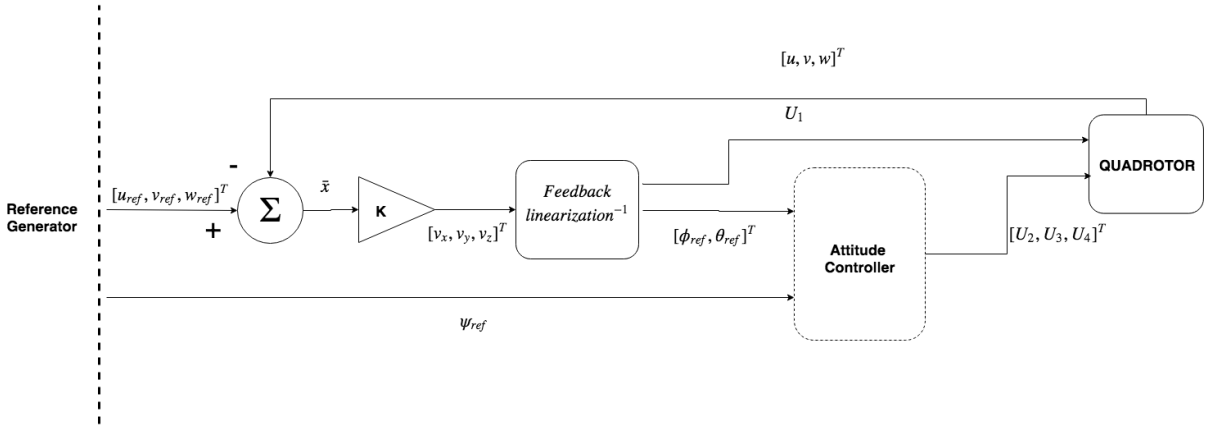


Figure 4.3.2: Velocity control scheme

Chapter 5

Validation

In this chapter, the different control strategies developed in Chapter 4 are tested in simulation, trying to emulate the behaviour of the quadrotor used in the real implementation stage. For achieving that purpose the following steps are performed throughout this chapter: first, the differences that the more complex validation model introduce, are presented. Then, a brief description of the target quadrotor parameter values is carried out. Afterwards, a set of proposed scenarios in order to test the different control variations, and the initial conditions considered, are developed. The following step is to tune the designed controllers in order to meet some performance requirements. Finally, the different control architectures presented are evaluated in the already defined simulation scenarios.

5.1 Validation model

Due to the controller design purpose of the models developed in Chapter 3, only the effects that have a significant relevance in the quadrotor dynamics were considered, omitting those terms with a residual influence, that is, several magnitude orders smaller for a reduced size quadrotor flying at low speed, that would increase the controller complexity exponentially. Furthermore, in Chapter 4 the hypothesis of near to hovering condition was made, getting rid of the coupling existing between angular velocities in B-frame (p, q, r) and Euler angle rates ($\dot{\phi}$, $\dot{\theta}, \dot{\psi}$), see Eq. (4.1.8) for $T(\Theta) = I$.

Nevertheless, before implementing the obtained controller in the real platform, a validation stage must be carried out. For that purpose, a more complex model developed by a team of the *Drexel University* [31], and later modified by the Advanced Control Systems Group in order to adjust to the available quadrotor, was used. Later in this chapter, a detailed description of

the extended terms introduced in the validation model regarding the model used in the design stage, Eqs. (4.1.9) to (4.1.14), is carried out.

In the work developed by [31], a quadrotor model is implemented as a system-function (S-function) in Matlab®. This model is dependent of a set of parameters externally loaded, that define the specifications of the quadrotor model studied: The AscTec® *Hummingbird* quadcopter (see Section 5.2). The main differences between the extended validation model, and the simplified model used for control design, can be listed as follows:

1. "Near to hover" hypothesis not considered: In the model used for validation, the hypothesis that the angular velocities in B-frame were the same than the Euler-angle rates, is not considered. According to that, the validation model would be given by Eqs. (4.1.1) to (4.1.6). So, in order to know the quadrotor attitude in Euler-angles (E-frame) the first order differential equations expressed in Eq. (5.1.1), must be considered.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} = T(\Theta)_B^E \omega_B \tag{5.1.1}$$

2. Motor dynamics: for all the control structures developed in Chapter 4, the motor speeds are considered the final control commands applied to the quadrotor. Those motor speeds in rpm (Ω_1, Ω_2, Ω_3, Ω_4) are obtained from (U_1, U_2, U_3, U_4) through the inverse of Eqs. (4.1.15) to (4.1.18), assuming that can be achieved instantaneously.

   However, in the real world the brushless electric motors that quadrotors use, have their own dynamics as well as a specific working range, that is: a minimum rotational speed (greater that zero), and a maximum rotational speed. A detailed study of small electric motors can be found in [24]. In the validation model used, the electric motors are modelled in the following manner:

$$\dot{x}(t) = Ax(t) + B\,\Omega_{desired}(t) \tag{5.1.2}$$

$$\Omega(t) = Cx(t) \tag{5.1.3}$$

   with

$$A = -\frac{1}{T} \quad B = 1 \quad C = \frac{1}{T} \tag{5.1.4}$$

where T is a delay time constant specified in Section 5.2. The output Ω is saturated from above and below, imitating the working range of the motors as follows:

$$\Omega_{applied} = \begin{cases} \Omega_{min} & if \quad \Omega < \Omega_{min} \\ \Omega & if \quad \Omega_{min} \leq \Omega \leq \Omega_{max} \\ \Omega_{max} & if \quad \Omega > \Omega_{max} \end{cases} \tag{5.1.5}$$

3. Disturbances: The provided Simulink® model, also allows to study the effect of external disturbances actuating on the quadrotor. These disturbances are modelled as forces affecting the translational equations WRT E-frame, and torques affecting the rotational equations WRT B-frame in the validation model. The resulting model for unknown external disturbances is shown below:

$$\ddot{x} = (cos\phi sin\theta cos\psi + sin\phi sin\psi)\frac{U_1}{m} + \boldsymbol{F_{distx}} \tag{5.1.6}$$

$$\ddot{y} = (cos\phi sin\theta sin\psi - sin\phi cos\psi)\frac{U_1}{m} + \boldsymbol{F_{disty}} \tag{5.1.7}$$

$$\ddot{z} = -g + cos\phi cos\theta\frac{U_1}{m} + \boldsymbol{F_{distz}} \tag{5.1.8}$$

$$\dot{p} = qr\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}q\Omega + \frac{U_2}{I_x} + \boldsymbol{M_{distx}} \tag{5.1.9}$$

$$\dot{q} = pr\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}p\Omega + \frac{U_3}{I_y} + \boldsymbol{M_{disty}} \tag{5.1.10}$$

$$\dot{r} = pq\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z} + \boldsymbol{M_{distz}} \tag{5.1.11}$$

Furthermore, there are other effects that could be modelled in order to improve the realism of the validation model. Some of these unmodelled terms are:

- Dissmetry of lift: The dissymmetry of lift is the difference in lift that exists between the advancing half of the rotor disk and the retreating half in a moving rotating propeller. It is caused by the fact that in directional flight, the aircraft relative wind is added to the rotational relative wind on the advancing blade, and subtracted from the retreating blade. Helicopters vary the pitch angle of the blade cyclically in order to avoid this deviation in the lift vector. However, the quadrotor propellers do not modify its blade angle causing a tilt in the lift vector of each propeller that increases its deviation from the vertical as the relative velocity with respect the wind increases.

  Although this phenomenon must be considered for really big quadrotors, during the development of this project it was assumed that the target were small quadrotors and the

dissmetry of lift effect was not took into account in the model used for control design, neither in the validation model provided by [31].

- Ground effect: The ground effect is a condition of improved performance encountered when operating near (within 1/2 rotor diameter) of the ground. It is due to the interference of the surface with the airflow pattern of the rotor system, and it is more pronounced as the ground is approached. Increased blade efficiency while operating in ground effect is due to two separate and distinct phenomena:

  1. The reduction of the velocity of the induced airflow. Since the ground interrupts the airflow under the quadcopter, the entire flow is altered. This reduces downward velocity of the induced flow.
  2. The downward and outward airflow patterns tend to restrict vortex generation. This makes the outboard portion of the rotor blade more efficient and reduces overall system turbulence.

  As the focus of this project is to deal with small quadrotors, and as previously stated: the ground effect only affects when operating within 1/2 of the rotor diameter of the ground, then this effect was discarded during the modelling stage. It was expected that the ground effect was actually taken into consideration in the validation model, aiming to check the behaviour of the drone in the take-off and landing stages. Nevertheless, the provided validation Simulink® model did not comprise it, so it was decided to only check the controllers performance during flight and omit the take-off and landing stages.

The model released by [31] is developed for velocities (translational and rotational) in the B-frame, performing and E-frame integration for obtaining the position variables and Euler-angles in E-frame. According to this, the user is able to obtain in every moment the following variables:

- (x, y, z) position in E-frame.
- (u, v, w) translational velocities in B-frame.
- (ϕ, θ, ψ) Euler angles in E-frame (Z-Y'-X" convention).
- (p, q, r) angular velocities in B-frame.

5.2 Asctec Hummingbird model parameters

All the models developed so far, are dependent on a set of parameters that are specific for each quadrotor. As explained in Chapter 1, the final aim of this thesis is to implement and test the proposed control strategies in a real platform: an AscTec® *Hummingbird* quadcopter; an analysis of its main characteristics is carried out in Chapter 6. Thus, the parameters that define the validation model must correspond with the *Hummingbird* platform.

Although some of the parameters can be obtained from the user manuals provided by AscTec® company [32], others like the aerodynamic coefficients must be estimated carrying out identification experiments as the ones performed in [33] for this specific platform. After an intensive search, and several tries with the real quadrotor, Table 5.2.1 shows the final values used for obtaining the controllers and for simulate the *Hummingbird* performance.

Parameter	**Value**	**Units**
g	9.81	m/s^2
d	0.171	m
m	0.698	Kg
Ix	0.0034	$Kg\,m^2$
Iy	0.0034	$Kg\,m^2$
Iz	0.006	$Kg\,m^2$
J_{TP}	$1.302 \cdot 10^{-6}$	$N\,m\,s^2$
ct	$7.6184 \cdot 10^{-8}$	$N\,s^2$
cq	$2.6839 \cdot 10^{-9}$	$N\,m\,s^2$
cr	37.625	$\%RPM$
b	1075	RPM
minThr	5	$\%RPM$
T	0.056	s

Table 5.2.1: Parameter values Hummingbird

This set of parameters is loaded as a Matlab® struct, and used for the validation model in order to imitate faithfully the real platform behaviour. In the work developed by [31], they provide a user-friendly graphical interface as the one shown in Fig. 5.2.1, that allows to generate the parameter set for each target quadrotor with a better insight of the meaning of each parameter.

Simulating the characteristics of the real platform: the read speeds are ranged in the interval $m = [0, 200]$, that related with speed in RPM through the following equation:

$$motorRPM = c_r \cdot m + b = 37.625 \cdot m + 1075 \tag{5.2.1}$$

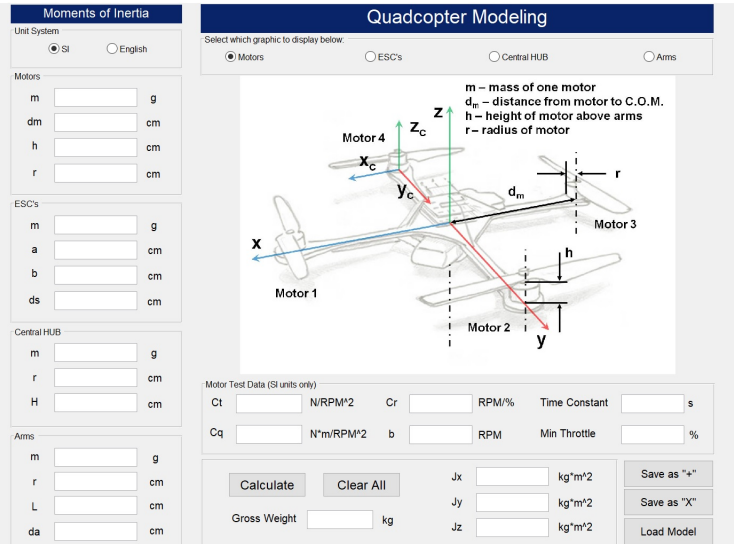


Figure 5.2.1: Parameter definition graphical interface

5.3 Proposed scenarios

Throughout Chapter 4, different control strategies were developed depending on the variables provided by the quadcopter sensors. In order to test the performance in simulation of all of them, different scenarios were considered:

- **First scenario**: Control over the E-frame position, and the yaw angle (ψ_{ref}) is desired. The available variables are the current position (x, y, z) for the outer control loop, and the Euler angles (ϕ, θ, ψ) for the attitude controller.

- **Second scenario**: Also a controller that allows to stabilize the E-frame position and the yaw angle (ψ_{ref}) is aimed. The difference with respect the first scenario is that the platform sensors provide information about the angular velocities in B-frame (p, q, r), allowing the implementation of a disturbance rejection strategy for the attitude controller, like the one developed in Section 4.2.3.

- **Third scenario**: Control over the B-frame velocities, and the yaw angle (ψ_{ref}). The available measurements are the translational speed in B-frame (u, v, w) for the outer velocities controller, and the Euler angles (ϕ, θ, ψ) for the attitude controller.

5.4 Initial Conditions for Simulation

In order to compare the different control schemes, and different settings for each one of them, all the simulation runs will proceed in the same conditions. Aiming to simulate the posterior

implementation in the real platform: the considered initial condition is the drone hovering at 2 meters over the floor, as expressed in Table 5.4.1.

Those initial conditions are selected because the procedure to test the designed controllers in real experiment flight is: first stabilise the platform in a hovering profile making use of the already defined low-level controller that the *Hummingbird* quadcopter counts on, and once it is safe, manually switch to de designed one to check its performance during flight.

VARIABLE	INITIAL CONDITION
ϕ	0 rad
θ	0 rad
ψ	0 rad
p	0 rad/s
q	0 rad/s
r	0 rad/s
x	0 m
y	0 m
z	2 m
u	0 m/s
v	0 m/s
w	0 m/s
ω_{motor1}	4500 rpm
ω_{motor2}	4500 rpm
ω_{motor3}	4500 rpm
ω_{motor4}	4500 rpm

Table 5.4.1: Initial conditions for simulation

5.5 Parameter tuning

So far, the development of the controllers carried out in Chapter 4 was made in a generic manner. However, there are some parameters that must be tuned in order to adjust the controllers performance for this specific problem.

5.5.1 Attitude controller LPV-LQR parameters

5.5.1.1 Scheduling variables intervals

In the LPV control technique, the scheduling variables (x_2, x_4, Ω) = ($\dot{\phi}$, $\dot{\theta}$, Ω) are assumed to take values in an interval known a priori. For determining the extreme values of those intervals, several simulation runs were carried out, and the selected ranges are such that encompass the ($\dot{\phi}$, $\dot{\theta}$, Ω) values in a standard attitude control profile. Finally, the considered intervals used

throughout the whole control design stage are the following:

$$\dot{\phi} = x_2 \in [-2, 2]\ rad/s \tag{5.5.1}$$

$$\dot{\theta} = x_4 \in [-2, 2]\ rad/s \tag{5.5.2}$$

$$\Omega \in [-50, 50]\ rad/s \tag{5.5.3}$$

5.5.1.2 Weights

The adjustment of the LPV-LQR parameters (Q , R and γ) in the attitude controller is made by means of the root mean square error (RMSE) approach. This approach allows to find suitable control parameters by minimizing it. In the tuning process it was observed that the system intrinsic fast dynamics require a tight control, meanwhile the electric motors of the quadrotor allows almost instantaneous changes in their rotation speed. Hence, the weights in Q corresponding to the error in the states are set greater than the input weights in R.

In order to tune the LQR parameters, those that minimize the RMSE for each of the angles (ϕ, θ, ψ) tracking a real scenario reference will be chosen. That is: the roll and pitch (ϕ_{ref}, θ_{ref}) given by the position controller, and an externally defined yaw angle (ψ_{ref}). In order to achieve that, a simple trajectory is defined with the following references:

- Initial conditions in Table 5.4.1
- Reference change: $0.5\,m$ step in x position at $(t = 5\,s)$
- Reference change: $1\,m$ step in y position at $(t = 10\,s)$
- Reference change: $\pi/2\,rad$ step in ψ_{ref} at $(t = 15\,s)$

Maintaining the $\psi_{ref} = 0$, with the first change in x position, the pitch angle performance is checked, while with the change in y position the roll angle is analysed. For the previously described trajectory, the obtained RMSE as a function of the Q and R values are shown in Table 5.5.1.

Although it could be expected that the RMSE would continue decreasing as the weights of matrix Q are increased, the introduction of saturations in the validation model, simulating the maximum and minimum rotational speeds of the motors affects the performance. After several simulation runs the selected weights can be seen in bold in Table 5.5.1. Figure 5.5.1 shows

RMSE			LQR Parameters	
ϕ	θ	ψ	Q (diagonal values)	R (diagonal values)
0.0309	0.0156	0.2024	$[1, 1, 1, 1, 1, 1]$	$[0.1, 0.1, 0.1]$
0.0307	0.0152	0.2009	$[5, 5, 5, 5, 5, 5]$	$[0.1, 0.1, 0.1]$
0.0312	**0.0156**	**0.1965**	$\mathbf{[10, 10, 10, 10, 10, 10]}$	$\mathbf{[0.1, 0.1, 0.1]}$
0.0311	0.0169	0.2122	$[50, 50, 50, 50, 50, 50]$	$[0.1, 0.1, 0.1]$
0.0362	0.0232	0.2562	$[100, 100, 100, 100, 100, 100]$	$[0.1, 0.1, 0.1]$

Table 5.5.1: LQR parameter tuning

the general Simulink ® layout used for simulation, where the red block contains the validation model released by [31].

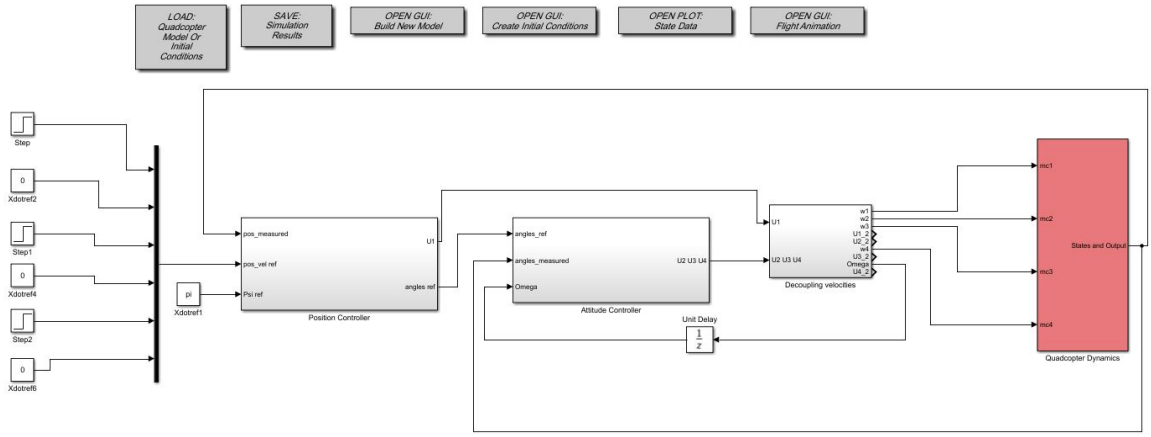


Figure 5.5.1: General Simulink Scheme

Figure 5.5.2 shows the references generated by the outer loop position controller for the tuning trajectory aforementioned, as well as the system response for the selected Q and R matrices. The same values of Q and R are fixed for the observer design.

5.5.1.3 Decay rates

As explained in Section 4.2.2, in the LQR problem an extra performance term is added: η for the controller and λ for the observer. Thanks to those decay rates the dynamics of the controller/observer can be set as fast as desired.

Controller decay rate η

It is fundamental for the correct performance of the whole cascade control scheme that the dynamics of the inner loop are faster than the ones of the outer loop. In such a way that for the

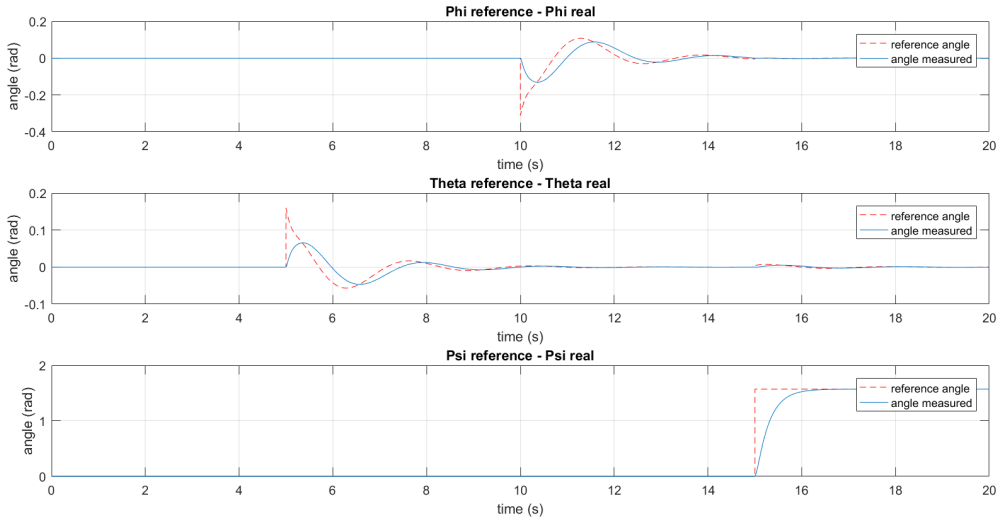


Figure 5.5.2: Plant response for the selected Q and R weights

position control, with a slower sample time, it could be assumed that the required angles are perfectly tracked.

According to the aforementioned discussion, and after an iterative process, finally the decay rate η in the attitude controller design is fixed to 6. Assuring that the slowest closed-loop pole of the subsystem is located to the left of -6.

Observer decay rate λ

Related to the attitude observer decay rate λ, it will be imposed as a rule of thumb that its dynamics are several times faster than the controller ones. The final value of the λ decay rate is fixed to 30, imposing an observer dynamics at least five times faster than the controller ones.

5.5.2 Position controller

5.5.2.1 Position controller gains

For selecting the feedback linearization control gains, it was used as insight the work developed by [34] and [35]. After an iterative process, it was decided to fix the gain values of each separate controller in $k_1^i = 3, k_2^i = 4$. Fixing the closed-loop poles of the each subsystem expressed in Eqs. (4.3.13) to (4.3.15), in $p = [-1+0j, -3+0j]$. In that manner, it is assured that the slowest poles of the attitude subsystem is at least two times faster than the fastest pole of the position subsystem.

According to that, the set of equations: Eqs. (4.3.16) to (4.3.18), can be expressed as a state feedback of the error states through the K gain matrix shown in Eq. (5.5.4).

$$K = \begin{bmatrix} 3 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 4 \end{bmatrix} \tag{5.5.4}$$

5.5.2.2 Position observer gains

In the position observer design, the observer gains are selected such that the observer closed-loop poles are fixed in $p = [-10 + 0j, -10 + 0j]$, assuring in that manner faster dynamics than the position controller ones. In order to achieve that pole placement, for each of the subsystem whose matrices A and C are given in Eq. (4.3.36) the observer gains are $L_1^i = 20, L_2^i = 100$. For the augmented system as expressed in Eq. (4.3.37), the total matrix L has the following form:

$$L = \begin{bmatrix} 20 & 0 & 0 \\ 100 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 20 \\ 0 & 0 & 100 \end{bmatrix} \tag{5.5.5}$$

5.5.3 Velocity controller

5.5.3.1 Velocity controller gains

When instead of the position controller, the outer loop is substituted in order to track velocities in B-frame, the resulting feedback linearized system Eqs. (4.3.46) to (4.3.48) can be easily controlled. Fixing the controller gain of each susbsystem to $k^i = 4$, its closed-loop poles are located in $p = [-4+0j]$. Gathering the three subsystems into one equation, the resulting control gain matrix K is as follows

$$K = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix} \tag{5.5.6}$$

Table 5.5.2 collects all the control tuning parameters aforementioned, as well as their final value.

Description	Tunning Variable	Selected Value
Att. Controller LQR parameter	Q (diag)	$[10, 10, 10, 10, 10, 10]$
Att. Controller LQR parameter	R (diag)	$[0.1, 0.1, 0.1]$
Att. Controller Decay Rate	η	6
Att. Observer LQR parameter	Q (diag)	$[10, 10, 10, 10, 10, 10]$
Att. Observer LQR parameter	R (diag)	$[0.1, 0.1, 0.1]$
Att. Observer Decay Rate	λ	30
Pos. controller gain (1-subsystem)	$[k_1^i, k_2^i]$	$[3, 4]$
Pos. observer gain (1-subsystem)	$[L_1^i, L_2^i]^T$	$[20, 100]^T$
Vel. controller gain (1-subsystem)	$[k^i]$	$[4]$

Table 5.5.2: Control parameter values

5.6 Results obtained in simulation

Below, the performance in simulation of the suitable control schemes for the different scenarios presented in Section 5.3, is analysed. In addition, the system will be exposed to disturbances affecting its attitude subsystem, in order to check the effectiveness of the disturbance rejection mechanism.

For the first two scenarios, the Simulink® general layout has the same form than the one depicted in Fig. 5.5.1. In simulation, direct control commands on the rotational motor speeds must be passed to the model. Thus, before the system plant in red, an intermediate block called *decoupling velocities* appears. This block relates the considered control commands (U_1, U_2, U_3, U_4) with the addressed rotational motor speeds through Eqs. (4.1.15) to (4.1.18).

5.6.1 First scenario

For the sake of simplicity in the obtained graphs, a simple trajectory will be generated in this first scenario. That trajectory is a concatenation of different step changes in the position of the quadrotor for the different axes WRT the E-frame, meanwhile the yaw angle is desired to track a sinusoidal wave. Starting in the initial conditions given in Table 5.4.1, the quadrotor is asked to track the reference changes shown in Table 5.6.1.

Reference Change	Value	Time (s)
X position	$0.5\,m$	2.5
Y position	$0.75\,m$	10
Z position	$1\,m$	17.5
ψ angle	$\frac{\pi}{2}\sin(\frac{2\pi}{10}t + 0)$	$0 \rightarrow 20$

Table 5.6.1: Trajectory definition

As explained in Section 5.3, for this first scenario only the Euler angles (ϕ, θ, ψ), feeding the attitude controller, and the position variables (x, y, z), feeding the position controller, are measured. Hence, for both controllers, observers as described in Chapter 4 must be implemented in order to estimate the remaining states. For the simulations shown both observers are well initialized $(t = 0)$, with the initial conditions of the quadrotor.

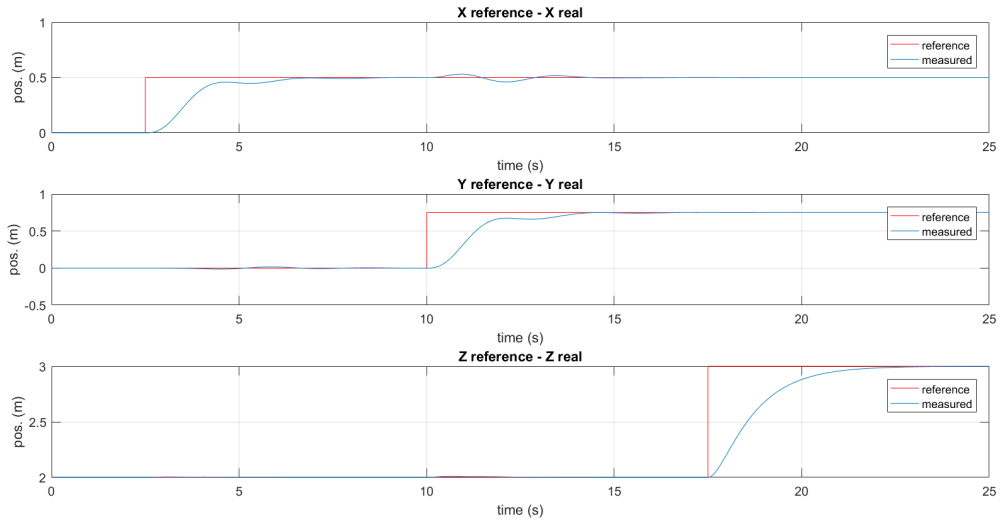


Figure 5.6.1: Quadrotor position response

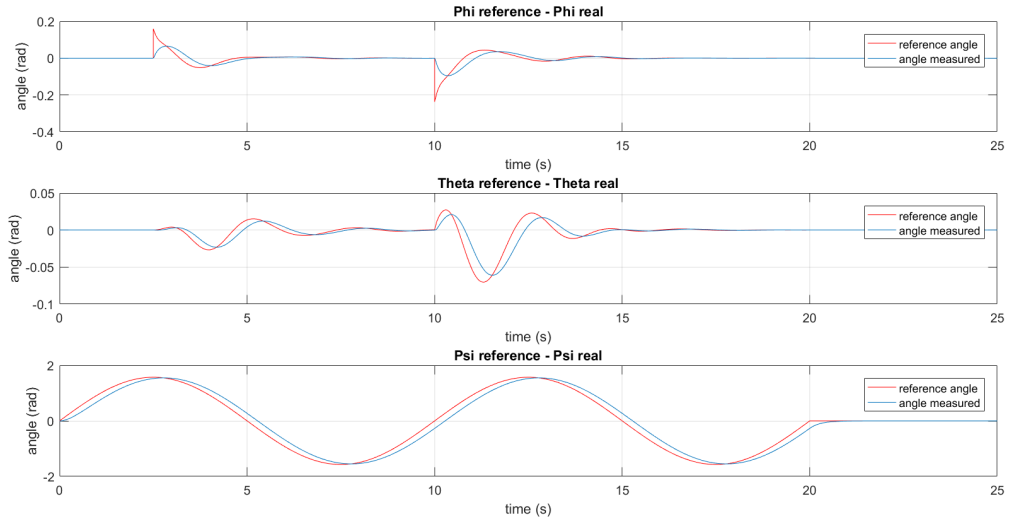


Figure 5.6.2: Quadrotor attitude response

For the trajectory described in Table 5.6.1, the position achieved by the outer position control loop is shown in Fig. 5.6.1. While Fig. 5.6.2 shows the inner-loop attitude control performance

tracking: ϕ_{ref}, θ_{ref} given by the position controller, and the externally imposed sinusoidal reference ψ_{ref}. The control architecture used is an LPV-LQR attitude controller/observer for the inner loop, and a proportional feedback-linearization position controller/observer in the outer loop.

With the proposed control methodology, the speed for which the quadrotor stabilizes over a new (x, y, z) point cannot be externally set, but is imposed through the position controller dynamics. If instead of a set of waypoints, the path planner generates a continuous E-frame reference trajectory, with the translational speeds determined according the position subsystem dynamics, complex trajectories could be tracked like a spiral as shown in Fig. 5.6.3. Figure 5.6.4 shows the position and attitude response of the system for the 3D spiral reference with a yaw angle ($\psi_{ref} = 0$).

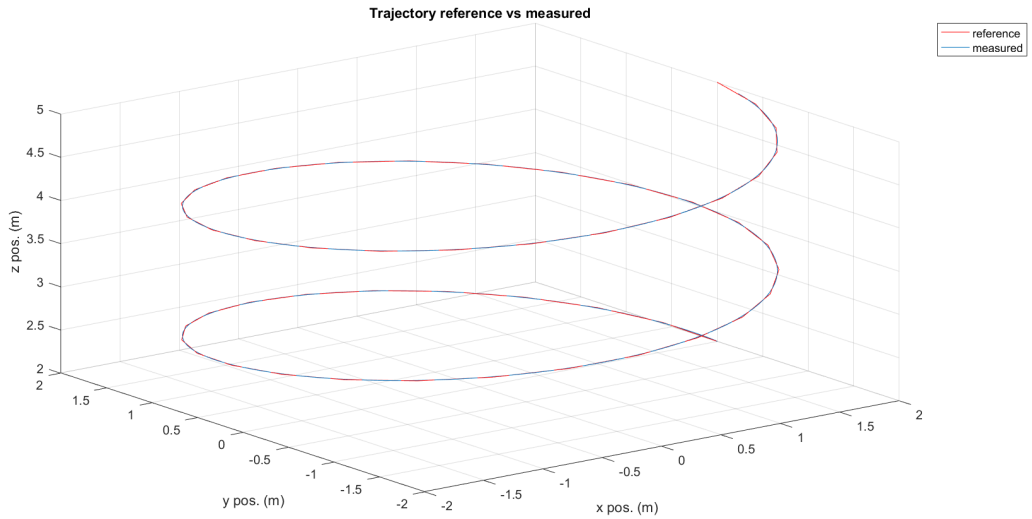
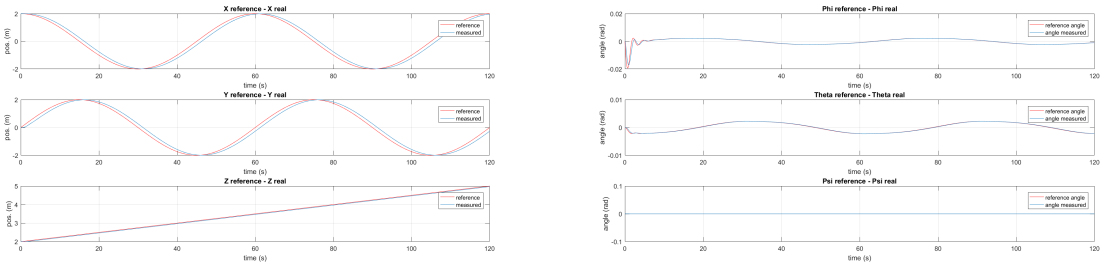


Figure 5.6.3: Quadrotor response - spiral trajectory

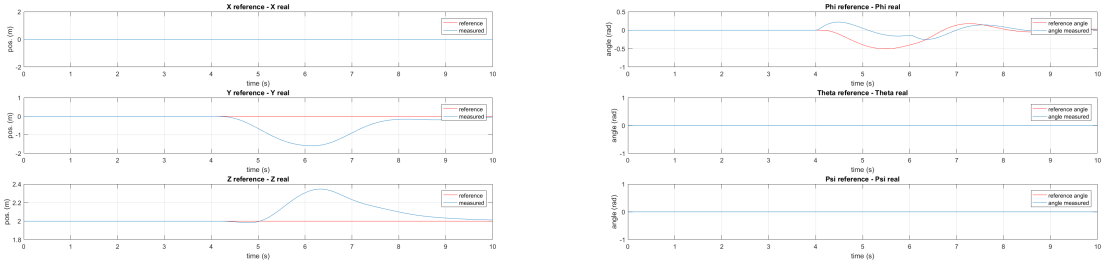


(a) Position response (b) Attitude response

Figure 5.6.4: Quadrotor position/attitude - spiral trajectory

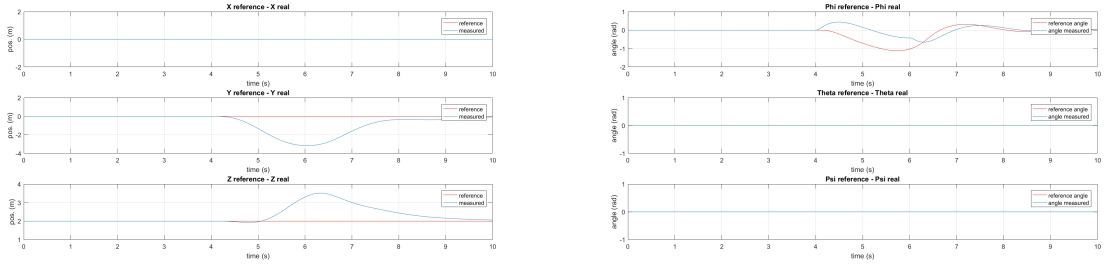
5.6.1.1 Disturbances

Aiming to show the effect of disturbances in the whole system performance, the quadrotor is asked to maintain its initial conditions (Table 5.4.1) while different disturbances are applied. The considered disturbances are torques applied in the B-frame axis of the validation model. Applying a step disturbance over the B-frame X-axis that acts in the time interval $t = [4s, 6s]$, the system response to different disturbance intensities can be seen in Figs. 5.6.5 to 5.6.7. Those figures also show, how even a small disturbance in the attitude subsystem, has an important effect into the position subsystem.



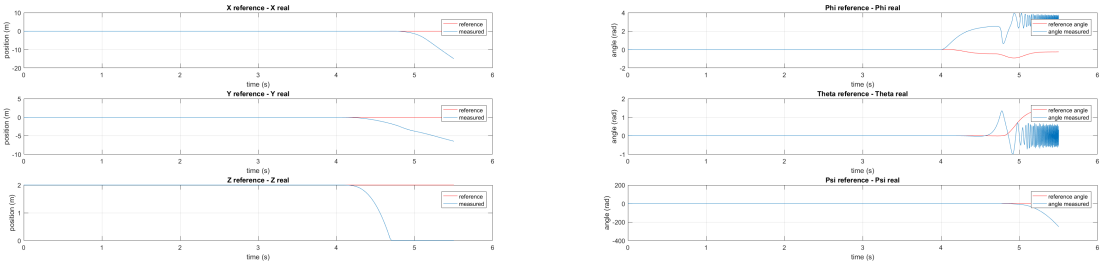
(a) Position response (b) Attitude response

Figure 5.6.5: Disturbance moment - $M_x = 0.05\,Nm$



(a) Position response (b) Attitude response

Figure 5.6.6: Disturbance moment - $M_x = 0.1\,Nm$



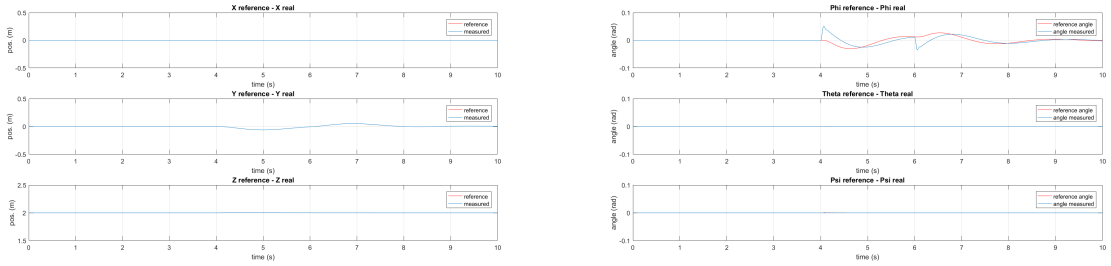
(a) Position response (b) Attitude response

Figure 5.6.7: Disturbance moment - $M_x = 0.5\,Nm$

5.6.2 Second scenario

The second scenario allows to address the problem of disturbance sensitivity, which as previously shown is specially relevant when affecting the attitude model. As commented in Chapter 4, this is a realistic approach as the IMU of the *Hummingbird* (and most of the IMUs) gives information about the Euler angles (ϕ, θ, ψ) and the angular velocity in B-frame (p, q, r). Thanks to Eq. (4.1.8) those angular velocities can be easily transformed into Euler-angle rates $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ and a Unknown Input Observer (UIO) as detailed in Section 4.2.3 could be implemented for disturbance rejection.

The model used for validation allows to directly measure the angular velocities (p, q, r) simulating a real scenario. In order to demonstrate the system robustness against disturbances, a system trying to maintain the initial conditions of Table 5.4.1 is disturbed with a $M_x = 0.5\,Nm$ moment in the B-frame X-axis, which destabilizes the system without the disturbance rejection mechanism (see Fig. 5.6.7). The system behaviour with the disturbance rejection can be seen in Fig. 5.6.8, and the obtained disturbance estimation is shown in Fig. 5.6.9.



(a) Position response (b) Attitude response

Figure 5.6.8: Disturbance moment $M_x = 0.5\,Nm$ - Disturbance Rejection

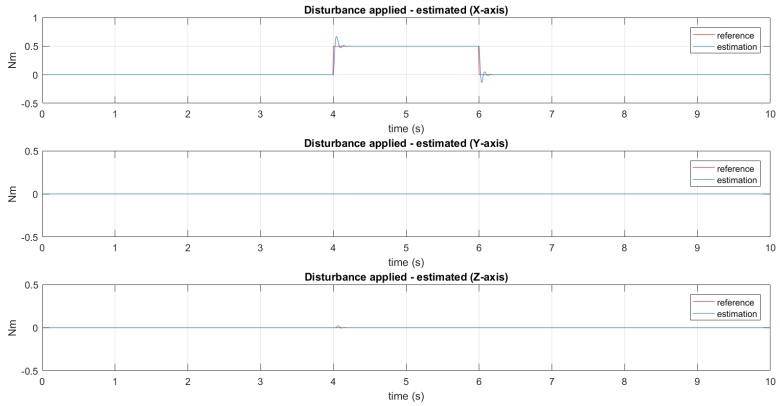
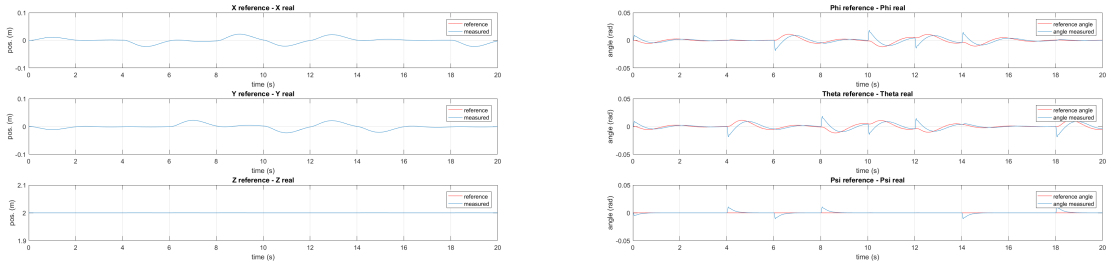


Figure 5.6.9: Disturbance estimation

Aiming to demonstrate the effectiveness of the disturbance rejection mechanism designed, an independent disturbance will be introduced in each one of the B-frame axes. The disturbances introduced will be Pseudo Random Binary Signals (PRBS) varying between $[-0.2\,Nm,\ 0.2\,Nm]$ generated by third order polynomials initialized each one with a different seed. The variation time of the signal is fixed in two seconds. The system response maintaining the initial conditions while subjected to the described disturbance can be seen in Fig. 5.6.10, the estimation of the applied disturbance is shown in Fig. 5.6.11.



(a) Position response (b) Attitude response

Figure 5.6.10: PRBS disturbance moments in each axis

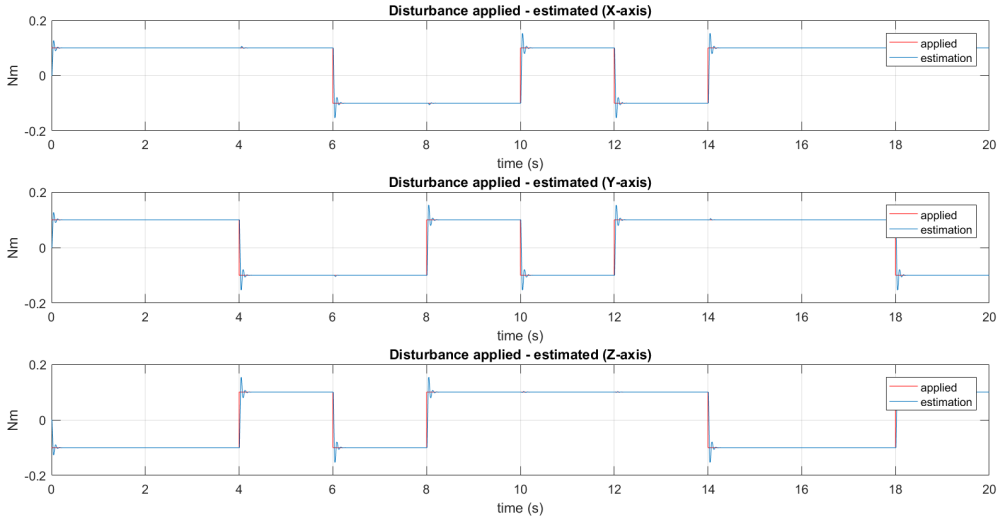


Figure 5.6.11: Estimated PRBS distrubances

5.6.3 Third scenario

As described in Section 5.3, a third scenario for tracking references in B-frame linear velocities is studied. This control strategy is developed in order to meet the guidance commands of some path planners, which instead of generating trajectories through a set of position waypoints, provide a desired velocity vector that the system must track.

It is important to highlight that the modifications with respect to the position control strategies lay only in the outer loop, the inner loop that controls the platform attitude maintains the same LPV-LQR structure as in previous scenarios. In order to demonstrate the performance of the proposed velocity control strategy a simple path generator will be used: The *Carrot Chasing*.

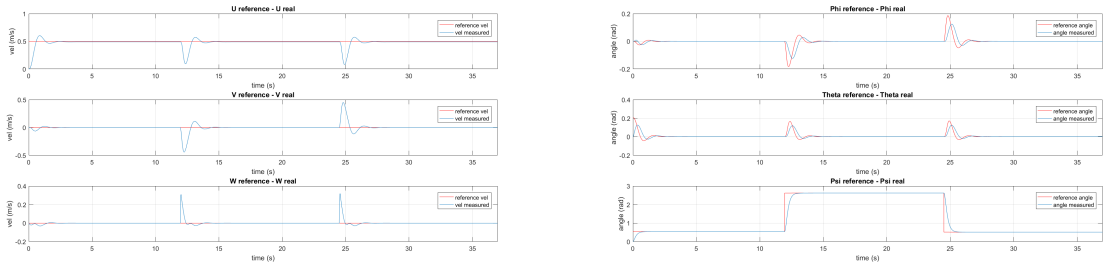
The *Carrot Chasing* algorithm it is a simple path planner that acts in 2D (constant height). Starting in an initial point (x_i, y_i), and for a desired final point (x_f, y_f), the planner assumes a velocity vector pointing in the positive direction of drone's X-axis, so the quadrotor is directed through the yaw angle. That is, in order to reach a goal point, the planner assumes a positive u velocity, and provides the needed yaw angle to reach the goal. Nevertheless, this trajectory planner presents some drawbacks and it only will be used for demonstrating the performance of the velocity control.

The trajectory generated by the path planner will start at the initial conditions fixed in Table 5.4.1, and the goal point will change among the ones shown in Table 5.6.2. Thus, when the drone is inside the 0.1 m circle around the goal point, that goal point will transform into the new initial point and the new goal point will be the next one in the table.

Initial Waypoint	**Goal Waypoint**
$(x_i, y_i) = (0, 0)$	$(x_f, y_f) = (5, 3)$
$(x_i, y_i) = (5, 3)$	$(x_f, y_f) = (0, 6)$
$(x_i, y_i) = (0, 6)$	$(x_f, y_f) = (5, 9)$

Table 5.6.2: Waypoints

For the trajectory generated by the *Carrot Chase* planner in terms of B-frame velocities and yaw angle (ψ_{ref}), the system response following the commands of velocities and angles can be seen in Fig. 5.6.12, while the trajectory that the drone follows WRT E-frame is shown in Fig. 5.6.13. The work developed by [31] also includes a graphical interface as a Simunlink® pop up link,



(a) Velocities response (b) Attitude response

Figure 5.6.12: Carrot chase system response

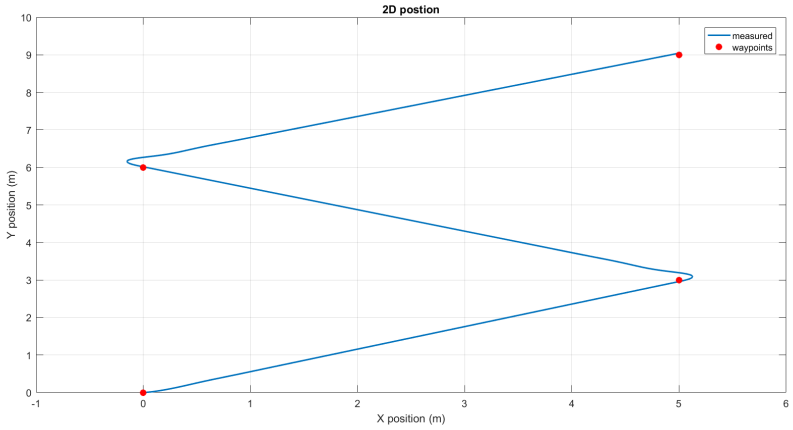


Figure 5.6.13: Carrot chase trajectory obtained

that plots the position and attitude dynamics of the quadrotor once the simulation is finished. Figure 5.6.14 shows the mentioned interface for the Carrot Chase trajectory generated for the Table 5.6.2 waypoints.

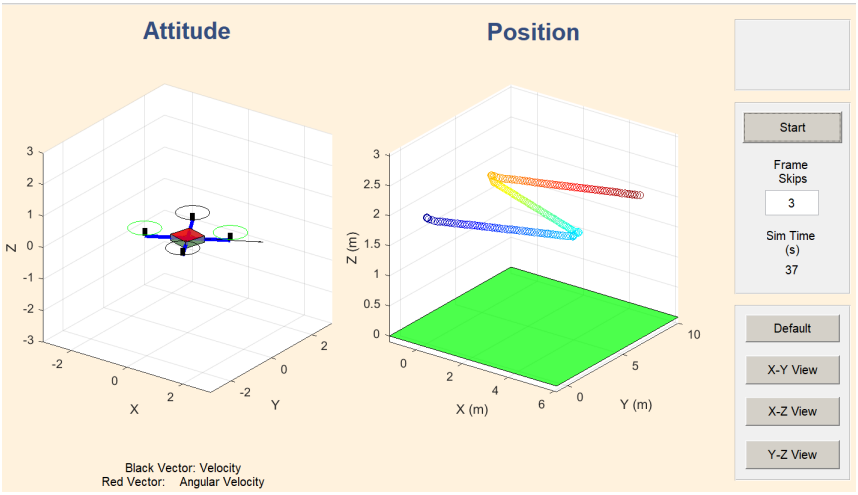


Figure 5.6.14: System dynamics GUI

Chapter 6

Implementation

This chapter presents the steps to follow, as well as the obtained results, in the real implementation of the designed controllers. Due to the lack of time, only the LPV attitude controller was implemented. In order to be able to actually fly the drone, the angle reference commands (ϕ_{ref}, θ_{ref} , ψ_{ref}) were set in such a way that their values can be manually modified through a remote controller, so the user can control the quadrotor position instead. The chapter is organised as follows: in the first sections, a description of the platform operation and the testing bench, is developed. Afterwards, an explanation of the singularities that the actual control implementation entails is provided. Finally, the results obtained in the testing bench are presented, as well as some considerations regarding the real flight test.

6.1 Platform main characteristics

As already said, the target platform for implementing the designed control architectures is an AscTec® Hummingbird, which the UPC Advanced Control Systems group has in their laboratory. Due to its reduced Maximum Take Off Weight (MTOW) the *Hummingbird* is classified as a micro-UAV. Table 6.1.1 shows its main technical specifications detailed in the company web page [32].

The values of all the parameters needed for the proposed model-based controllers were detailed in Section 5.2. Almost all of those parameters, but the aerodynamic coefficients, are also specified in the technical documentation provided by Asctec® *Research*. However, there are small differences between the given ones and the final values used in Table 5.2.1. These small differences are the result of some modifications performed in the quadrotor with respect the original one.

HUMMINGBIRD	
On-board computer	ARM (LPC2146)
Dimensions	$54 \times 54 \times 5.5\,cm$
MTOW	$0.73\,Kg$
MPL	$200\,g$
Max. flight time	$20\,mins$ (no payload)
Range	$4500mASL$
Max. thrust	$20N$
Max. airspeed	$15\,m/s$
Max. ascent speed	$5\,m/s$
Battery	$2100\,mAh$ (LiPo)
Motors	$4 \times 80\,W$ AscTec X-Bl 52s
Propellers	8" $(20.32\,cm)$ diameter
Telemetry	Xbee $2.4\,GHz$
Remote Control	Futaba FAAST $2.4\,GHz$

Table 6.1.1: Hummingbird Technical Data

In order to obtain the final parameter values, several measurements were carried out. Regarding the aerodynamic coefficients c_T and c_q, the values obtained by [33] for the same used platform were validated with some simple experiments, before accepting them as valid.

6.2 Flight Control Unit

The operation of the flight control unit that mounts the Hummingbird, is outlined in the diagram shown in Fig. 6.2.1. This unit contains all necessary sensors to function as an IMU, it also has two onboard ARM7 microprocessors, an Odroid-XU4 onboard PC and various communication interfaces. Below, a brief description of the elements that compound the diagram is made.

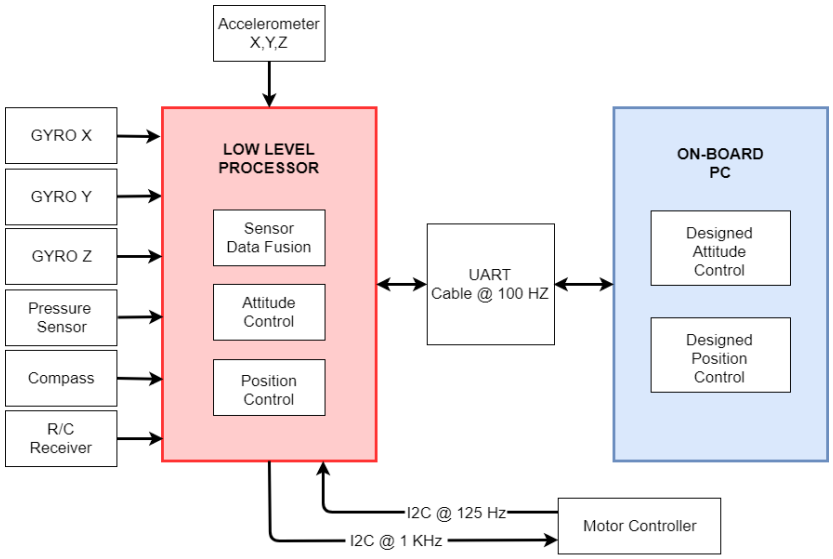


Figure 6.2.1: Flight control unit

6.2.1 Low Level processor

The Low Level processor (LLP) handles sensor data processing, data fusion as well as robust safety attitude control with an update rate of 1000 Hz. In case that the GPS module is available, the LLP also processes the position algorithm.

Furthermore, the processed sensor data can be retrieved from the LLP via a serial interface. In addition, the serial interface can also be used to send attitude commands ir order to control the quadrotor from an external PC.

Figure 6.2.2 shows the *AutoPilot* board provided by AscTec®. The image shows the Low Level processor in charge of the aforementioned actions, as well as the different gyroscopes, accelerometers and other sensors depicted in Fig. 6.2.1.

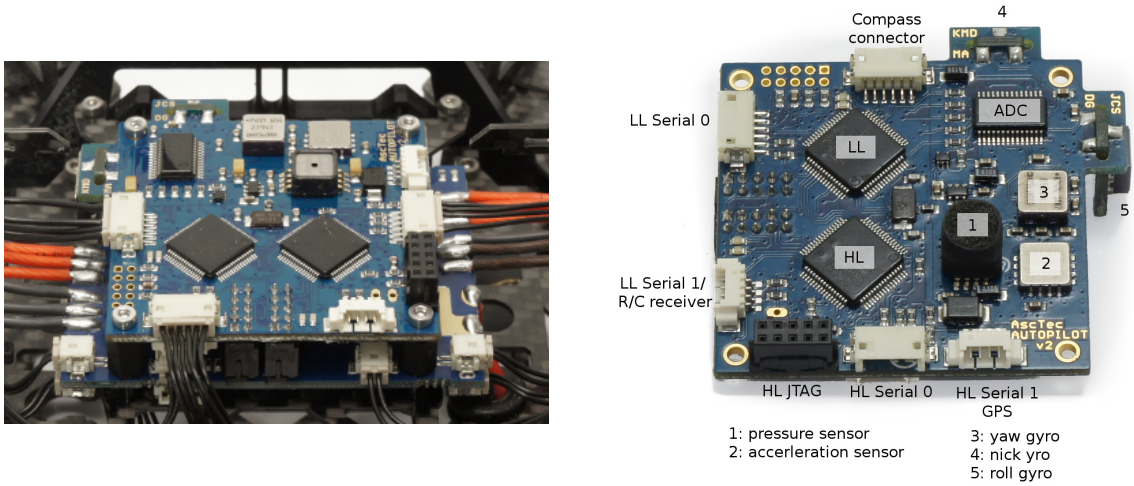


Figure 6.2.2: AscTec AutoPilot Board

Actually, in Fig. 6.2.2 it can be seen a second processor working as High Level processor (HLP) where the user can flash the designed control algorithms. Nevertheless, the functions of this HLP is substituted by the onboard PC described in Section 6.2.2, which communicates with the Low Level processor through the connections specified in Section 6.2.4.

6.2.2 Onboard PC

As previously commented, an external computing device is used for implementing the control algorithms. The laboratory quadcopter mounts an Odroid-XU4, as the one shown in Fig. 6.2.3, running the Ubuntu 14.04 Operating System. The technical specifications of the Odroid board are detailed in [36].

The control algorithm is implemented making use of the well known Robot Operating System (ROS) [37], a flexible framework for writing robot software. This can be done thanks to the different ROS packages released by AscTec® in order to interface with their platforms. These ROS packages allow to easily communicate with the Low Level processor through a set of already defined messages.

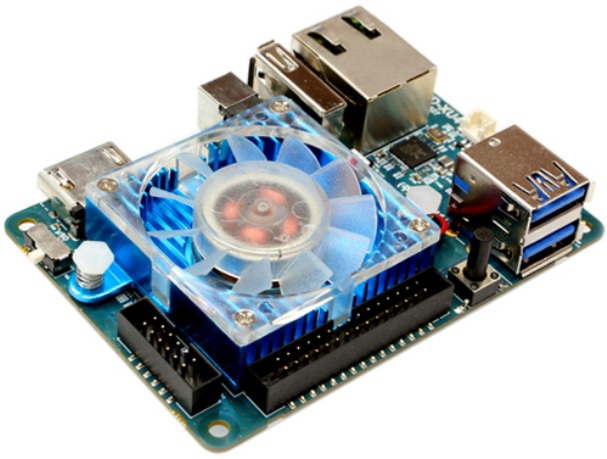


Figure 6.2.3: Odroid Computer

6.2.3 Sensors

Below, a brief description of the platform sensors in developed:

1. Pressure sensor: Located in the AutoPilot Board (see Fig. 6.2.2), allows to estimate the height over the floor through pressure differences.

2. Gyroscopes: Three different gyroscopes for measure the angular velocities in B-frame. The three of them are situated in the AutoPilot Board.

3. Accelerometer: Also located in the AutoPilot Board, a triaxial accelerometer able to measure the accelerations in B-frame is available.

4. Magnetometer: The AscTec 3D-MAG provides the quadrotor heading from measurements of the earth's magnetic field. It is assembled in the structure core.

5. GPS: Originally it is located in the upper part of the quadrotor's structure. The GPS measurements are matched with the inertial sensors onboard, providing a more accurate position estimation. Nevertheless, as all the developed test were carried out indoor, it did not make sense to keep the GPS module and it was removed.

It must be taken into consideration that in the Low Level processor the data fusion is performed, where the data from the different sensors is integrated in order to achieve a better precision. After the fusion, the available data for the controller design is:

- Euler angles (ϕ, θ, ψ)
- Angular velocities in B-frame (p, q, r)
- Position in E-frames coordinates (x_E, y_E, z_E)
- Velocities in E-frame coordinates ($\dot{x}_E$, $\dot{y}_E$, $\dot{z}_E$)

6.2.4 Connections

In order to communicate the Odroid board with the Low Level processor, for reading the sensor measurements or writing the designed control actions, a specific USB → AutoPilot connection cable, like the one in Fig. 6.2.4, is needed. This cable has crimp contacts and 6-pin DF13 jacks.

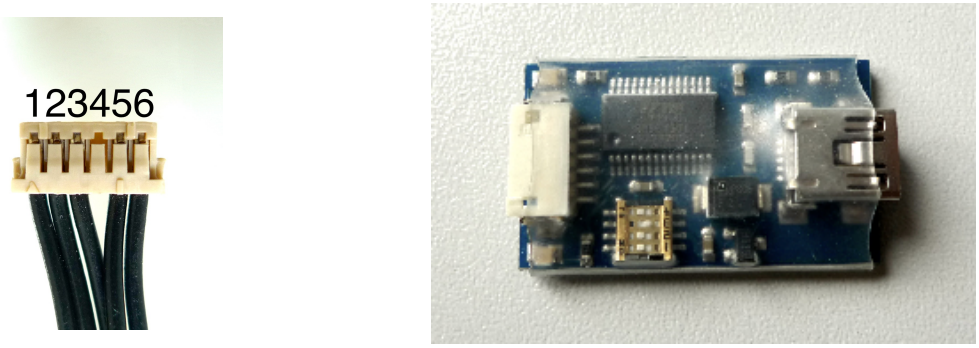


Figure 6.2.4: Connection cable

Figure 6.2.5 shows the connections of the described components in the real platform.

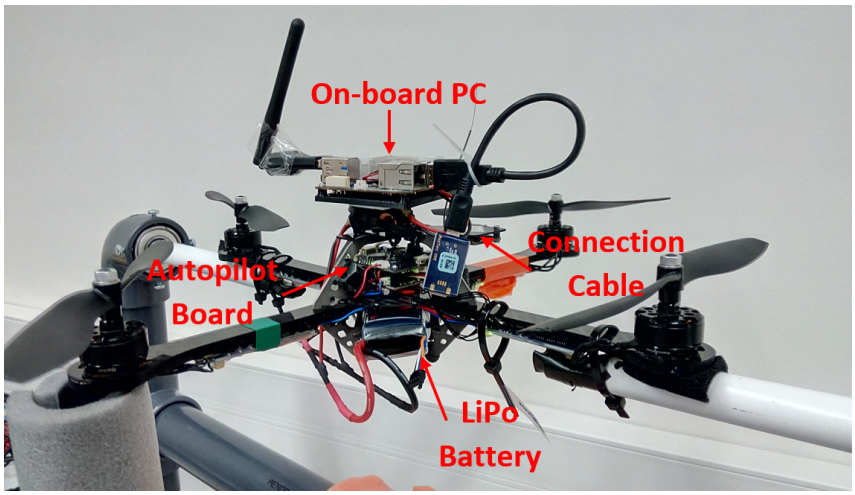


Figure 6.2.5: Hummingbird Connections

6.3 Testing bench

In order to safely test the performance of the designed attitude control algorithms, the Advanced Control Systems research group has in the laboratory a testing bench like the one shown in Fig. 6.3.1. This structure allows the quadrotor to freely spin around its B-frame X-axis, or Y-axis, in such a way that the performance of the roll angle, or pitch angle, can be tested without inflicting any harm to people, nor the platform.

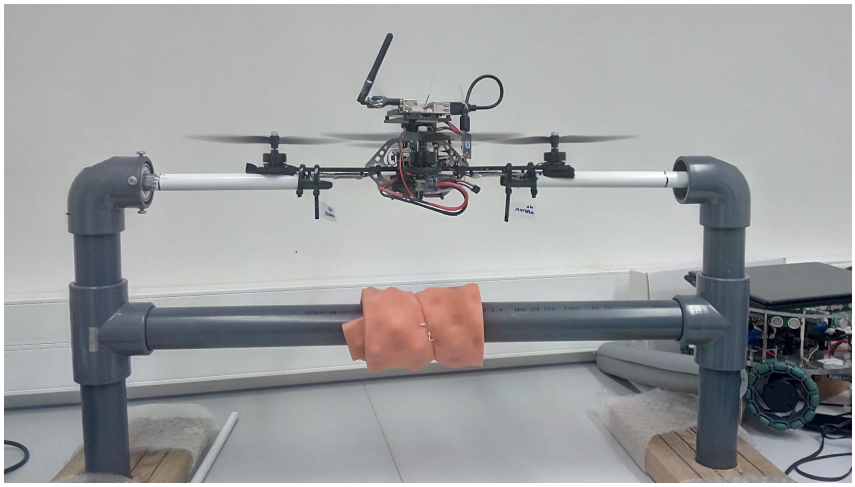


Figure 6.3.1: Attitude Control Testing Bench

6.4 Controller structure

6.4.1 Control Mixing

As shown in Fig. 6.2.1, the control commands generated by the Onboard PC must go through the Low Level processor (LLP) before actuating over the motor controllers. However, in the real implementation, the LLP does not allow to specify the control actions as motor speeds. Inside it, there is implemented a control mixing block, that relates what it is called a set of corrections over the angles and thrust, with the required motor speeds.

Hence, the data message transmitted by the designed controller to the LLP, with the required control action, must be sent in terms of those corrections that the control mixing block accepts. Those corrections that the LLP ask for, do not have physical meaning and are mainly designed for implementing classical controllers, i.e. using empirical transfer functions.

In order to deal with the problem of providing the appropriate control commands to the LLP,

it was decided to model the operation of that unknown control mixing block. The main idea behind it is the following: the designed control algorithm provides the required motor speed, as those motor speeds can not be directly addressed, the control mixing inputs that would provide those motor speeds are computed using the aforementioned model and sent to the LLP, which through the control mixing will transform in the desired motor speeds.

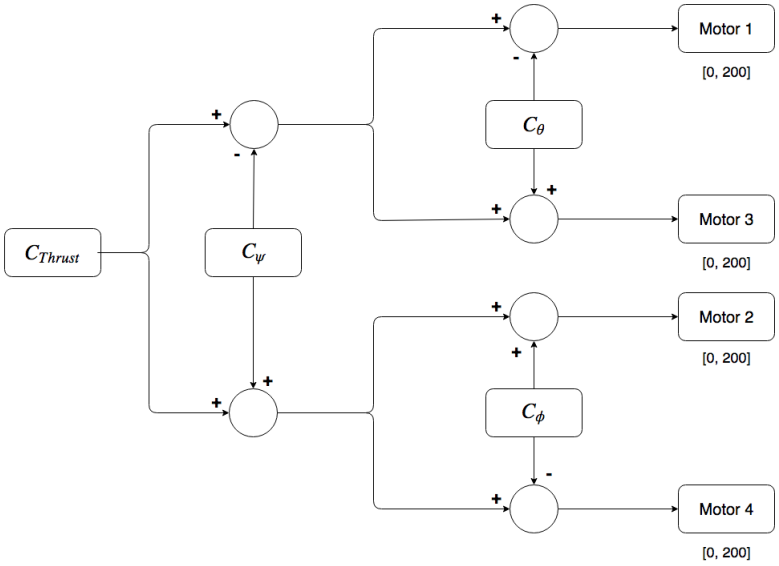


Figure 6.4.1: Control Mixing diagram

Recording the control corrections applied by a well proven PID controller, and the measures of the obtained motor speeds, it was possible to model the already implemented control mixing block as shown in Fig. 6.4.1. The available control commands, as well as their range, are expressed in Table 6.4.1. This control mixing inputs are related with the commands received from the remote controller in a manual flight, in such a way that for the angles corrections (C_ϕ, C_θ, C_ψ), the application of null moment maps with a value of 100. The system also maps the motor speed values in the range [0, 200], its relationship with the rotational speed in rpm is given by Eq. (6.4.1).

$$motorRPM = c_r \cdot m + b = 37.625 \cdot m + 1075 \tag{6.4.1}$$

Control Command	**Range**
Roll Correction - C_ϕ	[0, 200]
Pitch Correction - C_θ	[0, 200]
Yaw Correction - C_ψ	[0, 200]
Thrust Correction - C_T	[0, 200]

Table 6.4.1: Control Mixing inputs

The identified control mixing model can be expressed in matrix form as follows:

$$\begin{bmatrix} \text{Motor } 1_{speed} \\ \text{Motor } 2_{speed} \\ \text{Motor } 3_{speed} \\ \text{Motor } 4_{speed} \end{bmatrix} = \begin{bmatrix} 0 & 1 & -1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} C_{\phi 2} \\ C_{\theta 2} \\ C_{\psi 2} \\ C_T \end{bmatrix} \tag{6.4.2}$$

where $C_{\phi 2} = 100 - C_\phi$, $C_{\theta 2} = 100 - C_\theta$ and $C_{\psi 2} = 100 - C_\psi$.

Thus, for a desired set of motor speeds in rpm, these are ranged in [0, 200] through Eq. (6.4.1), and using the inverse of the relation shown in Eq. (6.4.2), the set of correction commands (C_ϕ, C_θ, C_ψ, C_T) that sent to the Low Level processor would achieve the desired motor speeds are obtained.

6.4.2 Attitude controller implementation

The logical order for implementing the proposed cascade control scheme is to start with the inner loop: the attitude controller. In addition, as the GPS module was not even installed, all the tests performed throughout this project were carried out indoors, aiming to check the attitude response of the system. It is important to highlight that the main control challenge lays in the attitude control due to the high instability that the system presents, rather than in the position control which allows a slower rate.

When it comes to the implementation of the attitude controller, as described in Section 6.2.3, the Low Level processor provides information of the Euler angles (ϕ, θ, ψ) and the angular velocities in B-frame (p, q, r). Thus, making use of Eq. (5.1.1), the Euler angle rates ($\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$) are obtained from the angular velocities. Therefore, all the states of the attitude subsystem: $x_1 = \phi$, $x_2 = \dot{\phi}$, $x_3 = \theta$, $x_4 = \dot{\theta}$, $x_5 = \psi$, $x_6 = \dot{\psi}$, are available with a sample rate of $100\,Hz$.

The implemented controller is the LPV-LQR state feedback controller developed in Section 4.2.2, with the tuning parameters specified in Table 5.5.2. As the data fusion performed in the Low Level Processor, gives reliable measures of all the states, the designed observer is not implemented. Besides, aiming to speed up the algorithm running time, the designed feed-forward LPV reference tracking matrix $F(t)$, is approximated generating a polytope with its extreme values and performing a scheduling as in the controller gain case. Although this does not hold

true, as the inverse DC-gain mapping of the system is not linear, this method provides a reasonable approximation for changing the reference command for small values.

The main weakness of the implemented algorithm is its sensitivity to the model, that is: the existence of small unmodeled moments acting in the quadrotor causes the system not to reach exactly the zero error in the steady state. One way to solve this issue could be the extension of the considered states, including also three new states with the integral action of the angles error.

However, in order to solve this problem for the proposed structure in Section 4.2.2, and being able to accomplish a safe flight, the roll and pitch angle commands were linked with the remote control joysticks. Thus, if during flight unexpected effects tilt the quadrotor from the hover position, the user can manually counteract that tendency changing the reference angle.

6.4.3 Control algorithm

The LPV attitude controller returns information of the (U_2, U_3, U_4) moments [Nm]. Thus, in order to obtain the required motor speeds for achieving those moments the information of the applied thrust U_1 [N] is required. This thrust command is given directly by the user through the remote control, and the read values are transformed into the applied force needed for computing the resulting control actions.

As previously stated, with the obtention of the four actions (U_1, U_2, U_3, U_4), the required motor speeds are computed. And through the inverse model of the LLP control mixing block, the applied control actions (C_ϕ, C_θ, C_ψ, C_T) that yield those motor speeds are derived. Algorithm 1 summarizes the main steps for implementing the designed LPV-LQR controller.

Algorithm 1 LPV-LQR Attitude control

Inputs:

$q_{att} = [q_w\, q_x\, q_y\, q_z]^T$: current Attitude quaternion

p: current X-axis angular velocity

q: current Y-axis angular velocity

r: current Z-axis angular velocity

ϕ_{ref}: desired roll angle

θ_{ref}: desired pitch angle

ψ_{ref}: desired yaw angle

$Thrust$: Thrust command

Outputs:

C_ϕ: applied correction in roll angle

C_θ: applied correction in pitch angle

C_ψ: applied correction in yaw angle

C_T: applied correction in thrust

while `control enabled` **do**

1. Transform measured quaternion to Euler angles
2. Transform measured angular velocities to Euler angle rates
3. Obtain the scheduling variable $\mathbf{\Omega}$
4. Read from the controller the (ϕ_{ref}, θ_{ref}, ψ_{ref}, U_1)
5. Get the scheduling weights π_i
6. Compute the scheduled Gain matrix K
7. Compute the state feedback and obtain (U_{2K}, U_{3K}, U_{4K})
8. Compute the scheduled Feed-forward matrix F
9. Compute the feed-forward action and obtain (U_{2F}, U_{3F}, U_{4F})
10. Add U_{iK} to U_{iF} for obtaining U_i
11. Compute the required motor speed from (U_1, U_2, U_3, U_4)
12. Compute the required (C_ϕ, C_θ, C_ψ, C_T)

6.5 Real implementation results

The results presented in Sections 6.5.1 to 6.5.2 were recorded stabilizing the pitch dynamics of the quadrotor in the testing bench described in Section 6.3. It is important to highlight the fact that the testing bench gears have a certain slack, inducing an unknown torque over the studied angle dynamics. This torque can even vary its magnitude and orientation during the test, pushing the drone to one side or the other indistinguishably. The unknown moment induced by the bench structure implies that each test was carried out under different disturbances,

causing the quadrotor to stabilize over a different angle each time, as mentioned in Section 6.4.2.

For the results shown in those Sections, a thrust force close to the one needed to counteract the quadrotor weight is applied through the remote control, firstly stabilizing the platform with the predefined controller implemented in the Low Level processor. Once the drone is stabilized in the hover attitude, through a small lever in the remote control, the controller is manually switched to the designed LPV-LQR attitude controller.

The exact moment of the controller switch, from the defined in the LLP to the designed one, is highlighted in the following graphs through a vertical black dotted line.

6.5.1 Hovering

Setting the pitch reference $\theta_{ref} = 0$, the obtained pitch angle is depicted to the right of the vertical black line in Fig. 6.5.1. It can be seen how the discrepancies between the model and the real platform affect the controller performance.

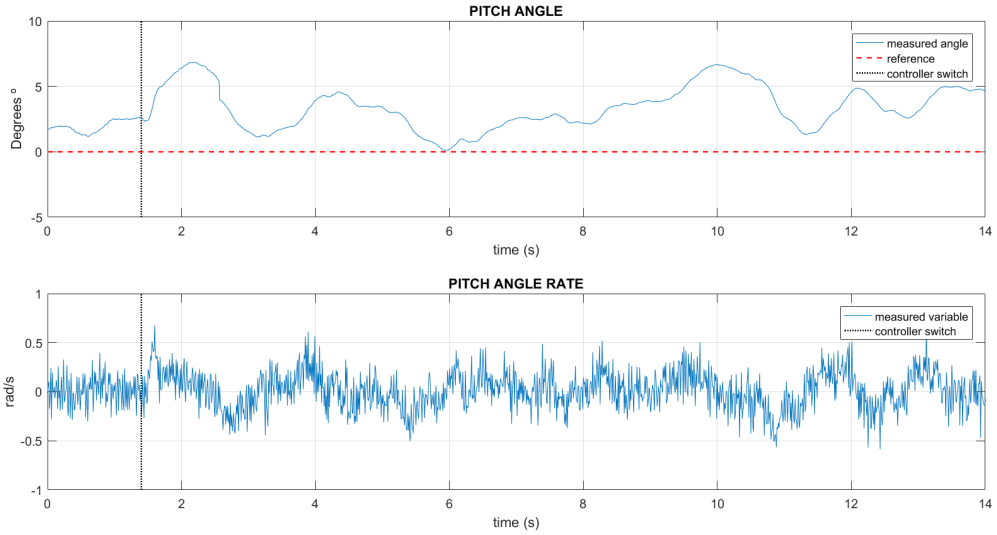


Figure 6.5.1: Pitch angle θ - Hover reference

One fundamental difference between the predefined LLP controller (to the left of the black line), and the designed one, is the updating rate. While the designed one is limited at 100 Hz due to the sampling rate provided via serial communication by the LLP, the predefined one runs at 1.000 Hz. The controllability of the system demonstrated to be really dependent of the updating rates, being unable to control the platform attitude until a frequency higher than 50 Hz was

achieved optimizing the code structure. Thus, the higher the controller frequency, the better the performance.

The motor speeds decrease in the front and rear motors, and the increase in the left and right ones, with respect the predefined controller, seen in Fig. 6.5.2; is due to the fact that in the designed controller the yaw correction is not applied as shown in Fig. 6.5.3, while the predefined one is trying to correct the yaw angle inducing a moment over the B-frame Z-axis.

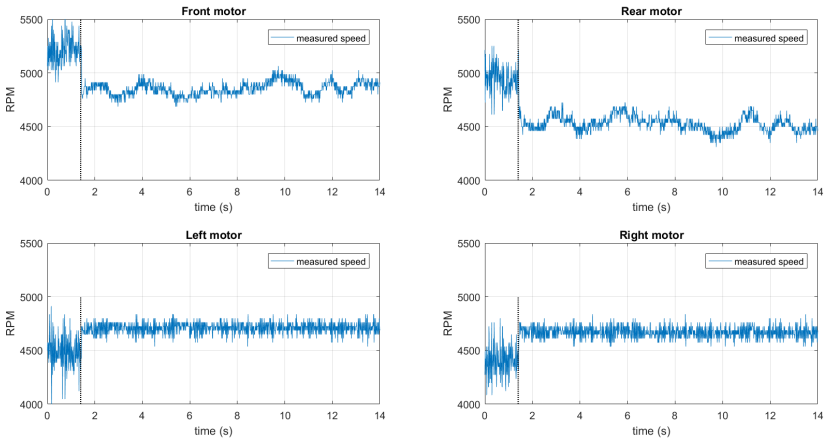


Figure 6.5.2: Motor speeds (rpm) - Hover reference

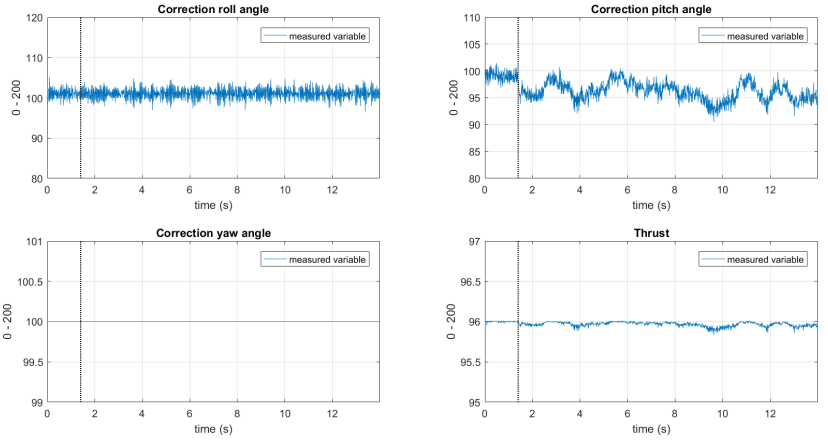


Figure 6.5.3: Applied control actions - Hover reference

6.5.2 Reference change

Here the pitch angle reference is set to $\theta_{ref} = -25°$, in such a way that the platform is stabilized over the hover with the predefined controller, and when manually switching to the designed LPV-LQR controller, a step in the reference of 25° is generated as shown in Fig. 6.5.4. The

mismatch between the $\theta_{ref} = -25°$ and the obtained one: $\theta \simeq -37.5°$, is due to two main reasons:

1. The presence of unmodelled moments inducted by the testing bench
2. The approximation of the feed-forward matrix $F(t)$, as explained in Section 6.4.2, performing gain scheduling over its extreme values, instead of actually computing the inverse of the DC-gain in each moment.

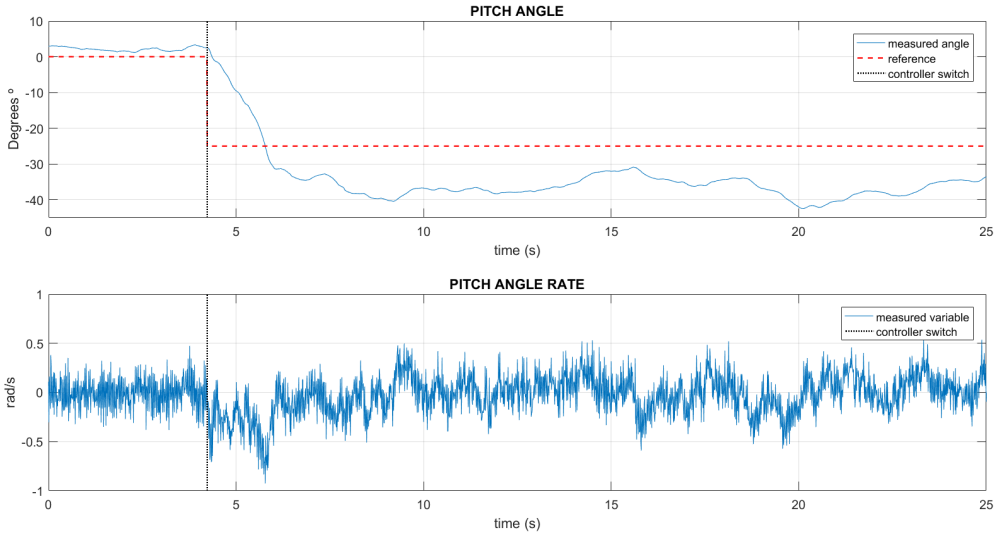


Figure 6.5.4: Pitch angle θ - Reference $-25°$

In Fig. 6.5.5 it is shown how the motor speeds are modified with the reference change, in such a way that the rear motor increases its speed over the hover rotational speed, and the front motor decreases it, for achieving the desired negative pitch angle.

6.5.3 Real flight

Finally, after demonstrate the correct performance of the designed controller in the testing bench, the quadrotor was tested in real flight conditions. As explained in Section 6.4.2, in order to control the quadrotor position, the user can manually change the roll (ϕ_{ref}) and pitch (θ_{ref}) reference angles through the remote controller.

During the tests, the magnetometer used for determining the heading, yaw angle (ψ), returned uncertain measures, with huge differences between samples. These variations in the magnetometer measures, were mainly due to the fact that all the tests were carried out indoors. In order

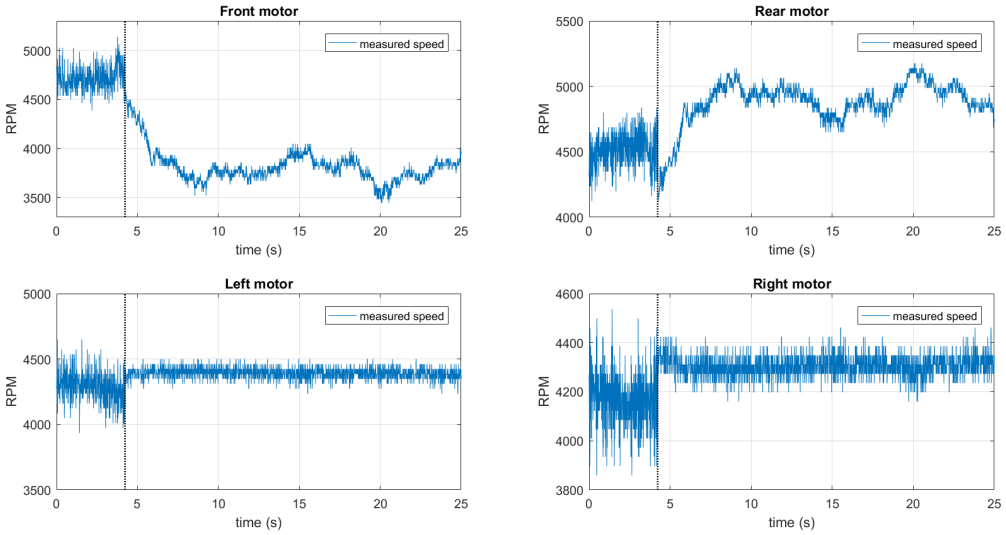


Figure 6.5.5: Motor speeds (rpm) - Reference $-25°$

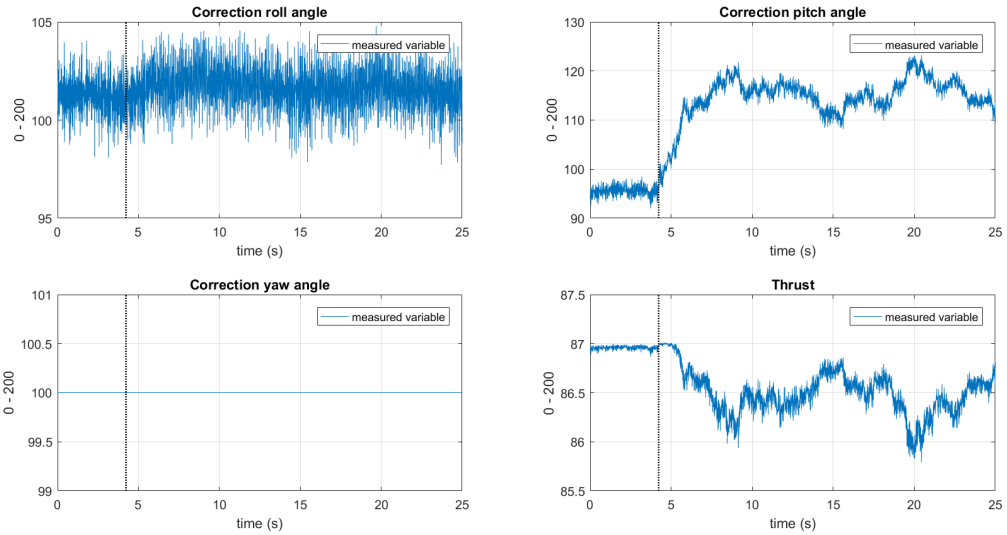


Figure 6.5.6: Applied control actions - Reference $-25°$

to solve that problem, and safely fly the quadrotor, the yaw control was disabled, and the yaw control commands (C_ψ), were directly read from the remote controller. Thus, the user could correct the heading angle facing unexpected situations.

Figure 6.5.7 shows the *Hummingbird* quadrotor during the performed flight tests.

Figure 6.5.7: Real flight

6.5.4 Conclusions of the real implementation

The real test probes the operation of the LPV attitude controller. Nevertheless, the control design has a total dependence on the model. Thus, the contribution of unmodelled effects such as: the assembly of the onboard PC or the moments generated by the testing bench, causes a degradation in the controller performance.

Another relevant issue is the updating rate. For the results shown in Sections 6.5.1 to 6.5.3, the updating rate was 100 Hz. The controllability of the system is completely dependent of the control frequency, being unable to control the system for updating rates below 50 Hz. Modifying the Low Level processor settings, sensor measurements could be obtained with a rate up to 1.000 Hz, allowing to speed up the actuating rate aiming for a better performance.

Chapter 7

Effects on economy, society and environment

In this Chapter, an introduction to possible impacts on economy, society and environment of high performance autonomous aerial vehicles are presented.

7.1 Socioeconomic impact

As stated in Chapter 1, the latest european and national legislations bet for a progressive integration of Unmanded Air Vehicles and humans. The reduced prices of the needed components, and kind regulations, have provoked the arising of countless start-ups that explode all the possible uses of this technology such as: prescision agriculture, industrial inspections, security, etc. This push from the small companies has forced the big aeronautical ones like Boeing or Airbus, to invest in quadcopters and other multirotors, although with a more ambitious goals [38]. Furthermore, big companies with huge facilities are creating new departments with their own drone fleet looking for cost reduction in their facilities inspection.

This trend to use the drones not as a simple hobby, but as a precision tool that could reduce human risk in specific tasks, is pushing the market to advance. New jobs are created every week in the flourishing drone industry, at the same time that their integration in the industrial environment could directly imply a reduction of costs and risks in tedious tasks. Therefore the development of this project arises in order to meet this market trend. Aiming to explore the capabilities of the available platforms, trying to improve their performance and robustness in unknown scenarios.

7.2 Environmental impact

The more tasks electrical powered drones substitute solid fuel aircraft's, the better environmental impact the development of this technology will have. Hence, if for a daily activity like the surveillance of a crowd, the security corps instead of using conventional helicopters, start to implement the drone technology [39], this would translate in a reduction in carbon emissions, and a noise reduction benefiting the surrounding inhabitants.

Although the introduction of electric motors in the transport industry could imply a great benefit for the planet, it also has its negative part. The consequent boom in batteries demand, leads to a new environmental challenges, like the exploitation of lithium resources [40].

Chapter 8

Project budget

The aim of this Chapter is double: on one side, the estimation of the project development cost as conducted in this thesis, and on the other side give an approximation of the unitary cost as if a company intend to commercialise the controller as a finished good. For that purposes the following costs are taken into consideration:

Hardware Costs

- Cost of the embedded computer in which the algorithms run (C_{PC}): 100 €
- Cost of adapting the platform (modifying the structure to include the onboard PC) (C_{adap}): 80 €

Development Costs

- Software licenses (C_{lic}): 6.000 €
- Algorithms development (C_{dev}):

  (Jr. engineer salary per hour) × (hours devoted to the project) = 15 € /h × 600 h = 9.000 €
- Technical support (C_{Ts}):

  (expert salary per hour) × (hours devoted to the project) = 40 € /h × 60 h = 2.400 €
- Testing Bench (C_{Tb}): 1.000 €
- Laboratory material (C_{lab}): 3.000 €
- General costs: electricity, water, others. ($C_{General}$): 2.000 €

Regarding the previous costs, it can be estimated that developing the project with the equipment already available in the university has an estimated cost of:

$$\text{Project development cost} = C_{lic} + C_{dev} + C_{Ts} + C_{General} = 19.400\,€$$

For a product consisting on a high performance controller that improves the quadcopter performance, which can be integrated with already existing drones. The unitary cost from a company point of view is estimated as follows:

$$\text{Product unitary cost } C_{unit} = C_{PC} + C_{adap} + \frac{C_{lic}+C_{dev}+C_{Ts}+C_{Tb}+C_{lab}+C_{General}}{n_{drone}}$$

where n_{drone} is the number of platforms equipped with this controller. For a n_{drone} of 500 quadrotors the resulting unitary cost is:

$$C_{unit} = 100\,€ + 80\,€ + \frac{6.000\,€+9.000\,€+2.400\,€+1.000\,€+3.000\,€+2.000\,€}{500} = 226.8\ €/\text{unit}$$

The equipment of this new system would have a moderate impact in a quadrotor price. Nevertheless, it is difficult to estimate the real unitary cost. In order to do a more accurate forecast, a thorough cost analysis would have to be done.

Chapter 9

Concluding remarks

9.1 Conclusions

In this thesis, the Unmanned Air Vehicle problem was addressed from the control point of view. Even though the first idea was only to work with the position control, the strong dependence with respect the attitude subsystem forced us to consider the control of the attitude-position six degrees of freedom problem. The presence of six outputs, and only four control actions, presents an underactuated control problem that was addressed through a cascade control scheme. Out of the six variables, it was decided to control the overall position (x, y, z) and the yaw angle (ψ), while the roll (ϕ) and the pitch (θ) angles are freely determined by the outer position controller.

A simplified mathematical model of the quadrotor's dynamics was obtained for design purposes. Decoupling the rotational and translational dynamics into two subsystems, two different non-lineal control techniques were applied: a Linear Parameter Varying (LPV) approach, whose vertex values were obtained minimizing the Linear Quadratic Regulator (LQR) problem, for controlling the attitude subsystem, and a feedback linearization approach in order to deal with the position subsystem non-linearities. Also a disturbance rejection mechanism was designed for the attitude control, where the system is more vulnerable to external disturbances.

Afterwards, the following step was to test the proposed controller in simulation using, Matlab/ Simulink®, where the obtained results were satisfactory. Once the controller was validated in simulation, the inner attitude loop was implemented in a real platform, achieving stability despite some degradation in the performance with respect the simulation results.

Some aspects to be remarked are listed below:

- Simulation is a really useful tool in the development of control algorithms. In the simulation stage only Matlab® was used. It represents a powerful tool for running simple tests, tuning the controllers, and for debugging due to its simplicity. However, it would have been better to back up the validation stage with another platform that allows to simulate more realistic physical effects like wind, the ground effect, etc. Aiming to reduce the gap between simulation and real world implementations.

- Although in this thesis the whole attitude-position control problem was addressed, in the drone research field it is considered that the main control problem lays on the attitude problem rather than in the position one. The difficulty of the attitude control was suffered specially in the real implementation due to the high instability that the system presents.

- The aforementioned instability provokes a great sensitivity to the control update rate for the attitude controller, although is not that relevant in the position problem. In the tests performed with the real platform, it was observed that it was impossible to control its attitude for control rates lower than 50 Hz. In order to achieve those high rates the code had to be redefined for avoiding time consuming computations.

- The proposed control algorithms are completely based on the derived model, admitting a short parameter uncertainty. Thus, it order to carry out the real implementation it was mandatory to obtain a reliable set of parameters that were valid for the target platform.

- Despite the effort devoted validating the algorithms in simulation, there is always a huge leap between simulation and real implementation. Although some problems, like the unavailability of sending the control commands as designed, could be solved, periodically some unexpected problems appears that slowed down the whole procedure for days.

9.2 Future work

During the development of this master thesis a lot of ideas for improving the controller have emerged, some of them are presented down below:

- To expand the considered states in the LPV attitude controller, including the angles integral error for correcting the steady-state errors that appear in the real implementation.

- To correctly program the feed-forward matrix for reference tracking, instead the approximation adopted in the real implementation.

- To modify the Low Level processor for being able to directly address the motor speeds, instead the correction commands sent now.
- The assembly of a GPS module in order to test the position control algorithms.
- To solve the magnetometer problems for implementing the yaw angle control in the real flight tests.
- To modify the Low Level processor asking for a higher sample rate, aiming to be able to achieve a higher update rate.
- To try or develop a more realistic simulation environment for validation purposes.
- To study in depth the Path Planning problem for Unmanned Air Vehicles.

Bibliography

[1] *Goldman Sachs drone market size - http://www.goldmansachs.com/our-thinking/technology-driving-innovation/drones/.*

[2] *SESAR Corus Project - https://www.sesarju.eu/projects/corus.*

[3] *Boletín Oficial del Estado - https://www.boe.es/boe/dias/2017/12/29/pdfs/BOE-A-2017-15721.pdf.*

[4] *Ardupilot Main Page - http://ardupilot.org/.*

[5] *Amazon Prime Air Delivery - https://www.amazon.com/Amazon-Prime-Air.*

[6] *RCToys. - http://www.rctoys.com.*

[7] V. Moreau N. Guenard, T. Hamel. Dynamic modeling and intuitive control strategy for an "x4-flyer". *IEEE International Conference on Control and Automation*, 2005.

[8] M. Shantz I. Kroo, F. Prinz. The mesicopter: A miniature rotorcraft concept. Technical report, Standford University, 2001.

[9] L. Beji K. M. Zemalache and H. Marref. Control of an under-actuated system: Application to a four rotors rotorcraft. *IEEE International Conference on Robotics and Biomimetics*, page 404 – 409, 2005.

[10] G. Fay. *Derivation of the aerodynamic forces for the mesicopter simulation.*

[11] J. Stumpf M. Achtelik K. M. Doth G. Hirzinger D. Gurdan and D. Rus. Energy-efficient autonomous four-rotor flying robot controlled at 1 khz. *IEEE International Conference on Robotics and Automation*, page 361 – 366, 2007.

[12] A. Dzul P. Castillo and R. Lozano. Real-time stabilization and tracking of a four-rotor mini rotorcraft. *IEEE Transaction on Control System Technology*, page 510 – 516, 2004.

[13] A. Noth S. Bouabdallah and R. Siegwart. *Pid vs lq control techniques applied to an indoor micro quadrotor.*

[14] Y. Morel and A. Leonessa. Direct adaptive tracking control of quadrotor aerial vehicles. *Florida Conference on Recent Advances in Robotics*, page 1– 6, 2006.

[15] T. Madani and A. Benallegue. Backstepping control for a quadrotor helicopter. *Proceedings of 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, page 3255 – 3260, 2006.

[16] A. Mokhtari and A. Benallegue. Dynamic feedback controller of euler angles and wind parameters estimation for a quadrotor unmanned aerial vehicle. *Proceedings of the 2004 IEEE International Conference on Robotics and Automatio*, page 2359 – 2366, 2004.

[17] T. Hamel N. Guenard and R. Mahony. A practical visual servo control for a unmanned aerial vehicle. *2007 IEEE International Conference on Robotics and Automation*, page 1342 – 1348, 2007.

[18] J. P. Ostrowski E. Altug and C. J. Taylor. Quadrotor control using dual camera visual feedback. *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, page 4294 – 4299, 2003.

[19] C. Coza and C. J. B. Macnab. *A new robust adaptive-fuzzy control method applied to quadrotor helicopter stabilization*, 2006.

[20] M. Tarbouchi J. Dunfied and G. Labonte. Neural network based control of a four rotor helicopter. *IEEE Intrnational Conference on Industrial Technology*, page 1543 – 1548, 2004.

[21] G. M. Hoffmann J. S. Jang S. L. Waslander and C. J. Tomlin. *Multi-agent quadrotor testbed control design: Integral sliding mode vs. reinforcement learning.*

[22] M. A. Garratt F. Santoso and S. G. Anavatti. State-of-the-art intelligent flight control systems in unmanned aerial vehicles. *IEEE Transcations on Automation science and Engineering*, 2017.

[23] O. Lopez J. Meseguer A. Sanz A. Cuerva, J.L. Espino. *Teoria de los Helicopteros.* Garceta, 2013.

[24] Tommaso Bresciani. Modelling, identification and control of a quadrotor helicopter. Master's thesis, Lund University, 2008.

[25] T. Perez. *Ship Motion Control:Course Keeping and Roll Stabilisation using Rudder and Fins.* Springer-Verlag, 2005.

[26] D. Rotondo. *Advances in Gain-Scheduling and Fault Tolerant Control Techniques*. Springer, 2017.

[27] E. Ostertag. Mono-and multivariable control and estimation: Linear, quadratic and lmi methods. *Springer Science & Business Media*, 2011.

[28] R. D. Braatz J. G. VanAntwerp. A tutorial on linear and bilinear matrix inequalities. *Journal of Process Control*, pages 363–385, 2000.

[29] E. Alcala V. Puig J. Quevedo T. Escobet. Gain scheduling lpv control for autonomous vehicles including friction force estimation and compensation mechanism. *IET Research Journals*, 2015.

[30] Hai-Hua Yu Guang-Ren Duan. *LMIs in Control Systems: Analysis, Design and Applications*. CRC Press, June 17, 2013.

[31] M. Mehrer S. Moreno D. Hartman, K. Landis and Dr. B. C. Chang J. Kim. Quadcopter dynamic modeling and simulation. *Drexel University*, https://github.com/dch33/Quad-Sim.

[32] *Asctec Research Home - http://wiki.asctec.de/display/AR/AscTec+Research+Home.*

[33] Pablo Sanchez Prieto. Estudio del control de un uav multirotor asctec hummingbird. Master's thesis, Universitat Politècnica de Catalunya, 2015.

[34] A. El Hajjaaji M. Belkheiri, A. Rabhi and C. Pegard. Different linearization control techniques for a quadrotor system. *IEEE Journal of Robotics and Automation*, 2012.

[35] Holger Voos. Nonlinear control of a quadrotor micro-uav using feedback-linearization. *University of Applied Sciences Ravensburg-Weingarten*, 2014.

[36] *Hardkernel Main Page - http://www.hardkernel.com/main/main.php.*

[37] *ROS Main Page - http://www.ros.org.*

[38] *Airbus drone concept - http://www.airbus.com/newsroom/press-releases/en/2018/04.*

[39] *La Vanguardia - http://www.lavanguardia.com/local/barcelona/20180222/44977060110/los-mossos-vigilaran-el-mwc-con-drones.html.*

[40] B. M. Zimmermann J. F. Peters, M. J. Baunmann and M. Weil. The environmental impact of li-ion batteries and the role of key parameters - a review. *Renewable and Sustainable Energy Reviews*, pages 491 – 506, 2017.