Treball de Fi de Màster

Master's degree in
Automatic Control and Robotics

Gain-scheduling Control of a Quadrotor Using the Takagi-Sugeno Approach

MEMÒRIA

Autor: Diego Ruiz Paz
Director/s: Vicenç Puig Cayuela

Convocatòria: Juny 2016

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

In this thesis, a Takagi-Sugeno model for an Attitude/Altitude model of a quadrotor system is developed. With this Takagi-Sugeno model, a gain-scheduling state-feedback controller, as well as a state observer, have been designed for altitude and orientation control. Then, two different control schemes are analyzed in order to have a good performance on tracking changing references in the Attitude/Altitude control. Then an Integral Backstepping controller has been designed for the control of horizontal position. The stability of the whole control system has been studied, and finally the models and controllers have been tested in a simulation environment.

Acknowledgements

I would like to thank my advisor Prof. Vicenç Puig for all the support, advices but mainly for his patience.

Contents

List of Figures

Acronyms and Abbreviations

AA model	**A**ttitude/**A**ltitude model
B-frame	**B**ody reference frame
E-frame	**E**arth reference frame
FF	**F**eed**F**orward control
H-frame	**H**ybrid reference frame
IB	**I**ntegral **B**ackstepping control
LMI	**L**inear **M**atrix **I**nequality
LTI	**L**inear **T**ime **I**nvariant
PDC	**P**arallel **D**istributed **C**ompensation
RM-FF	**R**eference **M**odel based **F**eed**F**orward control
TS	**T**akagi-**S**ugeno
TS-LIA	**T**akagi-**S**ugeno with **L**inear **I**nput **A**pproximation
UAV	**U**nmanned **A**erial **V**ehicle
VTOL	**V**ertical **T**ake-**O**ff and **L**anding

Constants

Symbol	Description	Value
b	Thrust factor	$54.2 \times 10^{-6}\ Ns^2$
d	Drag factor	$1.1 \times 10^{-6}\ Nms^2$
g	Acceleration of gravity	$9.81\ ms^{-2}$
l	Distance from the quadrotor's center to one propeller	$0.24\ m$
m	Mass of the quadrotor	$1\ kg$
I_X	Principal moment of inertia around X_B	$8.1 \times 10^{-3}\ Nms^2$
I_Y	Principal moment of inertia around Y_B	$8.1 \times 10^{-3}\ Nms^2$
I_Z	Principal moment of inertia around Z_B	$14.2 \times 10^{-3}\ Nms^2$
J_{TP}	Rotational moment of inertia around the propeller axis	$104 \times 10^{-6}\ Nms^2$
Ω_H	Angular speed of the propellers in hovering condition	$212.7\ rad\ s^{-1}$

Symbols and Variables

Symbol	Description	Unit
$\boldsymbol{e}_{est}$	State estimation error. $\boldsymbol{e}_{est} = \boldsymbol{x}_e - \hat{\boldsymbol{x}}_e$	
$\boldsymbol{e}_{xe}$	State and state reference error. $\boldsymbol{e}_{xe} = \boldsymbol{x}_{e\_ref} - \boldsymbol{x}_e$	
e_X	Position tracking error in X direction	m
e_V	Error between control velocity V_c and $\dot{X}$ in X direction	$m\ s^{-1}$
h_i	Activation function of the i-th subsystem	
p	Angular velocity of the quadrotor around x axis in B-frame	$rad\ s^{-1}$
p^d	Desired angular velocity of the quadrotor around x axis in B-frame	$rad\ s^{-1}$
q	Angular velocity of the quadrotor around y axis in B-frame	$rad\ s^{-1}$
q^d	Desired angular velocity of the quadrotor around y axis in B-frame	$rad\ s^{-1}$
r	Angular velocity of the quadrotor around z axis in B-frame	$rad\ s^{-1}$
r^d	Desired angular velocity of the quadrotor around z axis in B-frame	$rad\ s^{-1}$
u	x-coordinate velocity of the quadrotor in B-frame	$m\ s^{-1}$
$\boldsymbol{u}$	Input vector of the AA, TS and TS-LIA models	
$\boldsymbol{u}_F$	Input vector of the Apkarian filter	
$\boldsymbol{u}_{F\_ref}$	Filter input reference for tracking	
$\boldsymbol{u}^*$	Equilibrium input vector of the AA model	
v	y-coordinate velocity of the quadrotor in B-frame	$m\ s^{-1}$
v_Z	State variable of the AA model, equivalent to $\dot{Z}$	$m\ s^{-1}$
v_Z^d	Desired $\dot{Z}$	$m\ s^{-1}$
w	z-coordinate velocity of the quadrotor in B-frame	$m\ s^{-1}$
$\boldsymbol{x}$	State vector of the AA, TS and TS-LIA models	
$\boldsymbol{x}_e$	State vector of the TS-LIA model extended with an Apkarian filter	
$\hat{\boldsymbol{x}}_e$	Estimated state vector of the extended TS-LIA model	

$\boldsymbol{x}_{e\_ref}$	Extended state reference for tracking	
$\boldsymbol{x}_{ref}$	State reference for tracking	
$\boldsymbol{x}_F$	State vector of the Apkarian filter	
$\boldsymbol{x}^*$	Equilibrium state vector of the AA model	
$\boldsymbol{y}$	Output vector of measured position Z and orientation φ, θ and ψ	
$\boldsymbol{y}^d$	Vector of desired position Z^d and orientation φ^d, θ^d and ψ^d	
$\boldsymbol{z}$	Vector of premise variables	
$\boldsymbol{A}_i$	Matrix $\boldsymbol{A}$ of the i-th subsystem in a TS model	
$\boldsymbol{A}_e$	Matrix $\boldsymbol{A}$ of the TS-LIA model extended with an Apkarian filter	
$\boldsymbol{A}_F$	Matrix $\boldsymbol{A}$ of the Apkarian filter	
$\boldsymbol{B}_i$	Matrix $\boldsymbol{B}$ of the i-th subsystem in a TS model	
$\boldsymbol{B}_e$	Matrix $\boldsymbol{B}$ of the TS-LIA model extended with an Apkarian filter	
$\boldsymbol{B}_F$	Matrix $\boldsymbol{B}$ of the Apkarian filter	
$\boldsymbol{C}$	Coriolis-centripetal matrix	
$\boldsymbol{E}$	Movement matrix	
$\boldsymbol{G}$	Gravitational vector	
$G_c(s)$	IB controller transfer function	
$G_{ol}(s)$	Open-loop transfer function of the path following control scheme	
$G_X(s)$	Transfer function of $\ddot{X}$ dynamics	
$\boldsymbol{K}_i$	Controller gain of the i-th subsystem in the PDC framework	
K	Static gain of $G_{ol}(s)$	
K_d	Constant of the derivative term in the IB controller	s^{-1}
K_p	Constant of the proportional term in the IB controller	s^{-2}
K_v	Constant of velocity control in the IB controller	s^{-1}
K_X	Static gain of $G_X(s)$	
M_{ij}	Membership functions of premise variable i	
$\boldsymbol{M}$	Inertia matrix	
$\boldsymbol{M}_i$	$-\boldsymbol{L}_i^T\boldsymbol{P}$	
$\boldsymbol{L}_i$	Observer gain of the i-th subsystem	
O_B	Origin of the body-fixed reference frame	
O_E	Origin of the earth reference frame	
$\boldsymbol{O}$	Gyroscopic propeller matrix	

$\boldsymbol{P}$	Symmetric positive definite matrix of a Lyapunov function	
$\boldsymbol{Q}$	Inverse of $\boldsymbol{P}$	
$\boldsymbol{R}$	Rotation matrix	
$\mathbb{S}_{\alpha,\rho,\gamma}$	LMI region for pole placement (see (4.18))	
$\boldsymbol{T}$	Euler matrix	
U_1	Lift force in Z_B	N
U_2	Roll torque around X_B	$N\ m$
U_3	Pitch torque around Y_B	$N\ m$
U_4	Yaw torque around Z_B	$N\ m$
V_c	Virtual control velocity in the IB controller	$m\ s^{-1}$
$\boldsymbol{V}$	Linear velocity of the center of the quadrotor (O_E) in B-frame	
$\boldsymbol{W}_i$	$\boldsymbol{K}_i\boldsymbol{Q}$	
X	x-coordinate position of the quadrotor in E-frame	m
$\dot{X}$	x-coordinate velocity of the quadrotor in E-frame	$m\ s^{-1}$
X_B	X axis of the body-fixed reference frame	
X_E	X axis of the earth reference frame	
X^d	Desired X position in E-frame. X_T in a trajectory.	m
Y	y-coordinate position of the quadrotor in E-frame	m
$\dot{Y}$	y-coordinate velocity of the quadrotor in E-frame	$m\ s^{-1}$
Y_B	Y axis of the body-fixed reference frame	
Y_E	Y axis of the earth reference frame	
Y^d	Desired Y position in E-frame. Y_T in a trajectory.	m
Z	z-coordinate position of the quadrotor in E-frame	m
Z^d	Desired altitude Z in E-frame. Z_T in a trajectory.	m
$\dot{Z}$	z-coordinate velocity of the quadrotor in E-frame	$m\ s^{-1}$
Z_B	Z axis of the body-fixed reference frame	
Z_E	Z axis of the earth reference frame	
α	Minimum decay rate	s^{-1}
γ	Angle between a complex pole and -x axis. $\xi = \cos\gamma$	rad
$\boldsymbol{\zeta}$	Generalized velocity vector in H-frame	
$\dot{\boldsymbol{\zeta}}$	Generalized acceleration vector in H-frame	

θ	'Pitch' Euler angle	rad
θ^d	Desired 'Pitch' Euler angle	rad
$\dot{\theta}$	Rate of change of 'Pitch' angle	$rad\ s^{-1}$
λ	Real pole of the position tracking error model	s^{-1}
$\boldsymbol{\mu}$	Controller output	
$\boldsymbol{\nu}$	Generalized velocity vector in B-frame	
ξ	Damping ratio	
ξ_c	Damping ratio of $G_c(s)$	
ξ_X	Damping ratio of $G_X(s)$	
$\boldsymbol{\xi}$	Generalized position vector in E-frame	
$\dot{\boldsymbol{\xi}}$	Generalized velocity vector in E-frame	
ρ	Maximum natural frequency	$rad\ s^{-1}$
φ	'Roll' Euler angle	rad
φ^d	Desired 'Roll' Euler angle	rad
$\dot{\varphi}$	Rate of change of 'Roll' angle	$rad\ s^{-1}$
ψ	'Yaw' Euler angle	rad
ψ^d	Desired 'Yaw' Euler angle	rad
$\dot{\psi}$	Rate of change of 'Yaw' angle	$rad\ s^{-1}$
ω_n	undamped natural frequency	$rad\ s^{-1}$
ω_{nc}	Natural frequency of $G_c(s)$	$rad\ s^{-1}$
ω_{nX}	Natural frequency of $G_X(s)$	$rad\ s^{-1}$
ω_d	frequency of the output	$rad\ s^{-1}$
$\boldsymbol{\omega}$	Angular velocity vector in B-frame	
$\boldsymbol{\Theta}$	Angular position of the quadrotor in Euler Angles	
$\dot{\boldsymbol{\Theta}}$	Rate of change of Euler Angles	
$\boldsymbol{\Gamma}$	Linear position of the center of the quadrotor (O_E) in E-frame	
$\dot{\boldsymbol{\Gamma}}$	Linear velocity of the center of the quadrotor (O_E) in E-frame	
Ω	Overall propellers speed	$rad\ s^{-1}$
Ω^*	Propellers angular speed at equilibrium in the AA-model	$rad\ s^{-1}$
Ω_i	Angular speed of propeller i	$rad\ s^{-1}$
$\boldsymbol{\Omega}$	Propellers speed vector	
$\boldsymbol{\Omega}_{ref}$	Vector of input reference	

Chapter 1

Introduction

1.1 Motivation

Linear control theory provides powerful tools for analysis and design of controllers. However, in the case of non-linear systems the control techniques are often not systematic and hardly generalizable. The design of gain scheduled linear controllers at different operating points allows the application of all the tools from linear control theory to non-linear systems conveniently extended.

One popular approach for gain-scheduling control is based on the Linear Parameter Varying (LPV) paradigm. LPV models has been applied in a widely range of systems [1, p. 10], including Unmanned Aerial Vehicles (UAVs). Takagi-Sugeno (TS) paradigm is an alternative approach that has been proven to be equivalent to LPV paradigm [1]. Due to this equivalency between LPV and TS models, it would be interesting to explore the application of TS framework to a quadrotor system, that has been typically controlled applying the LPV paradigm.

1.2 Objectives

The main goal of this thesis is the design of a gain-scheduling controller for a quadrotor system using the Takagi-Sugeno paradigm. The specific objectives are the following:

- To develop a Takagi-Sugeno model from the Attitude/Altitude (AA) model of the quadrotor system.
- To design a state feedback controller for altitude and orientation control using the previous TS model.
- To design a controller for tracking of 3D trajectories, using the previous control and a new control for horizontal position.
- To test the resulting controllers in a simulation environment.

1.3 Outline of the thesis

This thesis has been structured as follows:

Chapter 2: This chapter reviews general concepts about the quadrotor system. The fundamentals of Takagi-Sugeno fuzzy models and Parallel Distributed Compensation technique are explained. Finally the basic concepts of quadratic stability and pole placement using LMI's are explained.

Chapter 3: This chapter has been dedicated to the derivation of two Altitude/Attitude quadrotor models: the simulation AA model, which will represent the real system in simulation; and the Takagi-Sugeno (TS) control model used for the design of the Altitude/Attitude controller. Two Takagi-Sugeno models has been derived: the first one (TS) is obtained directly from the AA model whereas the second one (TS-LIA) is obtained from the AA model considering a linear approximation in the input. Finally, the simulation model and the TS models has been simulated, compared and validated.

Chapter 4: In this chapter, the theoretical concepts about the control of altitude and orientation has been derived. In the first part, a state feedback controller and a state observer are developed. In the second part, two different control schemes for tracking of variable references are computed: the FeedForward (FF) control scheme and the Reference Model based FeedForward (RM-FF) control scheme.

Chapter 5: This chapter shows the development of an Integral Backstepping (IB) controller that, combined with the Altitude/Attitude controller, provides the tracking control of trajectories in 3D space. The general control scheme for path following requires the computation of references for roll and pitch orientations. The formulas for these computations are derived, and finally the stability of the global system is analysed.

Chapter 6: In this chapter, some examples of controllers has been tested in simulation. First, three different state feedback controllers and a state observer have been designed. Then, the RM-FF control scheme for reference tracking is implemented and simulated. Finally, some examples of 3D trajectory tracking are simulated.

Chapter 7: In this final chapter, the conclusions and further work are discussed.

Appendix A: The contents of the appendix are the following: In the first section, the Newton-Euler model of the quadrotor system is explained in detail. The second section summarizes the values of constants and parameters used in the models. The third section explains some generalities about Takagi-Sugeno models and how to reduce the number of rules. The last section shows how the membership functions are obtained for the case of triangular and rectangular polytopes.

Appendix B: Costs and sustainability.

Chapter 2

Background

In the first section of this chapter the basic concepts about the quadrotor system are explained. In the next sections, the fundamentals about Takagi-Sugeno models and Parallel distributed Compensation approach is presented. Finally, the problem of designing a state-feedback controller using LMIs is explained.

2.1 Quadrotor system

A quadrotor helicopter is an Unmanned Aerial Vehicle (UAV) with four propellers distributed as it is shown in the scheme of Figure 2.1. In that scheme, the quadrotor is seen from "above", being the "front" direction the one indicated with an arrow in the propeller named as '1'.

The quadrotor system can fly vertically or stay stationary in the air, just like any other VTOL (Vertical Take-Off and Landing) vehicle. However, one particularity of this kind of VTOL (unlike the traditional helicopter configuration of rotors) is that all the propellers are on the same plane. In a helicopter, a tail rotor (orthogonal to the main rotor) is needed to produce a 'yaw' rotation (a rotation with respect the axis orthogonal to the picture of Figure 2.1). In order to produce the yaw rotation in the quadrotor, the front and rear propellers (1 and 3) rotate in one direction, and the other pair of propellers (2 and 4) rotates in the opposite direction, just as it is indicated with arrows in the scheme.

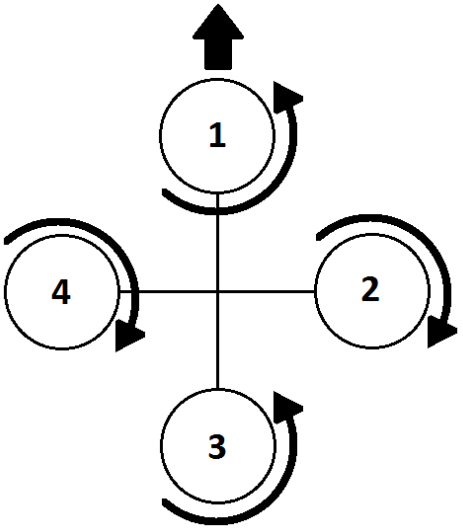


FIGURE 2.1: Scheme of the quadrotor system

The propellers rotating in the direction shown in Figure 2.1 always generate a force on the quadrotor pointing upwards. When all the propellers have the same angular speed, and the force generated compensates gravity, then the quadrotor is in *hovering* condition. In that state there are no rotations nor translations, so the quadrotor stays stationary in the air.

By changing some of the propellers speed from the hovering value, the following four basic movements can be developed:

- **Vertical acceleration**: If all four propellers rotates at the same angular speed, but the lift force is bigger (or lower) than the one due to gravity, then the quadrotor accelerates vertically upwards (or downwards). When the plane of the quadrotor is not orthogonal to gravity acceleration vector, the quadrotor also has an horizontal component in its acceleration vector.

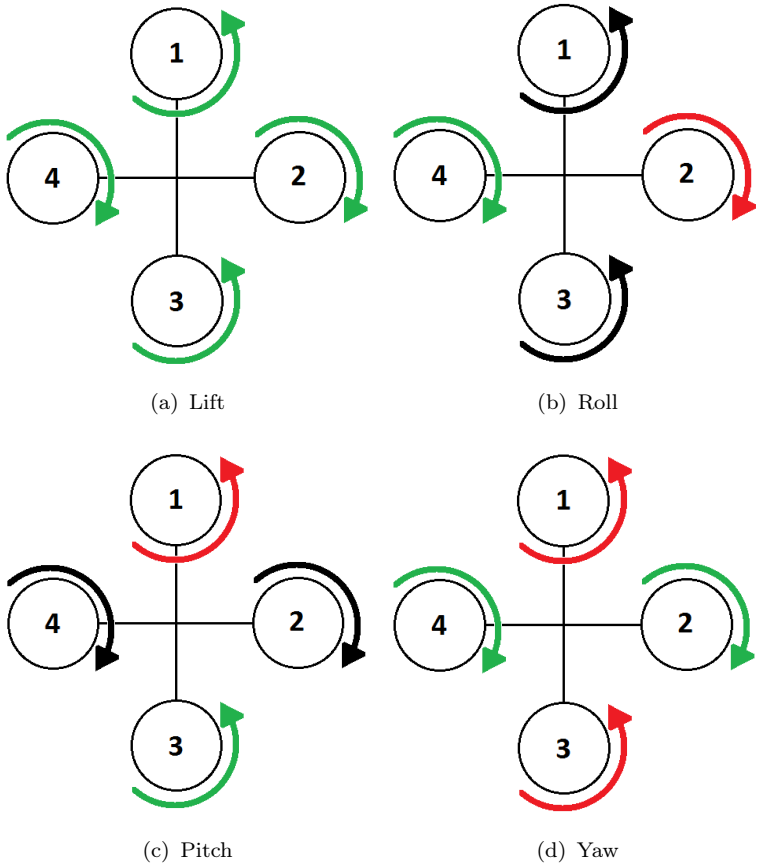
- **Roll**: When propellers 1 and 3 rotates at same speed but speed of 2 and 4 are different from each other, an angular acceleration is generated in the direction of the vertical axis shown in Figure 2.1.

- **Pitch**: When propellers 2 and 4 rotates at same speed but speed of 1 and 3 are different, an angular acceleration is generated in the direction of the horizontal axis

shown in Figure 2.1. Due to the symmetry of the system the dynamics regarding this movement will be similar to the roll rotation.

- **Yaw**: Finally, let us imagine that the four propellers generates a force that compensates gravity, but they do not have all the same angular speed. Instead of that, propellers 1 and 3 rotates at one speed and propellers 2 and 4 rotates at other different speed. As a result, the counter-torque between each pair of propellers is not compensated anymore [2]. The consequence is an angular acceleration, known as yaw, with respect the axis orthogonal to the plane of the quadrotor.

In Figure 2.2, it is shown the relation between the basic movements explained above and the angular speed of the propellers.



(a) Lift (b) Roll

(c) Pitch (d) Yaw

FIGURE 2.2: Scheme of basic movements when propellers rotate at higher (green lines), lower (red lines) or at equal (black lines) speed than the hovering value.

2.2 Takagi-Sugeno fuzzy models

This section will introduce some basic concepts about the construction of Takagi-Sugeno (TS) fuzzy models.

A TS fuzzy model allows the representation of a non-linear model as a set of local LTI (Linear Time Invariant) models [1, p. 10], each one called *subsystem*. A subsystem is the local representation of the system in the space of premise variables $\boldsymbol{z}(t) = [z_1(t)\ z_2(t)\ \ldots\ z_p(t)]$, which are known and could depend on the state variables and input variables. Each subsystem $\dot{\boldsymbol{x}}(t) = \boldsymbol{A_i}\ \boldsymbol{x}(t) + \boldsymbol{B_i}\ \boldsymbol{u}(t)$ has a fuzzy rule associated with the following form [3]:

$$\begin{aligned} \textbf{IF} \quad & z_1(t) \text{ is } M_{i1} \text{ and } z_2(t) \text{ is } M_{i2} \ \ldots \text{ and } z_p(t) \text{ is } M_{ip}, \\ \textbf{THEN} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{A_i}\ \boldsymbol{x}(t) + \boldsymbol{B_i}\ \boldsymbol{u}(t), \quad i = 1, 2, \ldots, r. \end{aligned} \tag{2.1}$$

There are r fuzzy rules, as many as subsystems. The i-th rule (2.1) of the TS model can be read as follows: if the premise variable $z_1(t)$ belongs to the fuzzy set M_{i1} up to some degree, and $z_2(t)$ belongs to the fuzzy set M_{i2} up to some degree, and the same for the other premise variables and fuzzy sets, then the TS fuzzy model is equivalent to subsystem $\dot{\boldsymbol{x}}(t) = \boldsymbol{A_i}\ \boldsymbol{x}(t) + \boldsymbol{B_i}\ \boldsymbol{u}(t)$ up to some degree. In the especial case where all the premise variables totally belongs to the corresponding fuzzy sets, then the TS model is exactly the i-th subsystem.

The fuzzy sets can be seen as labels that does not represent a concrete value of the premise variables, but a subjective value instead. In (2.1), the IF condition labels the premise variables with the meaning associated with the fuzzy set. For example, the meaning of M_{i1} could be that $z_1(t)$ is "big" or "positive" or "non-negative", etc. In order to compute the degree of membership of a premise variable to a fuzzy set, a *membership function* is needed. The process of mapping the premise variables into the subjective values is known as *fuzzification* [4].

The next step after the fuzzification is the inference of the model from the values of the membership functions. In this step, the output of each rule is computed, which in the

case of a TS model is just the non-linear model evaluated at one operating point (i.e. the subsystems matrices $\boldsymbol{A_i}$ and $\boldsymbol{B_i}$).

Remark 2.1. In this work, the fuzzy sets are based on the bounds of the premise variables. Therefore each premise variable has two membership functions: the one related with the lower bound and the one related with the upper bound. □

The last step is the *defuzzification*, where a mapping between the fuzzy output (i-th subsystem) and a particular linear model is done [4]. Given the input and state vectors $(\boldsymbol{u}(t), \boldsymbol{x}(t))$, and the premise variables $\boldsymbol{z}(t)$, the output model is computed as follows [3]:

$$\dot{\boldsymbol{x}} = \sum_{i=1}^{r} h_i(\boldsymbol{z}(t)) \left\{ \boldsymbol{A_i}\ \boldsymbol{x}(t) + \boldsymbol{B_i}\ \boldsymbol{u}(t) \right\} = \left(\sum_{i=1}^{r} h_i \boldsymbol{A_i} \right) \boldsymbol{x}(t) + \left(\sum_{i=1}^{r} h_i \boldsymbol{B_i} \right) \boldsymbol{u}(t) \tag{2.2}$$

Where the activation function h_i is:

$$h_i(\boldsymbol{z}(t)) = \frac{w_i(\boldsymbol{z}(t))}{\sum_{i=1}^{r} w_i(\boldsymbol{z}(t))}, \quad w_i(\boldsymbol{z}(t)) = \prod_{j=1}^{p} M_{ij}(\boldsymbol{z}(t)) \tag{2.3}$$

The weight w_i measures the degree of membership of all the premise variables in the related fuzzy set of the i-th rule. The activation function is the normalization of these weights, so that the sum of activation functions is one.

Remark 2.2. As it is shown in (2.3), the normalization is done after the product of membership functions. However, in this work an equivalent procedure is applied, where the membership functions maps the premise variables to values between 0 and 1, and then the activation function is computed as the product of membership functions. Let consider a simple case of two bounded premise variables $z_1(t)$ and $z_2(t)$ where the difference between the upper and lower bound is L_1 and L_2, respectively. The membership functions are the distances to the bounds, such that a bigger value means a higher degree of membership to the fuzzy set. For example, the degree of membership to the fuzzy set "small" in the case of $z_1(t)$ would be measured by the distance from $z_1(t)$ to the upper bound of $z_1(t)$.

Then, the fuzzy sets that appear in each rule are:

$$\begin{aligned}
&\textbf{R1:} \quad M_{11} = \text{“small”}, \;\; M_{12} = \text{“small”} \\
&\textbf{R2:} \quad M_{21} = M_{11} = \text{“small”}, M_{22} = L_2 - M_{12} = \text{“big”} \\
&\textbf{R3:} \quad M_{31} = L_1 - M_{11} = \text{“big”}, M_{32} = M_{12} = \text{“small”} \\
&\textbf{R4:} \quad M_{41} = L_1 - M_{11} = \text{“big”}, M_{42} = L_2 - M_{12} = \text{“big”}
\end{aligned}$$

The sum of weights is:

$$\begin{aligned}
\sum_{i=1}^{4} w_i(\boldsymbol{z}(t)) &= \sum_{i=1}^{4} \left(\prod_{j=1}^{2} M_{ij} \right) = M_{11}M_{12} + M_{21}M_{22} + M_{31}M_{32} + M_{41}M_{42} = \\
&= M_{11}M_{12} + M_{11}(L_2 - M_{12}) + (L_1 - M_{11})M_{12} + (L_1 - M_{11})(L_2 - M_{12}) = \\
&= L_1 L_2
\end{aligned}$$

The activation function of rule i can be written as the product of normalized membership functions $\bar{M}_{i1}$, $\bar{M}_{i2}$:

$$h_i(\boldsymbol{z}(t)) = \frac{w_i(\boldsymbol{z}(t))}{\sum_{i=1}^{4} w_i(\boldsymbol{z}(t))} = \frac{M_{i1}M_{i2}}{L_1 L_2} = \bar{M}_{i1}\bar{M}_{i2}, \quad \bar{M}_{i1} = \frac{M_{i1}}{L_1}, \;\; \bar{M}_{i2} = \frac{M_{i2}}{L_2}$$

□

TS fuzzy models are universal approximators [4], which means that any non-linear model can be expressed with any arbitrary accuracy by a set of rules and LTI subsystems as shown in (2.1) and (2.2). However, in general the accuracy of the model will depend on the number of fuzzy rules, i.e. the number of subsystems considered [4]. In the example shown in Remark 2.2, there are two premise variables and two membership functions for each variable, so there are $2^2 = 4$ rules.

In general, if there are two membership functions for each premise variable and all the combinations of fuzzy sets are considered, for k premise variables there are 2^k rules and subsystems. This could lead into a problem regarding the computational time in the design of the controller and the performance of the simulations. Therefore there is a trade-off

between accuracy and dimensionality that could make difficult to find an appropriate TS model.

About the construction of the fuzzy model, there are two different approaches to the problem of obtaining the set of LTI systems from the non-linear model: one is the local sector non-linearity and the other one is the local approximation of fuzzy partition spaces [3].

Let consider a simple non-linear first-order differential equation $\dot{x}(t) = f(x(t))$, defined in some interval of x around zero. The idea of local sector non-linearity is to find two linear differential equations $a_1x(t)$ and $a_2x(t)$ that bounds the function $f(x(t))$ in the interval. An advantage of this approach is that the fuzzy model is not an approximation of the non-linear model, and the non-linearity is embedded in the premise variables so the model exactly represents the non-linear system in the local region [3].

In the local approximation approach, the linear subsystems are found by approximating the non-linear terms to linear expressions. An example of this approach can be seen in [5], where the whole non-linear model is linearized using Taylor series expansion around three different operating points. The main advantage of this method is the reduction of the number of rules.

In this work, two different Takagi-Sugeno models has been derived: the first one is based on the sector non-linearity approach (see Section 3.2), whereas the second one is obtained by a combination of both approaches (see Section 3.3).

2.3 Parallel Distributed Compensation

Given a TS fuzzy model, a gain-scheduled controller can be design using the Parallel Distributed Compensation (PDC) approach [3]. In this approach a fuzzy controller is constructed with the same number of fuzzy rules than subsystems of the TS fuzzy model. For each subsystem of the fuzzy TS model, a state feedback controller is designed. Note that a linear controller can be designed since the subsystems are LTI systems. The i-th

rule of the fuzzy controller is shown in (2.4). Note also that the condition is equivalent to rule i-th of the fuzzy model (2.1).

$$\begin{aligned} \textbf{IF} \quad & z_1(t) \text{ is } M_{i1} \text{ and } z_2(t) \text{ is } M_{i2} \ \dots \text{ and } z_p(t) \text{ is } M_{ip}, \\ \textbf{THEN} \quad & \boldsymbol{u}(t) = \boldsymbol{K}_i \ \boldsymbol{x}(t), \quad i = 1, 2, \dots, r. \end{aligned} \tag{2.4}$$

The rule of a fuzzy controller can be read in a similar way than a rule of the TS model: if the premise variable $z_1(t)$ belongs to the fuzzy set M_{i1} up to some degree, and $z_2(t)$ belongs to the fuzzy set M_{i2} up to some degree, and the same for the other premise variables and fuzzy sets, then the controller gain is $\boldsymbol{K}_i$ up to some degree.

The defuzzification of the controller (2.5) is performed by computing a linear combination of the controllers for each subsystem $\boldsymbol{K}_i$ and using the same activation functions (2.3) than in the defuzzification of the TS model.

$$\boldsymbol{u}(t) = \sum_{i=1}^{r} h_i(\boldsymbol{z}(t)) \ \boldsymbol{K}_i \ \boldsymbol{x}(t) \tag{2.5}$$

By substituting (2.5) into (2.2), the closed loop system (2.6) is obtained.

$$\dot{\boldsymbol{x}} = \sum_{i=1}^{r} \sum_{j=1}^{r} h_i(\boldsymbol{z}(t)) h_j(\boldsymbol{z}(t)) \left\{ \boldsymbol{A_i} + \boldsymbol{B_i} \ \boldsymbol{K}_j \right\} \ \boldsymbol{x}(t) \tag{2.6}$$

2.4 Controller design based on LMI's

In this section, it will be explained how to compute the set of controllers $\boldsymbol{K}_i$ shown in (2.5) so that the closed-loop system (2.6) is stable. First, let define some concepts about Lyapunov stability, quadratic stability and $\mathbb{D}$-Stability in a LMI region $\mathbb{D}$.

2.4.1 Quadratic stability

Let consider an autonomous system $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ with $\boldsymbol{A}$ being a constant matrix. If we define the Lyapunov function $V(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x}$, then the system is *stable* if there exist $\boldsymbol{P} > 0$ such that condition (2.7) is satisfied [6, p. 96].

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} < 0 \tag{2.7}$$

If we have a family of matrices $\boldsymbol{A}(\delta(t))$ (where $\delta(t)$ is a parameter that is bounded by a polytope Δ) instead of a single matrix $\boldsymbol{A}$, then the system equation becomes $\dot{\boldsymbol{x}} = \boldsymbol{A}(\delta(t))\boldsymbol{x}$ and condition (2.7) should be satisfied for all possible values of $\delta(t)$. If exists $\boldsymbol{P} > 0$ such that (2.8) is satisfied then the system is *quadratically stable* [1, p. 24].

$$\boldsymbol{A}(\delta(t))^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}(\delta(t)) < 0 \quad \forall\delta(t) \in \Delta \tag{2.8}$$

Since there are an infinite number of matrices $\boldsymbol{A}(\delta(t))$ there is also an infinite number of constraints like (2.8) that should be fulfilled. From a practical point of view this makes the problem impossible to be solved. Let consider now that the system $\dot{\boldsymbol{x}} = \boldsymbol{A}(\delta(t))\boldsymbol{x}$ can be written in a polytopic form (2.9) as a Takagi-Sugeno (TS) polytopic system with premise variables $\boldsymbol{z}(t)$ and a set of r subsystems $\boldsymbol{A}_i$ for $i = \{1, \ldots, r\}$.

$$\dot{\boldsymbol{x}}(t) = \sum_{i=1}^{r} h_i(\boldsymbol{z}(t))\ \boldsymbol{A}_i\ \boldsymbol{x}(t) \tag{2.9}$$

It can be proven [1, p. 31] that a polytopic autonomous system (2.9) is quadratically stable if condition (2.8) is satisfied in the vertices (subsystems) of the polytope (2.10). Therefore there is no need to check stability in an infinite number of matrices, but only in subsystems matrices $\boldsymbol{A}_i$.

$$\boldsymbol{A}_i^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_i < 0 \quad \forall i = 1, \ldots, r \tag{2.10}$$

Note that the closed-loop system in (2.6) is an autonomous TS polytopic system. As it is shown in [3, p. 51], stability conditions (2.10) can be applied to the closed-loop system (2.6) and the following set of conditions are obtained

$$\begin{gathered} \boldsymbol{G}_{ii}^T\ \boldsymbol{P} + \boldsymbol{P}\ \boldsymbol{G}_{ii} < 0 \quad \forall i = 1, \ldots, r \\ \left(\frac{\boldsymbol{G}_{ij} + \boldsymbol{G}_{ji}}{2}\right)^T\ \boldsymbol{P} + \boldsymbol{P}\ \left(\frac{\boldsymbol{G}_{ij} + \boldsymbol{G}_{ji}}{2}\right) \leq 0 \quad \forall i, j \in \{1, \ldots, r\},\ i < j \end{gathered} \tag{2.11}$$

where $\boldsymbol{G}_{ij} = \boldsymbol{A}_i + \boldsymbol{B}_i\boldsymbol{K}_j$ and $h_i(\boldsymbol{z}(t))h_j(\boldsymbol{z}(t)) \neq 0$.

In the special case where matrices $\boldsymbol{B}_i$ are constant (i.e. $\boldsymbol{B}_i = \boldsymbol{B}$), the first set of inequalities in (2.11) are enough to prove stability. Therefore, assuming constant $\boldsymbol{B}$ for all the subsystems, if there exist $\boldsymbol{P} > 0$ such that conditions (2.12) are fulfilled, then the polytopic TS model (2.2) with state feedback control (2.5) is quadratically stable inside the polytope.

$$(\boldsymbol{A}_i + \boldsymbol{B}\boldsymbol{K}_i)^T\ \boldsymbol{P} + \boldsymbol{P}\ (\boldsymbol{A}_i + \boldsymbol{B}\boldsymbol{K}_i) < 0 \quad \forall i = 1, \ldots, r \tag{2.12}$$

Remark 2.3. The assumption of constant $\boldsymbol{B}$ can be achieve using a prefiltering of the input [1, p. 44]. This change is not restrictive and the main consequence is the addition of some new state variables (the ones from the filter) to the TS model. □

The design of the controller that stabilizes the closed-loop system boils down to solve the Linear Matrix Inequality (LMI) problem of finding a positive definite matrix $\boldsymbol{P}$ and a set of matrices $\boldsymbol{K}_i$ such that conditions (2.12) are fulfilled. However, since the constraints should be linear combinations of the unknown variable, the following change of variables is applied: $\boldsymbol{W}_i = \boldsymbol{K}_i\boldsymbol{Q}$ where $\boldsymbol{Q} = \boldsymbol{P}^{-1}$. The solution of the LMI problem is the set of matrices $\boldsymbol{W}_i$ such that conditions (2.13) are fulfilled.

$$\begin{cases} \boldsymbol{Q} > 0 \\ \boldsymbol{A}_i\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A}_i^T + \boldsymbol{B}\boldsymbol{W}_i + \boldsymbol{W}_i^T\boldsymbol{B}^T < 0 \quad \forall i = 1, \ldots, r \end{cases} \tag{2.13}$$

The i-th controller is computed from the solution as $\boldsymbol{K}_i = \boldsymbol{W}_i\boldsymbol{Q}^{-1}$.

2.4.2 Pole placement

We may want not only ensure stability to the closed-loop system but also impose some conditions regarding the performance. This is done by placing the complex poles of the system in a particular LMI region inside the complex plane.

A region $\mathbb{D}$ in the complex plane is an LMI region if it can be defined by (2.14) for some symmetric matrix $\boldsymbol{L}$ and matrix $\boldsymbol{M}$. An LMI region is always convex and symmetric about the real axis [6, p. 103].

$$\mathbb{D} = \left\{ s |\ s \in \mathbb{C},\ \boldsymbol{L} + s\boldsymbol{M} + \bar{s}\boldsymbol{M}^T < 0 \right\} \tag{2.14}$$

Let consider again the polytopic autonomous system (2.9). The system is *quadratically* $\mathbb{D}$*-stable* if the poles are in the LMI region $\mathbb{D}$. The required condition for $\mathbb{D}$-stability is shown in (2.15), where operator $\otimes$ represents the Kronecker product [1, p. 32].

$$\boldsymbol{L} \otimes \boldsymbol{P} + \boldsymbol{M} \otimes \boldsymbol{P}\boldsymbol{A}_i + \boldsymbol{M}^T \otimes \boldsymbol{A}_i^T \boldsymbol{P} < 0 \quad \forall i = 1, \ldots, r \tag{2.15}$$

If we assume constant matrices $\boldsymbol{B}$, condition (2.15) applied to the closed-loop system (2.6) produces a condition equivalent to (2.12) in the quadratic stabilization problem, where matrices $\boldsymbol{A}_i$ in (2.15) are substituted by matrices $\boldsymbol{G}_{ii} = \boldsymbol{A}_i + \boldsymbol{B}_i\boldsymbol{K}_i$

$$\boldsymbol{L} \otimes \boldsymbol{P} + \boldsymbol{M} \otimes \boldsymbol{P}(\boldsymbol{A}_i + \boldsymbol{B}\boldsymbol{K}_i) + \boldsymbol{M}^T \otimes (\boldsymbol{A}_i + \boldsymbol{B}\boldsymbol{K}_i)^T \boldsymbol{P} < 0 \quad \forall i = 1, \ldots, r \tag{2.16}$$

The design of the controller that sets the poles in the LMI region $\mathbb{D}$ involves solving the Linear Matrix Inequality (LMI) problem of finding a positive definite matrix $\boldsymbol{P}$ and a set of matrices $\boldsymbol{K}_i$, as in the quadratic stabilization problem.

Chapter 3

Quadrotor models

In the first section of this chapter, the dynamic model of the quadrotor system is presented. This model will represent the real system in the simulations, and is needed for the generation of the Takagi-Sugeno models. The modelling part is based mainly on the work of Bresciani [7], who applies the Newton-Euler formalism in order to derive the differential equations of the model. The Attitude/Altitude (AA) model is obtained in state space form and the equilibrium points are computed.

In the following sections, two Takagi-Sugeno models from the AA model of the quadrotor has been derived. The first TS model is obtained by applying the sector non-linearity approach. In order to reduce the number of premise variables, a second TS-LIA model has been derived by a combination of sector non-linearity and local approximation approaches. Finally, both TS models has been simulated and validated.

3.1 Quadrotor non-linear model

In order to design an appropriate controller for the quadrotor system, we need first and appropriate mathematical description of the system. This (the dynamical model) will consist of six second-order non-linear differential equations, each one for one degree of freedom of the system (three translations and three rotations). These equations will explain how the position and orientation of the quadrotor (kinematics) is affected by the

forces and torques generated by the propellers (dynamics). A complete derivation of this equations is given in [7], where the Newton-Euler formalism is applied. Here, a brief description of that procedure will be explained, following the same notation as much as possible.

Each one of the basic movements described in Section 2.1 is directly related with a force or a torque in a principal direction of the quadrotor (the two axes where the propellers are attached and the axis orthogonal to the picture in Figure 2.1). Although it seems natural to define the position and orientation in an Earth inertial reference frame, it is easier to formulate the dynamic equations in a body fixed frame where the axes are the principal directions just mentioned.

In Figure 3.1, both Earth inertial reference frame (E-frame) and quadrotor body-fixed reference frame (B-frame) are shown. Each reference frame is defined by an origin point and three orthogonal vectors. The E-frame has an arbitrary origin in O_E, and Z_E is pointing upwards. The body-fixed reference frame has the origin O_B attached center of the quadrotor, and the axes are such that X_B points to propeller 1, Y_B points to propeller 4, and Z_B is orthogonal to both X_B and Y_B.

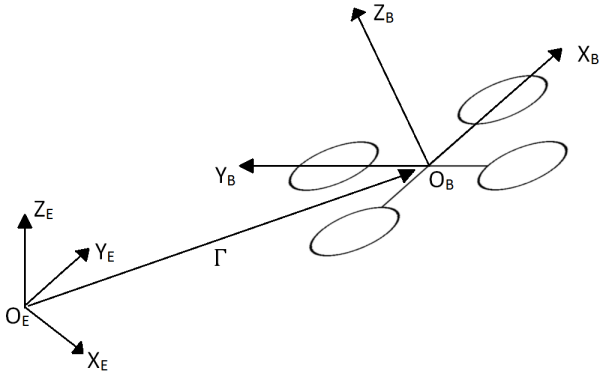


FIGURE 3.1: Earth reference frame (E-frame) and quadrotor reference frame (B-frame).

Let define the generalized position vector $\boldsymbol{\xi}$, which includes the coordinates of O_B in E-frame (named as $\mathbf{\Gamma} = [X \ Y \ Z]^T$, also shown in Figure 3.1) and the angular position $\mathbf{\Theta} = [\varphi \ \theta \ \psi]^T$, in Euler angles:

$$\boldsymbol{\xi} = \begin{bmatrix} \mathbf{\Gamma} & \mathbf{\Theta} \end{bmatrix}^T = \begin{bmatrix} X & Y & Z & \varphi & \theta & \psi \end{bmatrix}^T \tag{3.1}$$

The generalized position vector defines the position and orientation of B-frame with respect E-frame. To obtain B-frame from E-frame, we should translate the reference as indicates $\mathbf{\Gamma}$, and then first rotate an angle ψ about Z_E axis (named yaw), then rotate θ about the new (after yaw rotation) Y_E axis (named pitch), and finally rotate φ about the new (after yaw and pitch rotations) X_E axis (named roll).

The derivative of the generalized position vector $\boldsymbol{\xi}$ is the generalized velocity vector $\dot{\boldsymbol{\xi}}$, which includes the linear velocity of the quadrotor in E-frame $\dot{\mathbf{\Gamma}} = [\dot{X}\ \dot{Y}\ \dot{Z}]^T$, and the rate of change of Euler angles $\dot{\mathbf{\Theta}} = [\dot{\varphi}\ \dot{\theta}\ \dot{\psi}]^T$

$$\dot{\boldsymbol{\xi}} = \begin{bmatrix} \dot{\mathbf{\Gamma}} & \dot{\mathbf{\Theta}} \end{bmatrix}^T = \begin{bmatrix} \dot{X} & \dot{Y} & \dot{Z} & \dot{\varphi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T \tag{3.2}$$

We want the velocity (and also the acceleration) vector to be expressed in B-frame because the dynamic equations will be derived on that reference frame. Therefore, we can also define the generalized velocity vector in B-frame as $\boldsymbol{\nu}$, which includes the linear velocity of the quadrotor in B-frame $\boldsymbol{V} = [u\ v\ w]^T$ and the angular velocity of the quadrotor in B-frame $\boldsymbol{\omega} = [p\ q\ r]^T$

$$\boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{V} & \boldsymbol{\omega} \end{bmatrix}^T = \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^T \tag{3.3}$$

Linear and angular velocity vectors in both frames are related by the following formulas:

$$\dot{\mathbf{\Gamma}} = \boldsymbol{R}(\Theta) \cdot \boldsymbol{V}, \quad \dot{\mathbf{\Theta}} = \boldsymbol{T}(\Theta) \cdot \boldsymbol{\omega} \tag{3.4}$$

where $\boldsymbol{R}(\Theta)$ is the rotation matrix and $\boldsymbol{T}(\Theta)$ is the Euler matrix. Those matrices are defined as follows (considering $c_k = \cos(k)$, $s_k = \sin(k)$ and $t_k = \tan(k)$):

$$\boldsymbol{R}(\Theta) = \begin{bmatrix} c_\psi c_\theta & -s_\psi c_\varphi + c_\psi s_\theta s_\varphi & s_\psi s_\varphi + c_\psi s_\theta c_\varphi \\ s_\psi c_\theta & c_\psi c_\varphi + s_\psi s_\theta s_\varphi & -c_\psi s_\varphi + s_\psi s_\theta c_\varphi \\ -s_\theta & c_\theta s_\varphi & c_\theta c_\varphi \end{bmatrix} \tag{3.5}$$

$$\boldsymbol{T}(\Theta) = \begin{bmatrix} 1 & \sin\varphi\tan\theta & \cos\varphi\tan\theta \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi/\cos\theta & \cos\varphi/\cos\theta \end{bmatrix} \tag{3.6}$$

Remark 3.1. The angular velocity vector $\boldsymbol{\omega}$ is not equal to the rate of change of Euler angles $\dot{\boldsymbol{\Theta}}$. They are equivalent only if matrix $\boldsymbol{T}(\Theta)$ is diagonal, i.e. if the quadrotor is in hovering condition ($\theta = \varphi = 0$). □

Summarizing, up to now the following vectors have been defined: the linear and angular positions in E-frame ($\boldsymbol{\xi}$), linear and angular velocities in E-frame ($\dot{\boldsymbol{\xi}}$), and linear and angular velocities in B-frame ($\boldsymbol{\nu}$). Now an hybrid frame will be considered, such that the linear velocities and accelerations will be referred to E-frame, whereas the angular velocities and accelerations will be referred to B-frame. Therefore a new generalized velocity vector $\boldsymbol{\zeta}$ must be defined, which combines the linear part of $\dot{\boldsymbol{\xi}}$ with the angular part of $\boldsymbol{\nu}$

$$\boldsymbol{\zeta} = \begin{bmatrix} \dot{\boldsymbol{\Gamma}} & \boldsymbol{\omega} \end{bmatrix}^T = \begin{bmatrix} \dot{X} & \dot{Y} & \dot{Z} & p & q & r \end{bmatrix}^T \tag{3.7}$$

So, finally the position vector is $\boldsymbol{\xi}$, the velocity vector is $\boldsymbol{\zeta}$, and the acceleration vector is just the derivative of the velocity $\dot{\boldsymbol{\zeta}}$. According to [7], the dynamics equations in the hybrid frame can be written as

$$\boldsymbol{M}\ \dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\boldsymbol{\zeta})\ \boldsymbol{\zeta} = \boldsymbol{G} + \boldsymbol{O}(\boldsymbol{\zeta})\ \boldsymbol{\Omega} + \boldsymbol{E}(\boldsymbol{\xi})\ \boldsymbol{\Omega}^2 \tag{3.8}$$

The explanation of each term is shown in Section A.1.

A more useful representation of the model instead of the matrix version (3.8) is the set of differential equations

$$\begin{cases} \ddot{X} = (\sin\psi \sin\varphi + \cos\psi \sin\theta \cos\varphi)\ \dfrac{U_1}{m} \\ \ddot{Y} = (-\cos\psi \sin\varphi + \sin\psi \sin\theta \cos\varphi)\ \dfrac{U_1}{m} \\ \ddot{Z} = -g + (\cos\theta \cos\varphi)\ \dfrac{U_1}{m} \\ \dot{p} = \dfrac{I_Y - I_Z}{I_X}\ q\ r - \dfrac{J_{TP}}{I_X}\ q\ \Omega + \dfrac{U_2}{I_X} \\ \dot{q} = \dfrac{I_Z - I_X}{I_Y}\ p\ r + \dfrac{J_{TP}}{I_Y}\ p\ \Omega + \dfrac{U_3}{I_Y} \\ \dot{p} = \dfrac{I_X - I_Y}{I_Z}\ p\ q + \dfrac{U_4}{I_Z} \end{cases} \tag{3.9}$$

where the acceleration vector $\dot{\boldsymbol{\zeta}}$ has been isolated. Note that Ω is an scalar value different from the vector $\boldsymbol{\Omega}$ defined in (A.6).

The overall velocity Ω, the lift force U_1 and torques U_2, U_3 and U_4 are:

$$\begin{cases} U_1 = b\ (\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 = bl\ (-\Omega_2^2 + \Omega_4^2) \\ U_3 = bl\ (-\Omega_1^2 + \Omega_3^2) \\ U_4 = d\ (-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\ \Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{cases} \tag{3.10}$$

Equations in (3.9) describes the dynamics of the (simplified) quadrotor system. Note that there are three second-order differential equations and three first-order differential equations (instead of six second-order differential equations). As it was mentioned previously, the velocity vector $\boldsymbol{\zeta}$ is not just the derivative of the position vector $\boldsymbol{\xi}$ because of the different frames taken for the linear and angular variables (i.e. $\dot{\boldsymbol{\Theta}} \neq \boldsymbol{\omega}$). Therefore, the complete version of these equations should include another three first-order differential equations which relates the angular position $\boldsymbol{\Theta}$ with the angular velocity $\boldsymbol{\omega}$. This relation can be obtained from the second equation in (3.4), repeated and expanded for convenience as follows

$$\begin{cases} \dot{\varphi} = p + \sin\varphi \tan\theta \; q + \cos\varphi \tan\theta \; r \\ \dot{\theta} = \cos\varphi \; q - \sin\varphi \; r \\ \dot{\psi} = \dfrac{\sin\varphi}{\cos\theta} \; q + \dfrac{\cos\varphi}{\cos\theta} \; r \end{cases} \tag{3.11}$$

3.1.1 Attitude/Altitude model in state-space form

The control of the quadrotor will be divided on two different stages or blocks, as it will be explained in more detail in Chapter 5. One block is related with the control of attitude (desired value of the orientation in Euler angles, i.e. the desired $\boldsymbol{\Theta}$) and the control of altitude (height position Z). The second control block will provide to the first block a desired θ (pitch) and φ (roll) angles, which will allow the control of the horizontal position (i.e. X and Y). Therefore, for the first control block the first two equations in (3.9) will not be considered.

Let define the state vector $\boldsymbol{x}$, which includes (in a different order) the components of the position vector $\boldsymbol{\xi}$ (3.1) and the velocity vector $\boldsymbol{\zeta}$ (3.7) (without the mentioned linear positions and velocities regarding X and Y). Also the input vector $\boldsymbol{u}$ is defined, which contains the four propeller speeds (so its equivalent to vector $\boldsymbol{\Omega}$)

$$\boldsymbol{x} = \begin{bmatrix} Z & v_Z & \varphi & \theta & \psi & p & q & r \end{bmatrix}^T, \quad \boldsymbol{u} = \boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 & \Omega_2 & \Omega_3 & \Omega_4 \end{bmatrix}^T \tag{3.12}$$

where $v_Z = \dot{Z}$. Then (3.9) and (3.11) can be written together as a set of eight first-order differential equations in a non-linear state space form (3.13). Inputs (U_1, U_2, U_3, U_4) and Ω has been substituted by (3.10) so that the input are the speed of the propellers.

$$\begin{cases} \dot{z} = v_Z \\ \dot{v_Z} = -g + \dfrac{b}{m}\ \cos\theta\cos\varphi\ (\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ \dot{\varphi} = p + \sin\varphi\tan\theta\ q + \cos\varphi\tan\theta\ r \\ \dot{\theta} = \cos\varphi\ q - \sin\varphi\ r \\ \dot{\psi} = \dfrac{\sin\varphi}{\cos\theta}\ q + \dfrac{\cos\varphi}{\cos\theta}\ r \\ \dot{p} = \dfrac{I_Y - I_Z}{I_X}\ q\ r - \dfrac{J_{TP}}{I_X}\ q\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \dfrac{bl}{I_X}\ (\Omega_4^2 - \Omega_2^2) \\ \dot{q} = \dfrac{I_Z - I_X}{I_Y}\ p\ r + \dfrac{J_{TP}}{I_Y}\ p\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \dfrac{bl}{I_Y}\ (\Omega_3^2 - \Omega_1^2) \\ \dot{r} = \dfrac{I_X - I_Y}{I_Z}\ p\ q + \dfrac{d}{I_Z}\ (-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{cases} \tag{3.13}$$

As commented in the beginning of this section, the general goal of the controller is not to control the attitude, but the position (X, Y, Z) instead. It is assumed as a hypothesis that the quadrotor will be close to the hovering condition while it follows a desired trajectory [7, p. 33]. As a consequence of this assumption the Euler matrix (3.6) is close to the identity matrix (Remark 3.1). In other words, it can be assumed from (3.4) that $\dot{\mathbf{\Theta}} \approx \boldsymbol{\omega}$, so the rate of change of Euler angles are just $\dot{\varphi} = p$, $\dot{\theta} = q$ and $\dot{\psi} = r$.

After assuming this hypothesis, equations (3.13) becomes:

$$\begin{cases} \dot{z} = v_Z \\ \dot{v_Z} = -g + \dfrac{b}{m}\ \cos\theta\cos\varphi\ (\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ \dot{\varphi} = p \\ \dot{\theta} = q \\ \dot{\psi} = r \\ \dot{p} = \dfrac{I_Y - I_Z}{I_X}\ q\ r - \dfrac{J_{TP}}{I_X}\ q\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \dfrac{bl}{I_X}\ (\Omega_4^2 - \Omega_2^2) \\ \dot{q} = \dfrac{I_Z - I_X}{I_Y}\ p\ r + \dfrac{J_{TP}}{I_Y}\ p\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \dfrac{bl}{I_Y}\ (\Omega_3^2 - \Omega_1^2) \\ \dot{r} = \dfrac{I_X - I_Y}{I_Z}\ p\ q + \dfrac{d}{I_Z}\ (-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{cases} \tag{3.14}$$

3.1.2 Equilibrium points

The state vector $\boldsymbol{x}$ and the input $\boldsymbol{u}$ defines an equilibrium point for the system if, when the system is on that state and the input is applied, the state does not evolve over time. Therefore the equilibrium points are the set of solutions $\boldsymbol{x} = \boldsymbol{x^*}$ and $\boldsymbol{u} = \boldsymbol{u^*}$ for the system of equations $\dot{\boldsymbol{x}} = \boldsymbol{0}$.

From (3.14) it can be easily seen that $v_Z^* = p^* = q^* = r^* = 0$. From the last three equations we see that all the propellers speed must be the same to avoid angular accelerations (i.e. $\Omega_1^* = \Omega_2^* = \Omega_3^* = \Omega_4^* = \Omega^*$).

Finally, from the second equation of (3.14) we see that the lift force applied by the propellers must compensate the force of gravity:

$$\Omega^* = \frac{1}{2}\sqrt{\frac{m\ g}{b\ \cos\theta^d \cos\varphi^d}} \tag{3.15}$$

Remark 3.2. The equilibrium input shown in (3.15) is valid for any θ^d and φ^d different from $\pm\pi/2$. The equilibrium of the AA model (3.14) can be achieved without satisfying the hovering condition (i.e. being θ and φ not zero), because this condition is not required in order to have null vertical acceleration and null rotation accelerations. However, hovering condition is required in the complete model (3.9), because it is in equilibrium (without horizontal acceleration) only if $\theta^* = \varphi^* = 0$. Then, the equilibrium input for the complete model becomes $\Omega^* = \Omega_H$, computed as:

$$\Omega_H = \frac{1}{2}\sqrt{\frac{m\ g}{b}} \tag{3.16}$$

□

For any desired height Z^d and orientation $(\varphi^d, \theta^d, \psi^d)$, the equilibrium state $\boldsymbol{x^*}$ and input $\boldsymbol{u^*}$ is:

$$\boldsymbol{x^*} = \begin{bmatrix} Z^d & 0 & \varphi^d & \theta^d & \psi^d & 0 & 0 & 0 \end{bmatrix}^T, \quad \boldsymbol{u^*} = \Omega^* \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^T \tag{3.17}$$

3.2 Quadrotor Takagi-Sugeno model

Given the state space non-linear equations (3.14) of Altitude/Attitude model of the quadrotor, the idea is to obtain a set of LTI subsystems using the local sector non-linearity approach. The first step is to write equations (3.14) on the following linear form:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(\boldsymbol{z}(t))\ \boldsymbol{x}(t) + \boldsymbol{B}(\boldsymbol{z}(t))\ \boldsymbol{u}(t) \tag{3.18}$$

where $\boldsymbol{x}(t)$ and $\boldsymbol{u}(t)$ are the state and input vectors shown in (3.12). Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are not constant but depend on some premise variables $\boldsymbol{z}(t)$, which at the same time depends on the states or the inputs.

In order to find matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, the following observations has been considered:

- It is a good practice to try to have as many components as possible in matrix $\boldsymbol{A}$. A term with a product of two state/input variables can be split in two terms, so that each term is the component related with each state/input variable. For example, if x_1 and x_2 are two state variables and the term is ax_1x_2, with 'a' being constant, then it can be written as $(\frac{1}{2}ax_2)\ x_1 + (\frac{1}{2}ax_1)\ x_2$.
- There is an independent constant term '$-g$' in the second equation of (3.14) that should be multiplied by any state variable. One solution is to multiply and divide that term by the first state variable $Z(t)$.
- The non-linear terms Ω_i^2 can be split in two, so that one Ω_i is introduced in the matrix as a (variable) parameter.

From (3.14) and applying the observations just mentioned above, matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are:

$$\boldsymbol{A}(\boldsymbol{z}(t)) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{21}(t) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{67}(t) & a_{68}(t) \\ 0 & 0 & 0 & 0 & 0 & a_{76}(t) & 0 & a_{78}(t) \\ 0 & 0 & 0 & 0 & 0 & a_{86}(t) & a_{87}(t) & 0 \end{bmatrix} \tag{3.19}$$

$$\boldsymbol{B}(\boldsymbol{z}(t)) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ b_{21}(t) & b_{22}(t) & b_{23}(t) & b_{24}(t) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b_{61}(t) & b_{62}(t) & b_{63}(t) & b_{64}(t) \\ b_{71}(t) & b_{72}(t) & b_{73}(t) & b_{74}(t) \\ b_{81}(t) & b_{82}(t) & b_{83}(t) & b_{84}(t) \end{bmatrix} \tag{3.20}$$

Where the components of (3.19) and (3.20) are:

$$
\begin{aligned}
a_{21}(t) &= -\frac{g}{Z}, & a_{67}(t) &= \frac{I_Y - I_Z}{2I_X}\ r, & a_{68}(t) &= \frac{I_Y - I_Z}{2I_X}\ q \\
a_{76}(t) &= \frac{I_Z - I_X}{2I_Y}\ r, & a_{78}(t) &= \frac{I_Z - I_X}{2I_Y}\ p, & a_{86}(t) &= \frac{I_X - I_Y}{2I_Z}\ q \\
a_{87}(t) &= \frac{I_X - I_Y}{2I_Z}\ p, & b_{21}(t) &= \frac{b}{m}\cos\theta\cos\varphi\ \Omega_1, & b_{22}(t) &= \frac{b}{m}\cos\theta\cos\varphi\ \Omega_2 \\
b_{23}(t) &= \frac{b}{m}\cos\theta\cos\varphi\ \Omega_3, & b_{24}(t) &= \frac{b}{m}\cos\theta\cos\varphi\ \Omega_4, & b_{61}(t) &= \frac{J_{TP}}{I_X}\ q \\
b_{62}(t) &= -\frac{J_{TP}}{I_X}\ q - \frac{bl}{I_X}\ \Omega_2 & b_{63}(t) &= \frac{J_{TP}}{I_X}\ q, & b_{64}(t) &= -\frac{J_{TP}}{I_X}\ q + \frac{bl}{I_X}\ \Omega_4 \\
b_{71}(t) &= -\frac{J_{TP}}{I_Y}\ p - \frac{bl}{I_Y}\ \Omega_1 & b_{72}(t) &= \frac{J_{TP}}{I_Y}\ p, & b_{73}(t) &= -\frac{J_{TP}}{I_Y}\ p + \frac{bl}{I_Y}\ \Omega_3 \\
b_{74}(t) &= \frac{J_{TP}}{I_Y}\ p & b_{81}(t) &= -\frac{d}{I_Z}\ \Omega_1, & b_{82}(t) &= \frac{d}{I_Z}\ \Omega_2 \\
b_{83}(t) &= -\frac{d}{I_Z}\ \Omega_3 & b_{84}(t) &= \frac{d}{I_Z}\ \Omega_4
\end{aligned}
\tag{3.21}
$$

Remark 3.3. Note that matrix $\boldsymbol{A}(\boldsymbol{z}(t))$ in (3.19) has three columns plenty of zeros. During the development of this project an alternative version was considered. In that version the third, fourth and fifth equations of (3.14) were changed as follows:

$$
\begin{cases}
\dot{\varphi} = \dfrac{p}{2\varphi}\ \varphi + \dfrac{1}{2}p \\
\dot{\theta} = \dfrac{q}{2\theta}\ \theta + \dfrac{1}{2}q \\
\dot{\psi} = \dfrac{r}{2\psi}\ \psi + \dfrac{1}{2}r
\end{cases}
$$

Therefore new parameters a_{33}, a_{44} and a_{55} are introduced. This is done to avoid numerical problems. However, this solution also increments the number of rules in the TS model, so it was decided to work with the simplest version shown in (3.19) (the one that implies less number of subsystems). □

The output system (i.e. matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ and their parameters) will be computed by a linear combination of subsystems, as it is shown in (2.2). The set of subsystems can be seen as vertices of a polytope, which should include all the realizable systems (the ones obtained by substituting a feasible state and input vectors in (3.21)). One conservative

way to achieve this is to take into account the subsystems obtained from combinations of upper and lower bounds of the parameters in (3.21). In fact, these parameters will be written as a function of some premise variables $\boldsymbol{z}(t)$. The bounds of these variables are the ones that will define the polytope of subsystems.

3.2.1 Premise variables

The premise variables are obtained by looking at the states and inputs variables that appear in (3.21). The constant values of the parameters has been omitted, so the premise variables are only the varying part. The set of twelve premise variables $\boldsymbol{z}(t)$ is

$$\begin{aligned} z_1 &= 1/Z, & z_2 &= p, & z_3 &= q \\ z_4 &= r, & z_5 &= \cos\theta\cos\varphi\ \Omega_1, & z_6 &= \cos\theta\cos\varphi\ \Omega_2 \\ z_7 &= \cos\theta\cos\varphi\ \Omega_3, & z_8 &= \cos\theta\cos\varphi\ \Omega_4, & z_9 &= \Omega_1 \\ z_{10} &= \Omega_2, & z_{11} &= \Omega_3, & z_{12} &= \Omega_4 \end{aligned} \tag{3.22}$$

Remark 3.4. The premise variables that would be obtained from b_{62}, b_{64}, b_{71} and b_{73} are not included because they are linear combinations of other premise variables (see Case 3 in Section A.3). These parameters are computed as functions of premise variables from (3.22) and they are not considered in the set of fuzzy rules. □

Regarding the bounds of the premise variables the following observations are made:

- The lower and upper bounds of the premise variables z_2, z_3, z_4, z_9, z_{10}, z_{11} and z_{12} are just the bounds of the states p, q, r and the inputs Ω_1, Ω_2, Ω_3 and Ω_4, respectively.
- The bounds of $Z \in [\underline{Z}\ \overline{Z}]$ does not include $Z = 0$ to avoid numerical problems in premise variable z_1. $\underline{Z}$ and $\overline{Z}$ are both positive.
- The bounds of the input $\Omega_i\ \forall i \in \{1, 2, 3, 4\}$ are two positive numbers. Each propeller always rotates in the directions shown in Figure 2.1.

- The interval for angles θ and φ are $-\overline{\theta} < \theta < \overline{\theta}$ and $-\overline{\varphi} < \varphi < \overline{\varphi}$, so they are centered at zero. The interval does not include $\pm\pi/2$ to avoid numerical problems due to zeros at premise variables from z_5 to z_8.

The complete set of bounds for the premise variables (3.22) is

$$\begin{array}{llll}
\underline{z}_1 = 1/\overline{Z}, & \overline{z}_1 = 1/\underline{Z}, & \underline{z}_2 = \underline{p}, & \overline{z}_2 = \overline{p}, \\
\underline{z}_3 = \underline{q}, & \overline{z}_3 = \overline{q}, & \underline{z}_4 = \underline{r}, & \overline{z}_4 = \overline{r} \\
\underline{z}_5 = \cos\overline{\theta}\cos\overline{\varphi}\ \underline{\Omega}_1, & \overline{z}_5 = \overline{\Omega}_1, & \underline{z}_6 = \cos\overline{\theta}\cos\overline{\varphi}\ \underline{\Omega}_2, & \overline{z}_6 = \overline{\Omega}_2 \\
\underline{z}_7 = \cos\overline{\theta}\cos\overline{\varphi}\ \underline{\Omega}_3, & \overline{z}_7 = \overline{\Omega}_3, & \underline{z}_8 = \cos\overline{\theta}\cos\overline{\varphi}\ \underline{\Omega}_4, & \overline{z}_8 = \overline{\Omega}_4 \\
\underline{z}_9 = \underline{\Omega}_1, & \overline{z}_9 = \overline{\Omega}_1, & \underline{z}_{10} = \underline{\Omega}_2, & \overline{z}_{10} = \overline{\Omega}_2 \\
\underline{z}_{11} = \underline{\Omega}_3, & \overline{z}_{11} = \overline{\Omega}_3, & \underline{z}_{12} = \underline{\Omega}_4, & \overline{z}_{12} = \overline{\Omega}_4
\end{array} \tag{3.23}$$

Remark 3.5. As it is commented above, the interval for Z does not include $Z = 0$. In case we want the operating point to be at $Z^d = 0$, the system can be controlled at some positive value Z^d inside the interval, and then the E-frame can be translated in Z_E direction so that the new operating point becomes zero. The range of operation of $Z \in [\underline{Z},\ \overline{Z}]$ does not affect the dynamic of the system which, as it is seen in (3.14), is independent of Z position. □

From (3.22) it can be seen that some premise variables are not independent. In particular $z_i = \cos\theta\cos\varphi\ z_{i+4}$ for $i \in \{5, 6, 7, 8\}$. Note that for any fixed angles θ and φ, the relation between z_i and z_{i+4} is a straight line with slope $\beta(\theta, \varphi) = \cos\theta\cos\varphi$. Figure 3.2 shows the region of feasible values (gray area) for the pair of premise variables z_5 and z_9, but the figure is equivalent for the other pairs of variables $\{z_i,\ z_{i+4}\}$ for $i \in \{5, 6, 7, 8\}$.

3.2.2 Membership and activation functions

Assuming that all the premise variables in (3.22) belongs to Case 1 in Section A.4 (i.e. the number of fuzzy rules cannot be reduced), then there are two membership functions M_{i1} and M_{i2} for each premise variable i. All these pairs of membership functions satisfies equations (A.13) and have the form shown in (A.14). They are plotted in Figure A.4.

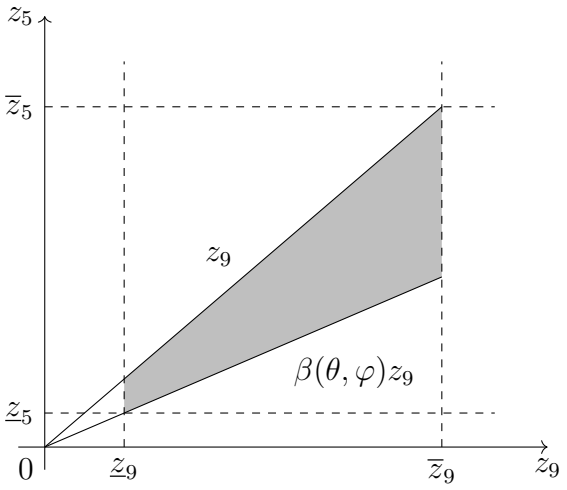


FIGURE 3.2: Relation between premise variables z_5 and z_9

Remark 3.6. The membership functions shown in (2.1) follow the same notation than in [3], where M_{ij} represents the fuzzy set of premise variable j in rule i. However, from now on a new notation is considered where M_{ij} represents the minimal ($j = 1$) or maximal ($j = 2$) fuzzy set of the premise variable i. □

Since all the vertices of the polytope are considered, there are $2^{12} = 4096$ subsystems. The conditions of the fuzzy rules (not all depicted here for obvious reasons) are the following:

Rule 1: z_1 is M_{11} and z_2 is M_{21} and ... and z_{12} is M_{121}

Rule 2: z_1 is M_{11} and z_2 is M_{21} and ... and z_{12} is M_{122}

⋮

Rule 2047: z_1 is M_{11} and z_2 is M_{22} and ... and z_{12} is M_{121}

Rule 2048: z_1 is M_{11} and z_2 is M_{22} and ... and z_{12} is M_{122}

Rule 2049: z_1 is M_{12} and z_2 is M_{21} and ... and z_{12} is M_{121}

Rule 2050: z_1 is M_{12} and z_2 is M_{21} and ... and z_{12} is M_{122}

⋮

Rule 4095: z_1 is M_{12} and z_2 is M_{22} and ... and z_{12} is M_{121}

Rule 4096: z_1 is M_{12} and z_2 is M_{22} and ... and z_{12} is M_{122}

As it is explained in Remark 2.2, the activation function $h_i(\boldsymbol{z}(t))$ for each subsystem can be computed as the product of normalized membership functions:

$$\begin{cases} h_1(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdots M_{111} \cdot M_{121} \\ h_2(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdots M_{111} \cdot M_{122} \\ \vdots \\ h_{2047}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdots M_{112} \cdot M_{121} \\ h_{2048}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdots M_{112} \cdot M_{122} \\ h_{2049}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdots M_{111} \cdot M_{121} \\ h_{2050}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdots M_{111} \cdot M_{122} \\ \vdots \\ h_{4095}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdots M_{112} \cdot M_{121} \\ h_{4096}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdots M_{112} \cdot M_{122} \end{cases} \tag{3.24}$$

Remark 3.7. The design of the controller can be done offline so the drawback related to the computational time due to the big number of rules it is not critical. However, as it will be seen it could affect the chances of finding a solution for the controller. For that reason it is considered the hypothesis that the pairs of premise variables (z_5, z_9), (z_6, z_{10}), (z_7, z_{11}) and (z_8, z_{12}) can be classified as Case 2 in Section A.4. As it is seen in Figure 3.2 this is not true, but the number of rules/subsystems would be reduced and this model has been validated (see Section 3.4). □

As discussed before, there are two membership functions for each premise variable from z_1 to z_4. If the pairs of premise variables (z_5, z_9), (z_6, z_{10}), (z_7, z_{11}) and (z_8, z_{12}) are labeled as $j = \{1, 2, 3, 4\}$ respectively, and assuming the hypothesis of Remark 3.7, then there are three membership functions N_{j1}, N_{j2}, N_{j3} (see 'triangular polytope' in Section A.4) for each pair j. The number of fuzzy rules and subsystems is computed then as $3^4 \cdot 2^4 = 1296$. The rules would have the following form:

Rule 1: z_1 is M_{11} and z_2 is M_{21} and ... and (z_8, z_{12}) is N_{41}

Rule 2: z_1 is M_{11} and z_2 is M_{21} and ... and (z_8, z_{12}) is N_{42}

Rule 3: z_1 is M_{11} and z_2 is M_{21} and ... and (z_8, z_{12}) is N_{43}

$\vdots$

Rule 646: z_1 is M_{11} and z_2 is M_{22} and ... and (z_8, z_{12}) is N_{41}

Rule 647: z_1 is M_{11} and z_2 is M_{22} and ... and (z_8, z_{12}) is N_{42}

Rule 648: z_1 is M_{11} and z_2 is M_{22} and ... and (z_8, z_{12}) is N_{43}

Rule 649: z_1 is M_{12} and z_2 is M_{21} and ... and (z_8, z_{12}) is N_{41}

Rule 650: z_1 is M_{12} and z_2 is M_{21} and ... and (z_8, z_{12}) is N_{42}

Rule 651: z_1 is M_{12} and z_2 is M_{21} and ... and (z_8, z_{12}) is N_{43}

$\vdots$

Rule 1294: z_1 is M_{12} and z_2 is M_{22} and ... and (z_8, z_{12}) is N_{41}

Rule 1295: z_1 is M_{12} and z_2 is M_{22} and ... and (z_8, z_{12}) is N_{42}

Rule 1296: z_1 is M_{12} and z_2 is M_{22} and ... and (z_8, z_{12}) is N_{43}

And again the activation function $h_i(\boldsymbol{z}(t))$ for each subsystem can be computed as follows:

$$
\begin{cases}
h_1(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdots N_{31} \cdot N_{41} \\
h_2(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdots N_{31} \cdot N_{42} \\
h_3(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdots N_{31} \cdot N_{43} \\
\vdots \\
h_{646}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdots N_{33} \cdot N_{41} \\
h_{647}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdots N_{33} \cdot N_{42} \\
h_{648}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdots N_{33} \cdot N_{43} \\
h_{649}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdots N_{31} \cdot N_{41} \\
h_{650}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdots N_{31} \cdot N_{42} \\
h_{651}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdots N_{31} \cdot N_{43} \\
\vdots \\
h_{1294}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdots N_{33} \cdot N_{41} \\
h_{1295}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdots N_{33} \cdot N_{42} \\
h_{1296}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdots N_{33} \cdot N_{43}
\end{cases}
\tag{3.25}
$$

3.3 Quadrotor Takagi-Sugeno model with linear input approximation

In the previous section the Takagi-Sugeno model of (3.14) has been derived. However, even taking into account some hypothesis that reduce the number of fuzzy rules (Remark 3.4 and Remark 3.7) there still being 1296 rules/subsystems. The dimension of the problem makes difficult the design of the controller, so a new model has been derived with the aim of reducing the number of premise variables.

The new approach is based on a combination of local sector non-linearity method as applied before, and local linear approximation method. The idea is to approximate the quadratic input terms Ω_i^2 by linear functions. This linear approximation of the input has been motivated by two facts, in addition to the reduction of premise variables. On one hand, it was assumed that the quadrotor operates close to the hovering condition, and as

it is explained in [7, p. 10], the deviation of the propeller speeds from the hovering value Ω_H should not be very large to avoid strong non-linearities or saturations. On the other hand in [3, p. 6] it is recommended not to include the input variables in the parameters of the TS model. This is done to avoid problems in the defuzzification process of controllers when the premise variables are functions of the input.

Let consider the input propeller speed Ω_i for any $i \in \{1, 2, 3, 4\}$. Then, Ω_i^2 can be linearized around Ω_H as it is shown in (3.26). Note that in order to have a positive value it should be satisfied that $\Omega_i > \Omega_H/2$ (see Figure 3.3). If Ω_i is inside an interval $\Omega_H \pm 50\%$, then the approximated value of the square is positive

$$\Omega_i^2 \approx 2\ \Omega_H\ \Omega_i - \Omega_H^2 \tag{3.26}$$

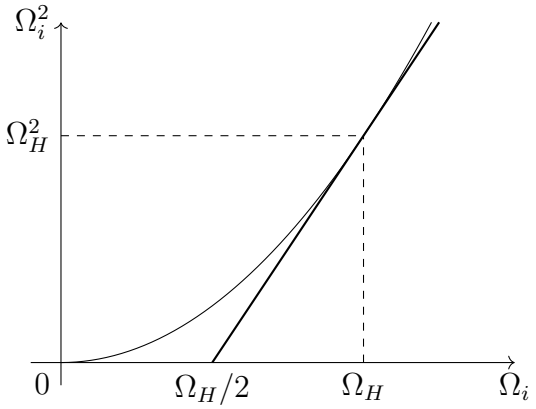


FIGURE 3.3: Linear approximation of the squares of propellers speeds

Substituting the Ω_i^2 approximated expression from (3.26) and the definition of Ω_H (3.16) in the AA model (3.14), the new model of the quadrotor with linear input approximation is (3.27). The new input force and torques are shown in (3.28).

$$\begin{cases} \dot{z} = v_Z \\ \dot{v_Z} = -g\ (\cos\theta\cos\varphi + 1) + \dfrac{b}{m}\ \cos\theta\cos\varphi\ 2\Omega_H\ (\Omega_1 + \Omega_2 + \Omega_3 + \Omega_4) \\ \dot{\varphi} = p \\ \dot{\theta} = q \\ \dot{\psi} = r \\ \dot{p} = \dfrac{I_Y - I_Z}{I_X}\ q\ r - \dfrac{J_{TP}}{I_X}\ q\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \dfrac{bl}{I_X}\ 2\Omega_H\ (-\Omega_2 + \Omega_4) \\ \dot{q} = \dfrac{I_Z - I_X}{I_Y}\ p\ r + \dfrac{J_{TP}}{I_Y}\ p\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \dfrac{bl}{I_Y}\ 2\Omega_H\ (-\Omega_1 + \Omega_3) \\ \dot{r} = \dfrac{I_X - I_Y}{I_Z}\ p\ q + \dfrac{d}{I_Z}\ \ 2\Omega_H\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) \end{cases} \tag{3.27}$$

$$\begin{cases} U_1 = b\ (2\Omega_H\ (\Omega_1 + \Omega_2 + \Omega_3 + \Omega_4) - 4\Omega_H^2) \\ U_2 = bl\ 2\Omega_H\ (-\Omega_2 + \Omega_4) \\ U_3 = bl\ 2\Omega_H\ (-\Omega_1 + \Omega_3) \\ U_4 = d\ 2\Omega_H\ (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) \end{cases} \tag{3.28}$$

The procedure to obtain the TS-LIA (Takagi-Sugeno with Linear Input Approximation) model is equivalent to the followed for the previous TS model. The independent term of the second equation in (3.27) has been multiplied and divided by the state Z as before, and also the terms with products of two state variables has been separated in two. The amount of parameters and their locations in matrices $\boldsymbol{A}(\boldsymbol{z}(t))$ and $\boldsymbol{B}(\boldsymbol{z}(t))$ are the same than in (3.19) and (3.20), respectively. However, the values (3.21) have changed to the ones shown in (3.29). Note that the parameters are independent of the input vector $\boldsymbol{\Omega}$.

$$
\begin{aligned}
&a_{21}(t) = -\frac{g(1+\cos\theta\cos\varphi)}{Z}, && a_{67}(t) = \frac{I_Y - I_Z}{2I_X}\, r, && a_{68}(t) = \frac{I_Y - I_Z}{2I_X}\, q \\
&a_{76}(t) = \frac{I_Z - I_X}{2I_Y}\, r, && a_{78}(t) = \frac{I_Z - I_X}{2I_Y}\, p, && a_{86}(t) = \frac{I_X - I_Y}{2I_Z}\, q \\
&a_{87}(t) = \frac{I_X - I_Y}{2I_Z}\, p, && b_{21}(t) = \frac{b}{m}\cos\theta\cos\varphi\ \Omega_1, && b_{22}(t) = b_{21}(t) \\
&b_{23}(t) = b_{21}(t), && b_{24}(t) = b_{21}(t), && b_{61}(t) = \frac{J_{TP}}{I_X}\, q \\
&b_{62}(t) = -\frac{J_{TP}}{I_X}\, q - \frac{bl}{I_X}\, 2\Omega_H && b_{63}(t) = b_{61}(t), && b_{64}(t) = -\frac{J_{TP}}{I_X}\, q + \frac{bl}{I_X}\, 2\Omega_H \\
&b_{71}(t) = -\frac{J_{TP}}{I_Y}\, p - \frac{bl}{I_Y}\, 2\Omega_H && b_{72}(t) = \frac{J_{TP}}{I_Y}\, p, && b_{73}(t) = -\frac{J_{TP}}{I_Y}\, p + \frac{bl}{I_Y}\, 2\Omega_H \\
&b_{74}(t) = b_{72}(t), && b_{81}(t) = -\frac{d}{I_Z}\, 2\Omega_H, && b_{82}(t) = -b_{81}(t) \\
&b_{83}(t) = b_{81}(t), && b_{84}(t) = -b_{81}(t)
\end{aligned}
\tag{3.29}
$$

3.3.1 Premise variables

In this case there are five premise variables (3.30), taking into account that the ones obtained from $b_{62}(t)$, $b_{64}(t)$, $b_{71}(t)$ and $b_{73}(t)$ are linear combination of others (see Remark 3.4). Note that z_1 and z_5 are not independent, so a reduction in the number of fuzzy rules was considered. However, as it is seen in Figure 3.4, that pair of variables can not be included in the triangular polytope case (see Section A.3).

$$
\begin{aligned}
&z_1 = \frac{1+\cos\theta\cos\varphi}{Z}, && z_2 = p, && z_3 = q \\
&z_4 = r, && z_5 = \cos\theta\cos\varphi
\end{aligned}
\tag{3.30}
$$

Finally, the bounds of the premise variables are

$$
\begin{aligned}
&\underline{z}_1 = \frac{1+\cos\overline{\theta}\cos\overline{\varphi}}{\overline{Z}}, && \overline{z}_1 = \frac{2}{\underline{Z}}, && \underline{z}_2 = \underline{p}, && \overline{z}_2 = \overline{p} \\
&\underline{z}_3 = \underline{q}, && \overline{z}_3 = \overline{q}, && \underline{z}_4 = \underline{r}, && \overline{z}_4 = \overline{r}, \\
&\underline{z}_5 = \cos\overline{\theta}\cos\overline{\varphi}, && \overline{z}_5 = 1
\end{aligned}
\tag{3.31}
$$

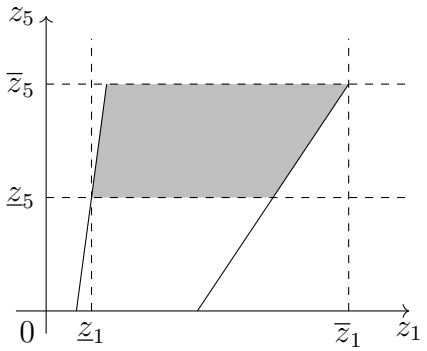


FIGURE 3.4: Relation between premise variables z_1 and z_5 in the TS-LIA model

3.3.2 Membership and activation functions

There are two membership functions M_{i1} and M_{i2} for each premise variable i, and they are computed as it is shown in (A.14). Since the number of membership functions is five and all the combinations of bounds of the premise variables are considered, the number of fuzzy rules/subsystems of the TS-LIA model is $2^5 = 32$.

Rule 1: z_1 is M_{11} and z_2 is M_{21} and z_3 is M_{31} z_4 is M_{41} and z_5 is M_{51}

Rule 2: z_1 is M_{11} and z_2 is M_{21} and z_3 is M_{31} z_4 is M_{41} and z_5 is M_{52}

$\vdots$

Rule 14: z_1 is M_{11} and z_2 is M_{22} and z_3 is M_{32} z_4 is M_{42} and z_5 is M_{51}

Rule 15: z_1 is M_{11} and z_2 is M_{22} and z_3 is M_{32} z_4 is M_{42} and z_5 is M_{52}

Rule 16: z_1 is M_{12} and z_2 is M_{21} and z_3 is M_{31} z_4 is M_{41} and z_5 is M_{51}

Rule 17: z_1 is M_{12} and z_2 is M_{21} and z_3 is M_{31} z_4 is M_{41} and z_5 is M_{52}

$\vdots$

Rule 31: z_1 is M_{12} and z_2 is M_{22} and z_3 is M_{32} z_4 is M_{42} and z_5 is M_{51}

Rule 32: z_1 is M_{12} and z_2 is M_{22} and z_3 is M_{32} z_4 is M_{42} and z_5 is M_{52}

The computation of activation functions $h_i(\boldsymbol{z}(t))$, based on the product of all the combinations of membership functions, is also equivalent to the previous TS model:

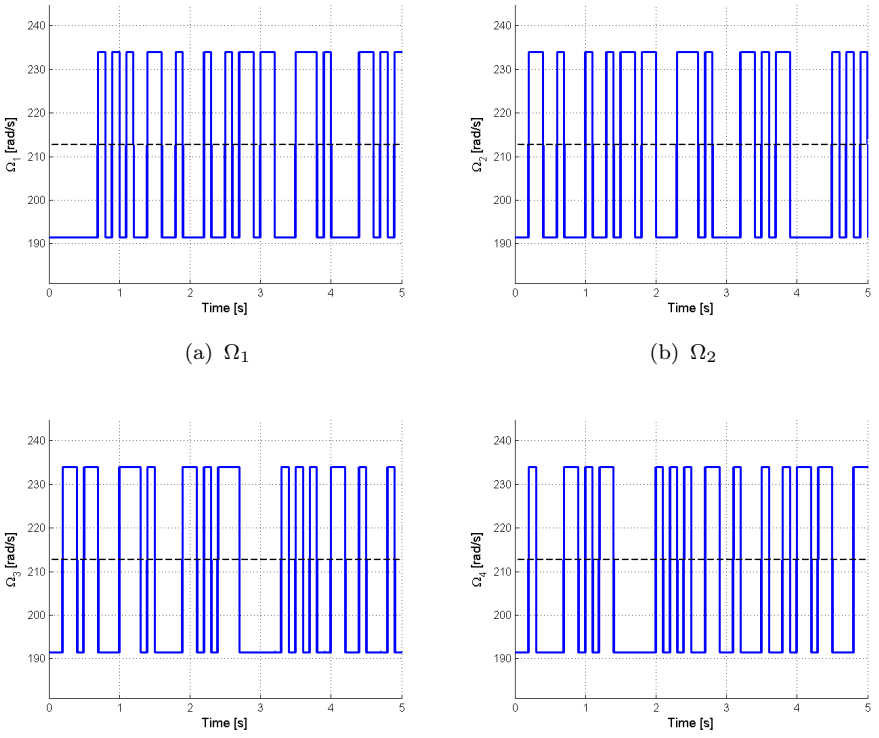
$$\begin{cases} h_1(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdot M_{31} \cdot M_{41} \cdot M_{51} \\ h_2(\boldsymbol{z}(t)) = M_{11} \cdot M_{21} \cdot M_{31} \cdot M_{41} \cdot M_{52} \\ \vdots \\ h_{14}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdot M_{32} \cdot M_{42} \cdot M_{51} \\ h_{15}(\boldsymbol{z}(t)) = M_{11} \cdot M_{22} \cdot M_{32} \cdot M_{42} \cdot M_{52} \\ h_{16}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdot M_{31} \cdot M_{41} \cdot M_{51} \\ h_{17}(\boldsymbol{z}(t)) = M_{12} \cdot M_{21} \cdot M_{31} \cdot M_{41} \cdot M_{52} \\ \vdots \\ h_{31}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdot M_{32} \cdot M_{42} \cdot M_{51} \\ h_{32}(\boldsymbol{z}(t)) = M_{12} \cdot M_{22} \cdot M_{32} \cdot M_{42} \cdot M_{52} \end{cases} \tag{3.32}$$

3.4 Validation of TS models

Three different models has been built in Simulink® for validation tests: the non-linear Altitude-Attitude (AA) model, the TS model and the TS-LIA model. The simulation of the non-linear AA model is done just by implementing the differential equations (3.14). The simulation of the last two Takagi-Sugeno models is done by implementing the linear system (2.2), where the set of subsystems are computed offline using the bounds of premise variables (3.23) or (3.31) for each case. The activation functions $h_i(\boldsymbol{z}(t))$ are obtained from (3.25) or (3.32) by first computing the premise variables from the states ((3.22) or (3.30)), and then obtaining the membership functions from the premise variables. The goal is to analyze if the TS models are equivalent to the AA model, and also TS and TS-LIA models are similar between them in spite of the approximation.

The same input signal $\boldsymbol{u} = [\Omega_1 \ \Omega_2 \ \Omega_3 \ \Omega_4]^T$ has been applied to all three models. The type of input signal used for validation is a Pseudo-Random Binary Signal (PRBS), the same used in [5]. PRBS is a periodic, deterministic signal with autocorrelation function similar to a white noise signal [8]. It allows the excitation of the system in a wide range of frequencies and it has only two possible values, so its a sequence of pulses of variable width. The minimum duration of each pulse has been selected to be 0.1 s. Figure 3.5

shows the PRBS inputs applied for the validation test, obtained form Matlab function *idinput*().



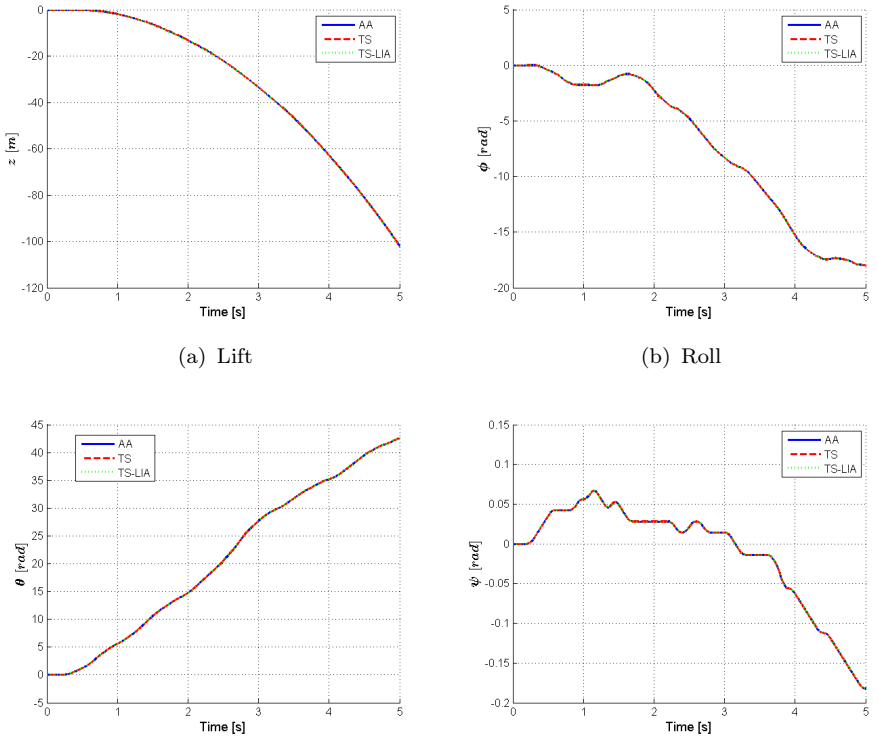
(a) Ω_1 (b) Ω_2

(c) Ω_3 (d) Ω_4

FIGURE 3.5: PRBS input signal for validation

Figure 3.6 shows the output height z and angular positions φ, θ and ψ when the validation PRBS input is applied. The values for the parameters and bounds of the states used in these simulations appears in Section A.2. Apparently there is not a significant difference between the models.

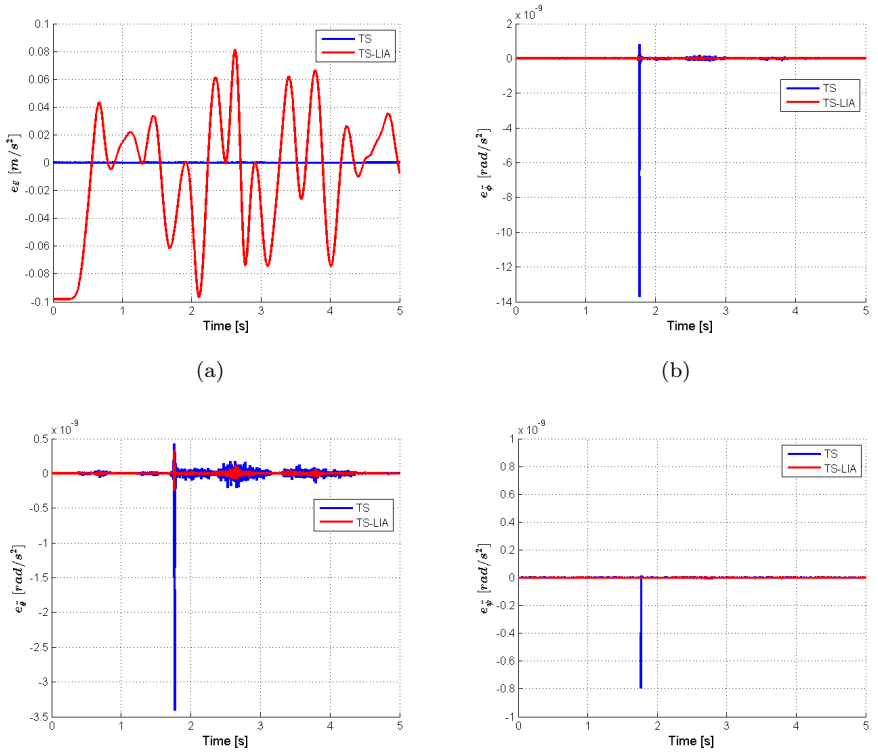
Let define the errors $e_{\ddot{z}}$, $e_{\ddot{\varphi}}$, $e_{\ddot{\theta}}$ and $e_{\ddot{\psi}}$ that measures the difference between the acceleration obtained form any of the Takagi-Sugeno models and the original AA model. In order to compare the two Takagi-Sugeno models (TS and TS-LIA), these acceleration errors have been computed and plotted in Figure 3.7. Note that regarding the TS model the errors are not significantly different from zero. The same occurs with the TS-LIA model in the case of angular accelerations. The error regarding the acceleration of $\ddot{z}$ (Figure 3.7 (a)) is more important for that model. However, as it can be seen in Figure 3.6 (a), this error does not imply a big difference in the behaviour of the model.



(a) Lift (b) Roll

(c) Pitch (d) Yaw

FIGURE 3.6: Output positions from AA, TS and TS-LIA models with a PRBS input



(a) (b)

(c) (d)

FIGURE 3.7: Acceleration errors between TS models and the AA model

Chapter 4

Attitude/Altitude control

The goal of this chapter is to design a state feedback controller for altitude and attitude using the TS-LIA model found in the previous chapter. In the first part the theoretical design of the state feedback controller and the state observer is described. In order to make the quadrotor follow variable references, two different control schemes are presented in the next sections: the FeedForward (FF) Control scheme and the Reference Model based FeedForward (RM-FF) Control scheme.

4.1 State feedback control and state observer

In this section, a fuzzy controller is designed based on the Parallel Distributed Compensation approach (see Section 2.3), so that the closed-loop system (2.6) is stable and has the poles in some LMI region to achieve the desired performance (quadratic $\mathbb{D}$-stability). Then, an state observer is designed assuming that only the position Z and the orientation $\mathbf{\Theta}$ variables are known.

4.1.1 Apkarian filter

As it is explained in Section 2.4.1, the number of LMI constraints needed to check quadratic stability is reduced if all the subsystems in the polytopic model has the same

matrix $\boldsymbol{B}$. This can be achieved by adding an Apkarian filter in the input of the system.

Let consider our TS-LIA model with equations shown in (3.27). This model can be written in linear form as it is shown in (3.18) and repeated below.

$$\dot{\boldsymbol{x}} = \boldsymbol{A}(\boldsymbol{z}(t))\ \boldsymbol{x} + \boldsymbol{B}(\boldsymbol{z}(t))\ \boldsymbol{u} \tag{4.1}$$

The filter should be such that the equilibrium of the states are the input values and the dynamics should be fast, so we could assume the dynamics of the filter negligible (i.e. the input of the filter is equivalent to the input of the quadrotor). One possible filter is shown in (4.2), where $\boldsymbol{A}_F = -100 \cdot \boldsymbol{I}_4$, $\boldsymbol{B}_F = 100 \cdot \boldsymbol{I}_4$ and $\boldsymbol{I}_4 \in \mathbb{R}^{4x4}$ is the identity matrix.

$$\begin{cases} \dot{\boldsymbol{x}}_F = \boldsymbol{A}_F\ \boldsymbol{x}_F + \boldsymbol{B}_F\ \boldsymbol{u}_F \\ \boldsymbol{y}_F = \boldsymbol{x}_F \end{cases} \tag{4.2}$$

When applying the filter, we are imposing that the output of the filter is the new input of the TS-LIA model (i.e. $\boldsymbol{u} = \boldsymbol{y}_F$). Then, the extended model is (4.3), the new input is $\boldsymbol{u}_F$ and the new state vector is $\boldsymbol{x}_e$ (4.4). Note that now matrix $\boldsymbol{B}_e$ is constant.

$$\dot{\boldsymbol{x}}_e = \begin{bmatrix} \boldsymbol{A}(\boldsymbol{z}(t)) & \boldsymbol{B}(\boldsymbol{z}(t)) \\ \boldsymbol{0} & \boldsymbol{A}_F \end{bmatrix} \boldsymbol{x}_e + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_F \end{bmatrix} \boldsymbol{u}_F = \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_e + \boldsymbol{B}_e\ \boldsymbol{u}_F \tag{4.3}$$

$$\boldsymbol{x}_e = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_F \end{bmatrix} \tag{4.4}$$

This prefiltering does not affect the procedure followed to obtain the TS-LIA model, so the premise variables, membership functions and activations functions remains the same. The extended TS-LIA model is shown in (4.5), where matrices $\boldsymbol{A}_{ei}$ and $\boldsymbol{B}_{ei}$ form LTI subsystem i-th are (4.6).

$$\dot{\boldsymbol{x}}_e = \sum_{i=1}^{32} h_i(\boldsymbol{z}(t)) \left\{ \boldsymbol{A_{ei}}\ \boldsymbol{x}_e + \boldsymbol{B_{ei}}\ \boldsymbol{u}_F \right\} \tag{4.5}$$

$$\boldsymbol{A}_{ei} = \begin{bmatrix} \boldsymbol{A}_i & \boldsymbol{B}_i \\ \boldsymbol{0} & \boldsymbol{A}_F \end{bmatrix}, \quad \boldsymbol{B}_{ei} = \boldsymbol{B}_e = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_F \end{bmatrix} \quad \forall i = 1, \ldots, 32 \tag{4.6}$$

4.1.2 State feedback controller design

Let consider the state feedback control law (4.7) for the extended TS-LIA model (4.5), where the activation functions are (3.32).

$$\boldsymbol{u}_F = \sum_{i=1}^{32} h_i(\boldsymbol{z}(t))\ \boldsymbol{K}_i\ \boldsymbol{x}_e \tag{4.7}$$

By substituting (4.7) into (4.5) the following closed loop system is obtained:

$$\dot{\boldsymbol{x}}_e = \sum_{i=1}^{32}\sum_{j=1}^{32} h_i(\boldsymbol{z}(t))h_j(\boldsymbol{z}(t))\left\{\boldsymbol{A_{ei}} + \boldsymbol{B_e}\ \boldsymbol{K}_j\right\}\ \boldsymbol{x}_e \tag{4.8}$$

As it is explained in Section 2.4, the design of the controller is done by solving an LMI problem involving the quadratic stability constraints (2.12). In case we want $\mathbb{D}$-stabilization, the set of LMI constraints (4.9) are needed, and they depend on the LMI region (2.14) where we want to place the poles.

$$\boldsymbol{L} \otimes \boldsymbol{P} + \boldsymbol{M} \otimes \boldsymbol{P}(\boldsymbol{A}_{ei} + \boldsymbol{B}_e\boldsymbol{K}_i) + \boldsymbol{M}^T \otimes (\boldsymbol{A}_{ei} + \boldsymbol{B}_e\boldsymbol{K}_i)^T\boldsymbol{P} < 0 \quad \forall i = 1, \ldots, 32 \tag{4.9}$$

A pair of conjugate complex poles s of the closed loop system can be written as $s = -\xi\omega_n \pm j\omega_d$ where ξ is the damping ratio, ω_n is the undamped natural frequency and ω_d is the frequency response defined as $\omega_d = \omega_n\ \sqrt{1-\xi^2}$. Three different LMI regions has been considered, each one related with a performance specification regarding $\alpha = \xi\omega_n$, ω_n and ξ:

- **Minimum decay rate** α: If we want to set a minimum decay rate α in the closed loop system response, the poles should be inside the LMI region defined in (4.10),

where $\alpha > 0$. According to the general definition of an LMI region (2.14), in this case $L_\alpha = 2\alpha$ and $M_\alpha = 1$ [6, p. 103].

$$\mathbb{S}_\alpha = \{s = x + jy \mid x < -\alpha\} \tag{4.10}$$

Applying condition (4.9) to the closed-loop system (4.8), the LMI condition associated to this LMI region is

$$2\alpha \boldsymbol{P} + (\boldsymbol{A}_{ei} + \boldsymbol{B}_e \boldsymbol{K}_i)^T \boldsymbol{P} + \boldsymbol{P}(\boldsymbol{A}_{ei} + \boldsymbol{B}_e \boldsymbol{K}_i) < 0 \quad \forall i = 1, \ldots, 32 \tag{4.11}$$

- **Maximum natural frequency** ω_n: Natural frequency is related with the maximum frequency response in the undamped case ($\xi = 0$). If we want to set a maximum ω_n condition, the LMI region associated is (4.12), where L_ρ and M_ρ are (4.13).

$$\mathbb{S}_\rho = \{s = x + jy \mid |x + jy| < \rho\} \tag{4.12}$$

$$L_\rho = \begin{bmatrix} -\rho & 0 \\ 0 & -\rho \end{bmatrix}, \quad M_\rho = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{4.13}$$

Substituting (4.13) in (4.9), the LMI condition associated to this LMI region is

$$\begin{bmatrix} -\rho \boldsymbol{P} & \boldsymbol{P}(\boldsymbol{A}_{ei} + \boldsymbol{B}_e \boldsymbol{K}_i) \\ (\boldsymbol{A}_{ei} + \boldsymbol{B}_e \boldsymbol{K}_i)^T \boldsymbol{P} & -\rho \boldsymbol{P} \end{bmatrix} < 0 \quad \forall i = 1, \ldots, 32 \tag{4.14}$$

- **Minimum damping ratio** ξ: The damping ratio determines how oscillatory the evolution of the state in the closed-loop system can be. This factor is related with the angle γ by the formula $\xi = \cos\gamma$ (see Figure 4.1). In order to set a minimum damping factor the following LMI region is defined:

$$\mathbb{S}_\gamma = \{s = x + jy \mid |y| < -x \tan(\gamma)\} \tag{4.15}$$

In this case matrices L_γ and M_γ are [6, p. 105]:

$$L_\gamma = 0, \quad M_\gamma = \begin{bmatrix} \sin(\gamma) & \cos(\gamma) \\ -\cos(\gamma) & \sin(\gamma) \end{bmatrix} \tag{4.16}$$

As in the previous cases, by substituting matrices (4.16) in (4.9) the LMI condition associated to this LMI region is obtained

$$\begin{bmatrix} (\boldsymbol{P}\boldsymbol{G}_i + \boldsymbol{G}_i^T\boldsymbol{P})\sin\gamma & (\boldsymbol{P}\boldsymbol{G}_i - \boldsymbol{G}_i^T\boldsymbol{P})\cos\gamma \\ (\boldsymbol{G}_i^T\boldsymbol{P} - \boldsymbol{P}\boldsymbol{G}_i)\cos\gamma & (\boldsymbol{P}\boldsymbol{G}_i + \boldsymbol{G}_i^T\boldsymbol{P})\sin\gamma \end{bmatrix} < 0 \quad \forall i = 1, \ldots, 32 \tag{4.17}$$

where $\boldsymbol{G}_i = \boldsymbol{A}_{ei} + \boldsymbol{B}_e\boldsymbol{K}_i$.

The intersection of LMI regions is also an LMI region. A new LMI region is defined as $\mathbb{S}_{\alpha,\rho,\gamma} = \mathbb{S}_\alpha \cap \mathbb{S}_\rho \cap \mathbb{S}_\gamma$ (4.18). Figure 4.1 shows where the poles should be located in the complex plane to satisfy the performance conditions.

$$\mathbb{S}_{\alpha,\rho,\gamma} = \{s = x + jy \mid x < -\alpha < 0,\ |x + jy| < \rho,\ |y| < -x\tan(\gamma)\} \tag{4.18}$$

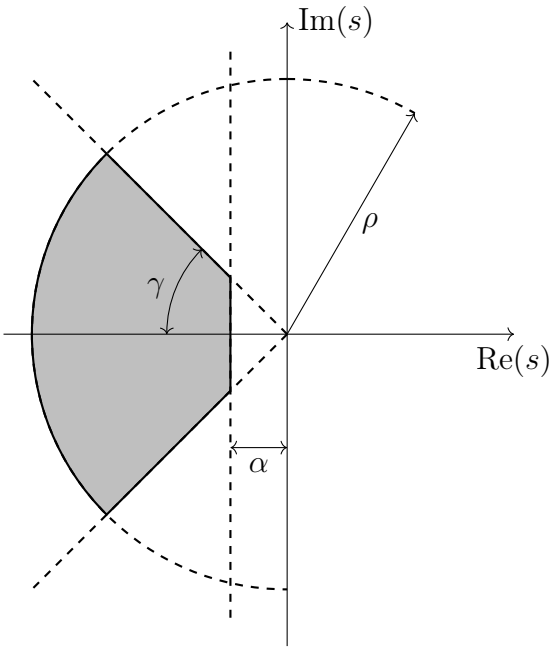


FIGURE 4.1: LMI region $\mathbb{S}_{\alpha,\rho,\gamma}$

As it is explained in Section 2.4, a change of variables is needed in conditions (4.11), (4.14) and (4.17) in order to have a LMI problem, so we define $\boldsymbol{W}_i = \boldsymbol{K}_i\boldsymbol{Q}$ and $\boldsymbol{Q} = \boldsymbol{P}^{-1}$.

With these variables, the LMI conditions becomes (4.19). The controllers are obtained from the set of solutions $\boldsymbol{W}_i$ by doing $\boldsymbol{K}_i = \boldsymbol{W}_i\boldsymbol{Q}^{-1}$.

$$\begin{cases} \boldsymbol{Q} > 0 \\ 2\alpha\boldsymbol{Q} + \boldsymbol{G}_{11i} < 0 \quad \forall i = 1, \ldots, 32 \\ \begin{bmatrix} -\rho\boldsymbol{Q} & \boldsymbol{A}_{ei}\boldsymbol{Q} + \boldsymbol{B}_e\boldsymbol{W}_i \\ \boldsymbol{Q}\boldsymbol{A}_{ei}^T + \boldsymbol{W}_i^T\boldsymbol{B}_e^T & -\rho\boldsymbol{Q} \end{bmatrix} < 0 \quad \forall i = 1, \ldots, 32 \\ \begin{bmatrix} \boldsymbol{G}_{11i}\sin\gamma & \boldsymbol{G}_{12i}\cos\gamma \\ \boldsymbol{G}_{21i}\cos\gamma & \boldsymbol{G}_{11i}\sin\gamma \end{bmatrix} < 0 \quad \forall i = 1, \ldots, 32 \end{cases} \tag{4.19}$$

Where $\boldsymbol{G}_{11i} = \boldsymbol{A}_{ei}\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A}_{ei}^T + \boldsymbol{B}_e\boldsymbol{W}_i + \boldsymbol{W}_i^T\boldsymbol{B}_e^T$, $\boldsymbol{G}_{12i} = \boldsymbol{A}_{ei}\boldsymbol{Q} - \boldsymbol{Q}\boldsymbol{A}_{ei}^T + \boldsymbol{B}_e\boldsymbol{W}_i - \boldsymbol{W}_i^T\boldsymbol{B}_e^T$ and $\boldsymbol{G}_{21i} = -\boldsymbol{A}_{ei}\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A}_{ei}^T - \boldsymbol{B}_e\boldsymbol{W}_i + \boldsymbol{W}_i^T\boldsymbol{B}_e^T$

4.1.3 State Observer design

The state feedback control designed before assumes that all the states are measurable. However, usually this is not the case and a state estimator is needed. If only positions measurements are provided (i.e. position Z and orientation angles φ, θ, ψ), a new output equation (4.20) is added to the TS-LIA extended model (4.5), where the output vector is $\boldsymbol{y} = [Z\ \varphi\ \theta\ \psi]^T$. Matrix $\boldsymbol{C}$ is the identity matrix in case all the states are measurable.

$$\boldsymbol{y} = \boldsymbol{C}\ \boldsymbol{x}_e = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}_e \tag{4.20}$$

Considering our extended non-linear model (4.3), the observer is a dynamical system that has as state the vector $\hat{\boldsymbol{x}}_e$, which is the estimation of state vector $\boldsymbol{x}_e$. The observer has two inputs: the control input $\boldsymbol{u}_F$ and the output vector $\boldsymbol{y}$. Then, the state equation of the linear observer has the following form:

$$\dot{\hat{\boldsymbol{x}}}_e = \boldsymbol{A}_o\hat{\boldsymbol{x}}_e + \boldsymbol{B}_o\boldsymbol{u}_F + \boldsymbol{L}\boldsymbol{y} \tag{4.21}$$

Let define the state estimation error $\boldsymbol{e}_{est} = \boldsymbol{x}_e - \hat{\boldsymbol{x}}_e$. Matrices $\boldsymbol{A}_o$ and $\boldsymbol{B}_o$ are chosen as $\boldsymbol{A}_o = \boldsymbol{A}_e(\boldsymbol{z}(t)) - \boldsymbol{L}\boldsymbol{C}$ and $\boldsymbol{B}_o = \boldsymbol{B}_e$ so that the dynamic equation of the estimation error is an autonomous system with equilibrium at origin

$$\begin{aligned}\dot{\boldsymbol{e}}_{est} = \dot{\boldsymbol{x}}_e - \dot{\hat{\boldsymbol{x}}}_e &= \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_e + \boldsymbol{B}_e\ \boldsymbol{u}_F - (\boldsymbol{A}_o\hat{\boldsymbol{x}}_e + \boldsymbol{B}_o\boldsymbol{u}_F + \boldsymbol{L}\boldsymbol{y}) \\ &= \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_e + \boldsymbol{B}_e\ \boldsymbol{u}_F - (\boldsymbol{A}_e(\boldsymbol{z}(t)) - \boldsymbol{L}\boldsymbol{C})\hat{\boldsymbol{x}}_e - \boldsymbol{B}_e\ \boldsymbol{u}_F - \boldsymbol{L}\boldsymbol{C}\boldsymbol{x}_e \\ &= (\boldsymbol{A}_e(\boldsymbol{z}(t)) - \boldsymbol{L}\boldsymbol{C})(\boldsymbol{x}_e - \hat{\boldsymbol{x}}_e) = (\boldsymbol{A}_e(\boldsymbol{z}(t)) - \boldsymbol{L}\boldsymbol{C})\boldsymbol{e}_{est}\end{aligned} \tag{4.22}$$

Matrix $\boldsymbol{L}$ is the gain of the observer and should be chosen such that $\boldsymbol{A}_e(\boldsymbol{z}(t)) - \boldsymbol{L}\boldsymbol{C}$ is stable, so the estimation error evolves to the equilibrium of (4.22) at origin. Since we have a polytopic model, matrix $\boldsymbol{A}_e(\boldsymbol{z}(t))$ is actually a linear combination of matrices $\boldsymbol{A}_{ei}$ as it is shown in (4.5). Therefore the procedure of design a state observer is similar to the design of a state-feedback controller. As in the case of the controller, the observer gain $\boldsymbol{L}$ is a linear combination of gains $\boldsymbol{L}_i$ obtained from the design of the observer for each subsystem. The state equation of the observer for the i-th subsystem is

$$\dot{\hat{\boldsymbol{x}}}_e = (\boldsymbol{A}_{ei} - \boldsymbol{L}_i\boldsymbol{C})\hat{\boldsymbol{x}}_e + \boldsymbol{B}_e\boldsymbol{u}_F + \boldsymbol{L}_i\boldsymbol{y} \tag{4.23}$$

We want to set the poles of $\boldsymbol{H}_i = \boldsymbol{A}_{ei} - \boldsymbol{L}_i\boldsymbol{C}$ in a LMI region just like it is done in the design of the controller for $\boldsymbol{G}_i = \boldsymbol{A}_{ei} + \boldsymbol{B}_e\boldsymbol{K}_i$. LMI constraints (4.11), (4.14) and (4.17) now becomes

$$\begin{cases} 2\alpha\boldsymbol{P} + \boldsymbol{H}_i^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{H}_i < 0 \quad \forall i = 1,\dots,32 \\ \begin{bmatrix} -\rho\boldsymbol{P} & \boldsymbol{P}\boldsymbol{H}_i \\ \boldsymbol{H}_i^T\boldsymbol{P} & -\rho\boldsymbol{P} \end{bmatrix} < 0 \quad \forall i = 1,\dots,32 \\ \begin{bmatrix} (\boldsymbol{P}\boldsymbol{H}_i + \boldsymbol{H}_i^T\boldsymbol{P})\sin\gamma & (\boldsymbol{P}\boldsymbol{H}_i - \boldsymbol{H}_i^T\boldsymbol{P})\cos\gamma \\ (\boldsymbol{H}_i^T\boldsymbol{P} - \boldsymbol{P}\boldsymbol{H}_i)\cos\gamma & (\boldsymbol{P}\boldsymbol{H}_i + \boldsymbol{H}_i^T\boldsymbol{P})\sin\gamma \end{bmatrix} < 0 \quad \forall i = 1,\dots,32 \end{cases} \tag{4.24}$$

where $\boldsymbol{H}_i = \boldsymbol{A}_{ei} - \boldsymbol{L}_i\boldsymbol{C}$ and $\boldsymbol{P} > 0$.

Again a change of variables should be done in order to solve the LMI problem, so it is defined the new variable $\boldsymbol{M}_i = -\boldsymbol{L}_i^T\boldsymbol{P}$. After applying the change in (4.24), the goal is to find matrices $\boldsymbol{P}$ and $\boldsymbol{M}_i$ such that constraints

$$\begin{cases} \boldsymbol{P} > 0 \\ 2\alpha\boldsymbol{P} + \boldsymbol{H}_{11i} < 0 \quad \forall i = 1, \dots, 32 \\ \begin{bmatrix} -\rho\boldsymbol{P} & \boldsymbol{P}\boldsymbol{A}_{ei} + \boldsymbol{M}^T\boldsymbol{C} \\ \boldsymbol{A}_{ei}^T\boldsymbol{P} + \boldsymbol{C}^T\boldsymbol{M}_i & -\rho\boldsymbol{P} \end{bmatrix} < 0 \quad \forall i = 1, \dots, 32 \\ \begin{bmatrix} \boldsymbol{H}_{11i}\sin\gamma & \boldsymbol{H}_{12i}\cos\gamma \\ \boldsymbol{H}_{21i}\cos\gamma & \boldsymbol{H}_{11i}\sin\gamma \end{bmatrix} < 0 \quad \forall i = 1, \dots, 32 \end{cases} \tag{4.25}$$

are fulfilled, where $\boldsymbol{H}_{11i} = \boldsymbol{P}\boldsymbol{A}_{ei} + \boldsymbol{A}_{ei}^T\boldsymbol{P} + \boldsymbol{C}^T\boldsymbol{M}_i + \boldsymbol{M}^T\boldsymbol{C}$, $\boldsymbol{H}_{12i} = \boldsymbol{P}\boldsymbol{A}_{ei} - \boldsymbol{A}_{ei}^T\boldsymbol{P} - \boldsymbol{C}^T\boldsymbol{M}_i + \boldsymbol{M}^T\boldsymbol{C}$ and $\boldsymbol{H}_{21i} = \boldsymbol{A}_{ei}^T\boldsymbol{P} - \boldsymbol{P}\boldsymbol{A}_{ei} + \boldsymbol{C}^T\boldsymbol{M}_i - \boldsymbol{M}^T\boldsymbol{C}$. Once the solutions $\boldsymbol{M}_i$ are obtained, the gains of the observer are computed as $\boldsymbol{L}_i = -(\boldsymbol{M}_i\boldsymbol{P}^{-1})^T$.

4.2 Reference tracking

If the state feedback control law (4.7) is applied to the extended TS-LIA model (4.3), and the design procedure is done as it is explained in Section 4.1.2, then the closed-loop system is an stable autonomous system with equilibrium at origin. If we want to set the equilibrium in a state different from the origin, one approach is to consider two additional inputs for the closed-loop system: an input reference $\boldsymbol{u}_{F\_ref}$ and a state reference $\boldsymbol{x}_{e\_ref}$. Figure 4.2 shows a general scheme of this approach.

The block named as 'F' is the Apkarian filter (4.2) and the AA model is (3.14) and represents the quadrotor system. Note that now the control law is (4.26) since the input to the controller is the state error $\boldsymbol{e}_{xe} = \boldsymbol{x}_{e\_ref} - \boldsymbol{x}_e$ and the input applied to the quadrotor system is not directly the output of the controller. The control input to the system now

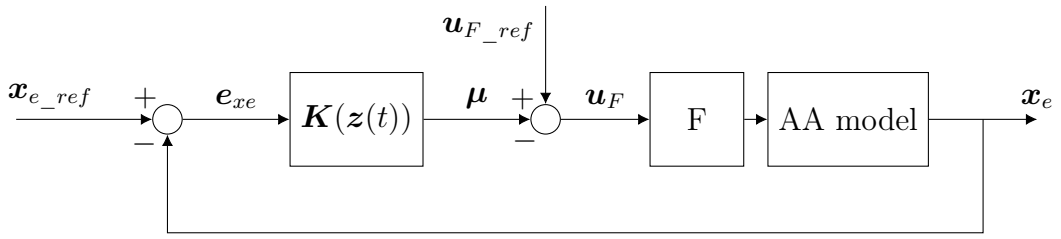


FIGURE 4.2: Control scheme with state and input reference

is $\boldsymbol{u}_F = \boldsymbol{u}_{F\_ref} - \boldsymbol{\mu}$. It is easy to prove that in case $\boldsymbol{x}_{e\_ref} = \boldsymbol{u}_{F\_ref} = \boldsymbol{0}$ the control law (4.26) becomes (4.7).

$$\boldsymbol{\mu} = \boldsymbol{K}(\boldsymbol{z}(t))\ \boldsymbol{e}_{xe}, \quad \boldsymbol{K}(\boldsymbol{z}(t)) = \sum_{i=1}^{32} h_i(\boldsymbol{z}(t))\ \boldsymbol{K}_i \tag{4.26}$$

The same controllers $\boldsymbol{K}_i$ found in Section 4.1.2 are valid in this scheme. This can be seen by computing the closed-loop system (4.27), where $\boldsymbol{u}_F$ is obtained from (4.26). If $\boldsymbol{A}_e(\boldsymbol{z}(t)) + \boldsymbol{B}_e\boldsymbol{K}(\boldsymbol{z}(t))$ is stable then (4.27) has some stable equilibrium point defined by $\boldsymbol{u}_{F\_ref}$ and $\boldsymbol{x}_{e\_ref}$.

$$\begin{aligned}
\dot{\boldsymbol{x}}_e &= \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_e + \boldsymbol{B}_e\ \boldsymbol{u}_F \\
&= \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_e + \boldsymbol{B}_e(\boldsymbol{u}_{F_ref} - \boldsymbol{K}(\boldsymbol{z}(t))(\boldsymbol{x}_{e_ref} - \boldsymbol{x}_e)) \\
&= (\boldsymbol{A}_e(\boldsymbol{z}(t)) + \boldsymbol{B}_e\boldsymbol{K}(\boldsymbol{z}(t)))\ \boldsymbol{x}_e + \boldsymbol{B}_e(\boldsymbol{u}_{F_ref} - \boldsymbol{K}\boldsymbol{x}_{e_ref})
\end{aligned} \tag{4.27}$$

There are different ways of computing $\boldsymbol{u}_{F\_ref}$ and $\boldsymbol{x}_{e\_ref}$. In this work, two approaches has been studied. In the first one, the input reference and the state reference are both computed from the desired output position and orientation, based on the knowledge of the steady state conditions. In the second approach, a reference model computes the input reference whereas the state reference should be given.

4.2.1 Feedforward control

In this approach, the control scheme is the one shown in Figure 4.3. The state and input references are computed from the desired position vector $\boldsymbol{y}^d = [Z^d\ \varphi^d\ \theta^d\ \psi^d]^T$ as $\boldsymbol{u}_{F\_ref} = \boldsymbol{N}_u\boldsymbol{y}^d$ and $\boldsymbol{x}_{e\_ref} = \boldsymbol{N}_x\boldsymbol{y}^d$.

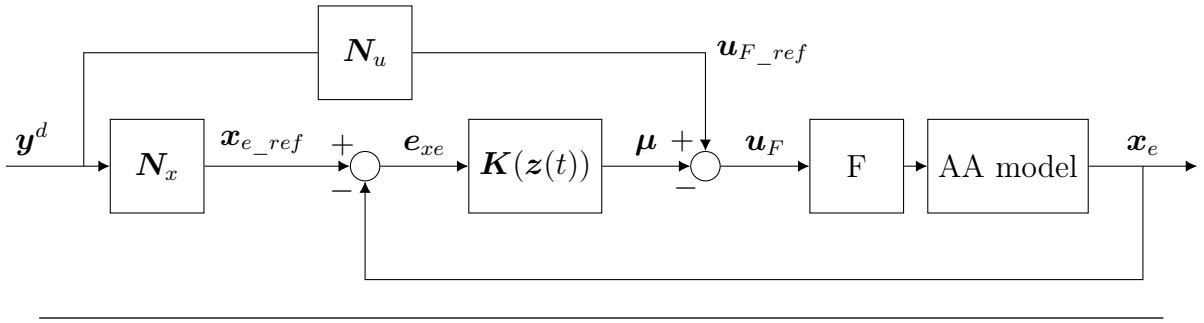


FIGURE 4.3: Feedforward control scheme

In steady state conditions, $\dot{\boldsymbol{x}}_e = \mathbf{0}$, $\boldsymbol{x}_e = \boldsymbol{x}_{e\_ref} = \boldsymbol{N}_x \boldsymbol{y}^d$ and $\boldsymbol{C}\boldsymbol{x}_e = \boldsymbol{y}^d$, where $\boldsymbol{C}$ is the same matrix defined in (4.20). Since $\boldsymbol{e}_{xe} = \mathbf{0}$, the output of the controller is $\boldsymbol{\mu} = \mathbf{0}$ so the input to the system is $\boldsymbol{u}_F = \boldsymbol{u}_{F\_ref} = \boldsymbol{N}_u \boldsymbol{y}^d$. If we apply these conditions to the extended TS-LIA model (4.3), the following equations are found:

$$\begin{cases} \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{N}_x \boldsymbol{y}^d + \boldsymbol{B}_e\ \boldsymbol{N}_u \boldsymbol{y}^d = \mathbf{0} \\ \boldsymbol{C}\boldsymbol{N}_x \boldsymbol{y}^d = \boldsymbol{y}^d \end{cases} \tag{4.28}$$

The solution of (4.28) for $\boldsymbol{N}_x$ and $\boldsymbol{N}_u$ is:

$$\begin{pmatrix} \boldsymbol{N}_x \\ \boldsymbol{N}_u \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}_e(\boldsymbol{z}(t)) & \boldsymbol{B}_e \\ \boldsymbol{C} & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{I} \end{pmatrix} \tag{4.29}$$

Note that $\boldsymbol{N}_x$ and $\boldsymbol{N}_u$ are not constant and should be computed on-line given the current matrix $\boldsymbol{A}_e(\boldsymbol{z}(t))$. Substituting $\boldsymbol{u}_{F\_ref} = \boldsymbol{N}_u \boldsymbol{y}^d$ and $\boldsymbol{x}_{e\_ref} = \boldsymbol{N}_x \boldsymbol{y}^d$ in (4.27) the closed-loop system is found:

$$\dot{\boldsymbol{x}}_e = (\boldsymbol{A}_e(\boldsymbol{z}(t)) + \boldsymbol{B}_e \boldsymbol{K}(\boldsymbol{z}(t)))\ \boldsymbol{x}_e + \boldsymbol{B}_e(\boldsymbol{N}_u - \boldsymbol{K}(\boldsymbol{z}(t))\boldsymbol{N}_x)\ \boldsymbol{y}^d \tag{4.30}$$

The main advantage of this scheme is that it is easy to implement and only the desired position $\boldsymbol{y}^d$ must be provided. As it will be shown in the simulations, this scheme is suitable for tracking of constant references, or references that does not change rapidly. However for most of the variable references an steady state error appears. Another drawback of

this scheme is that the computation of an inverse matrix is required in (4.29), which can yield into numerical problems.

4.2.2 Reference model based feedforward control

In this approach [9], the reference for the state $\boldsymbol{x}$ of the TS-LIA model (not the extended state $\boldsymbol{x}_e$) should be provided. The input reference $\boldsymbol{u}_{F\_ref}$ is obtained using a reference model and desired positions, velocities and accelerations. The reference of the extended state $\boldsymbol{x}_{e\_ref}$ is found by definition (4.4) as the concatenation of $\boldsymbol{x}_{ref}$ and $\boldsymbol{u}_{F\_ref}$. Figure 4.4 shows the control scheme of this approach.

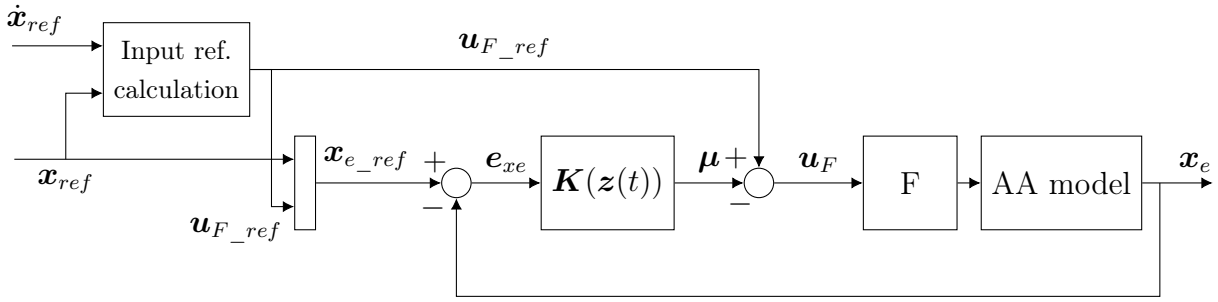


FIGURE 4.4: Reference model based feedforward control scheme

Let define the extended reference model (4.31). Note that matrices $\boldsymbol{A}_e(\boldsymbol{z}(t))$ and $\boldsymbol{B}_e$ are the same that appear in the extended TS-LIA model (4.3). As a consequence, these matrices depend on state vector $\boldsymbol{x}_e$ (instead of its own state vector $\boldsymbol{x}_{e\_ref}$). Therefore the premise variables, membership functions and activation functions are also the same.

$$\dot{\boldsymbol{x}}_{e\_ref} = \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_{e\_ref} + \boldsymbol{B}_e\ \boldsymbol{u}_{F\_ref} \tag{4.31}$$

The state vector of the extended reference model (4.32) is:

$$\boldsymbol{x}_{e\_ref} = \begin{bmatrix} Z^d & v_Z^d & \varphi^d & \theta^d & \psi^d & p^d & q^d & r^d & \Omega_{1\_ref} & \Omega_{2\_ref} & \Omega_{3\_ref} & \Omega_{4\_ref} \end{bmatrix}^T \tag{4.32}$$

Let recall the state error definition $\boldsymbol{e}_{xe} = \boldsymbol{x}_{e\_ref} - \boldsymbol{x}_e$ and the input error $\boldsymbol{\mu} = \boldsymbol{u}_{F\_ref} - \boldsymbol{u}_F$. By differentiating $\boldsymbol{e}_{xe}$ the error model is obtained (4.33). Since both extended reference

model and extended TS-LIA model share the same matrices $\boldsymbol{A}_e(\boldsymbol{z}(t))$ and $\boldsymbol{B}_e$, the error model has also those matrices. Therefore, all the procedure followed in the design of the controller for the extended TS-LIA model (4.3) is valid for the error model (4.33), and the controller obtained is the same. The control law (4.26) applied to (4.33) drives the state $\boldsymbol{x}_e$ toward the reference $\boldsymbol{x}_{e\_ref}$.

$$\begin{aligned}\dot{\boldsymbol{e}}_{xe} = \dot{\boldsymbol{x}}_{e\_ref} - \dot{\boldsymbol{x}}_e &= \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_{e\_ref} + \boldsymbol{B}_e\ \boldsymbol{u}_{F\_ref} - \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{x}_e - \boldsymbol{B}_e\ \boldsymbol{u}_F \\ &= \boldsymbol{A}_e(\boldsymbol{z}(t))(\boldsymbol{x}_{e\_ref} - \boldsymbol{x}_e) + \boldsymbol{B}_e(\boldsymbol{u}_{F\_ref} - \boldsymbol{u}_F) \\ &= \boldsymbol{A}_e(\boldsymbol{z}(t))\ \boldsymbol{e}_{xe} + \boldsymbol{B}_e\ \mu\end{aligned} \tag{4.33}$$

Let assume that the desired reference input $\boldsymbol{u}_{F\_ref}$ are the inputs of the TS-LIA model, i.e. they are the angular velocities Ω_{1_ref}, Ω_{2_ref}, Ω_{3_ref} and Ω_{4_ref}. From the reference model (4.31), the four differential equations (4.34) are considered. The other differential equations either do not depend on the input or are the ones referred to the filter, which is not needed for the computation of the reference inputs. The parameters of the model $\{a_{21}(t)\ b_{21}(t)\ \ldots\ b_{81}(t)\}$ are shown in (3.29).

$$\begin{cases} \dot{v_Z}^d = a_{21}(t)\ Z^d + b_{21}(t)\ (\Omega_{1\_ref} + \Omega_{2\_ref} + \Omega_{3\_ref} + \Omega_{4\_ref}) \\ \dot{p}^d = a_{67}(t)\ q^d + a_{68}(t)\ r^d + b_{61}(t)\ (\Omega_{1\_ref} + \Omega_{3\_ref}) + b_{62}(t)\ \Omega_{2\_ref} + b_{64}(t)\ \Omega_{4\_ref} \\ \dot{q}^d = a_{76}(t)\ p^d + a_{78}(t)\ r^d + b_{72}(t)\ (\Omega_{2\_ref} + \Omega_{4\_ref}) + b_{71}(t)\ \Omega_{1\_ref} + b_{73}(t)\ \Omega_{3\_ref} \\ \dot{r}^d = a_{86}(t)\ p^d + a_{87}(t)\ q^d + b_{81}(t)\ (\Omega_{1\_ref} - \Omega_{2\_ref} + \Omega_{3\_ref} - \Omega_{4\_ref}) \end{cases} \tag{4.34}$$

Remark 4.1. Taking advantage of the fact that the TS-LIA model is linear with respect the input, the system of equations (4.34) can be solved explicitly for Ω_{i_ref} for $i = \{1, 2, 3, 4\}$, so an online computation of an inverse matrix is not needed in this case. □

Note that not all the desired positions, velocities and accelerations are needed to compute the input references. In particular, the desired position Z^d, the desired angular velocities

p^d, q^d, r^d, and the desired accelerations $\dot{v}_Z^d$, $\dot{p}^d$, $\dot{q}^d$, $\dot{r}^d$ should be provided. The explicit computation of input vector $\boldsymbol{\Omega}_{ref} = \begin{bmatrix}\Omega_{1\_ref} & \Omega_{2\_ref} & \Omega_{3\_ref} & \Omega_{4\_ref}\end{bmatrix}^T$ is derived below.

Let consider the system of linear equations (4.34) and the vector of input references $\boldsymbol{\Omega}_{ref} = \begin{bmatrix}\Omega_{1\_ref} & \Omega_{2\_ref} & \Omega_{3\_ref} & \Omega_{4\_ref}\end{bmatrix}^T$. By writing the equations in matrix form $\boldsymbol{A}_r\boldsymbol{\Omega}_{ref} = \boldsymbol{B}_r$, the solution is $\boldsymbol{\Omega}_{ref} = \boldsymbol{A}_r^{-1}\boldsymbol{B}_r$

$$\begin{bmatrix}\Omega_{1\_ref}\\ \Omega_{2\_ref}\\ \Omega_{3\_ref}\\ \Omega_{4\_ref}\end{bmatrix} = \begin{bmatrix}b_{21}(t) & b_{21}(t) & b_{21}(t) & b_{21}(t)\\ b_{61}(t) & b_{62}(t) & b_{61}(t) & b_{64}(t)\\ b_{71}(t) & b_{72}(t) & b_{73}(t) & b_{72}(t)\\ b_{81}(t) & -b_{81}(t) & b_{81}(t) & -b_{81}(t)\end{bmatrix}^{-1} \begin{bmatrix}\dot{\boldsymbol{v}}_Z^d - a_{21}(t)Z^d\\ \dot{p}^d - (a_{67}(t)\ q^d + a_{68}(t)\ r^d)\\ \dot{q}^d - (a_{76}(t)\ p^d + a_{78}(t)\ r^d)\\ \dot{r}^d - (a_{86}(t)\ p^d + a_{87}(t)\ q^d)\end{bmatrix} \tag{4.35}$$

The explicit solution of $\boldsymbol{A}_r^{-1}$ is

$$\boldsymbol{A}_r^{-1} = \begin{bmatrix}-\frac{b_{72}(t)+b_{73}(t)}{2b_{21}(t)(b_{71}(t)-b_{73}(t))} & 0 & \frac{1}{b_{71}(t)-b_{73}(t)} & \frac{b_{72}(t)-b_{73}(t)}{2b_{81}(t)(b_{71}(t)-b_{73}(t))}\\ -\frac{b_{61}(t)+b_{64}(t)}{2b_{21}(t)(b_{62}(t)-b_{64}(t))} & \frac{1}{b_{62}(t)-b_{64}(t)} & 0 & -\frac{b_{61}(t)-b_{64}(t)}{2b_{81}(t)(b_{62}(t)-b_{64}(t))}\\ \frac{b_{71}(t)+b_{72}(t)}{2b_{21}(t)(b_{71}(t)-b_{73}(t))} & 0 & -\frac{1}{b_{71}(t)-b_{73}(t)} & \frac{b_{71}(t)-b_{72}(t)}{2b_{81}(t)(b_{71}(t)-b_{73}(t))}\\ \frac{b_{61}(t)+b_{62}(t)}{2b_{21}(t)(b_{62}(t)-b_{64}(t))} & -\frac{1}{b_{62}(t)-b_{64}(t)} & 0 & \frac{b_{61}(t)-b_{62}(t)}{2b_{81}(t)(b_{62}(t)-b_{64}(t))}\end{bmatrix} \tag{4.36}$$

It can be proven that $\boldsymbol{A}_r^{-1}$ always exists. Indeed, $b_{71}(t) \neq b_{73}(t)$ and $b_{62}(t) \neq b_{64}(t)$ $\forall t$. If we look at those parameters in (3.29), we see that $b_{71}(t)$ and $b_{73}(t)$ are two parallel straight lines, and the same happens with $b_{62}(t)$ and $b_{64}(t)$.

Chapter 5

Path following

In the previous chapter, it was shown how to control the altitude and orientation of a quadrotor system. In this chapter, the reference tracking scheme has been extended to a general control scheme which allows the tracking of trajectories in X, Y and Z. First, the Integral Backstepping (IB) control that provides the desired acceleration in X and Y is defined. Then, the computation from the desired accelerations to the roll and pitch references is done. Finally, the stability of the general control scheme is discussed.

5.1 General control scheme

The general control scheme for tracking 3D trajectories is shown in Figure 5.1 [5]. The *IB control* block is the Integral Backstepping controller that provides the accelerations in X and Y in order to control horizontal position. The *A.A. control* block includes the RM-FF control explained in Section 4.2.2. That scheme is used instead of the FF control because, as it will be seen in the simulations, it provides zero error in tracking of Z position and orientation control.

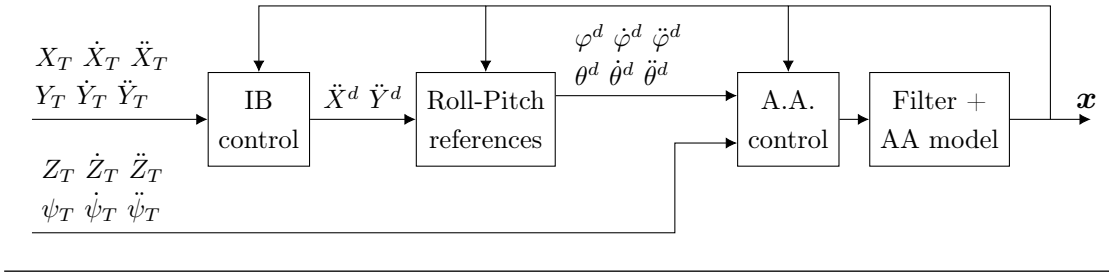


FIGURE 5.1: Control scheme for path following

5.2 Integral Backstepping control

The computation of the desired acceleration in X and Y directions is done using the integral backstepping control technique [10]. This control technique will be explained for X direction only, but the same can be applied for Y direction.

It is assumed that trajectory points X_T are perfectly known, and also the time derivatives $\dot{X}_T$ and $\ddot{X}_T$. Let define the tracking error $e_X = X_T - X$ where X_T is the desired x-coordinate position in the trajectory and X is the current x-coordinate position. The time derivative of this error is $\dot{e}_X = \dot{X}_T - \dot{X}$ where $\dot{X}_T$ is the desired velocity in the trajectory and $\dot{X}$ is the current velocity. First, the control law with proportional and integral terms (5.1) is defined, where K_d, K_p are positive constants and V_c is the virtual control velocity. Note that velocity V_c takes into account the desired velocity in the trajectory $\dot{X}_T$ and the velocity needed to minimize the tracking error e_X.

$$V_c = K_d\, e_X + K_p \int_0^t e_X(\tau)\, d\tau + \dot{X}_T \tag{5.1}$$

In order to find K_d and K_p, let consider the case that the current velocity $\dot{X}$ is the desired control velocity V_c. Then, by substituting (5.1) into the definition of $\dot{e}_X$, the following differential equation is found:

$$\dot{e}_X = -K_d\, e_X - K_p \int_0^t e_X(\tau)\, d\tau \tag{5.2}$$

By differentiating (5.2), we see that K_d and K_p determine the dynamics of the tracking position error

$$\ddot{e}_X + K_d\ \dot{e}_X + K_p\ e_X = 0 \tag{5.3}$$

The poles of (5.3) are chosen such that the system is critically damped. Therefore, both poles are $\lambda_1 = \lambda_2 = \lambda$ where λ is a real negative number, and the constants are obtained from

$$K_d = -2\lambda, \quad K_p = \lambda^2 \tag{5.4}$$

Let define the velocity tracking error as $e_V = V_c - \dot{X}$. Note that this is not the same error than the derivative of tracking error $\dot{e}_X$. The goal of the next step is to achieve $e_V = 0$. The desired dynamic for e_V is defined as

$$\dot{e}_V + K_v\ e_V + e_x = 0 \tag{5.5}$$

where K_v is constant and positive.

The derivative $\dot{e}_V$ can be obtained by substituting (5.1) into the definition of e_V and then computing the derivative:

$$\dot{e}_V = K_d\ \dot{e}_X + K_p\ e_X + \ddot{X}_T - \ddot{X} \tag{5.6}$$

Finally, the desired acceleration $\ddot{X}^d$ is obtained substituting (5.6) into (5.5).

$$\ddot{X}^d = K_d\ \dot{e}_X + (K_p + 1)\ e_X + K_v\ e_V + \ddot{X}_T \tag{5.7}$$

5.3 Computation of roll and pitch references

Let consider the dynamic equations of the quadrotor regarding X and Y (5.8), which are the first two equations of the quadrotor model shown in (3.9).

$$\begin{cases} \ddot{X} = (\sin\psi \sin\varphi + \cos\psi \sin\theta \cos\varphi) \dfrac{U_1}{m} \\ \ddot{Y} = (-\cos\psi \sin\varphi + \sin\psi \sin\theta \cos\varphi) \dfrac{U_1}{m} \end{cases} \tag{5.8}$$

The horizontal acceleration of the quadrotor depends on the thrust U_1 and angular positions φ, θ for a given yaw ψ. Therefore, the desired acceleration $\ddot{X}^d$ and $\ddot{Y}^d$ should be converted into a desired φ^d and θ^d assuming that ψ is measured. Also it is assumed that U_1 only compensates gravity (i.e. $\ddot{Z} = 0$), and therefore it is computed as:

$$U_1 = \frac{m\ g}{\cos\theta\ \cos\varphi} \tag{5.9}$$

Considering the desired accelerations $\ddot{X}^d$ and $\ddot{Y}^d$ and substituting (5.9) into (5.8) the following set of equations is found:

$$\begin{cases} \ddot{X}^d = g\left(\dfrac{\sin\psi}{\cos\theta}\tan\varphi + \cos\psi\tan\theta\right) \\ \ddot{Y}^d = g\left(-\dfrac{\cos\psi}{\cos\theta}\tan\varphi + \sin\psi\tan\theta\right) \end{cases} \tag{5.10}$$

The solution of (5.10) for φ and θ provides the required pitch and roll angles to have the desired horizontal acceleration:

$$\begin{cases} \theta^d = \arctan\left(\dfrac{\cos\psi\ \ddot{X}^d + \sin\psi\ \ddot{Y}^d}{g}\right) \\ \varphi^d = \arctan\left(\dfrac{\sin\psi\ \ddot{X}^d - \cos\psi\ \ddot{Y}^d}{g}\ \cos\theta^d\right) \end{cases} \tag{5.11}$$

In case of using the RM-FF control scheme for the Attitude/Attitude control (see Section 4.2.2), we will need not only the desired orientation φ^d and θ^d but also their time derivatives $\dot{\varphi}^d$, $\dot{\theta}^d$, $\ddot{\varphi}^d$ and $\ddot{\theta}^d$. In order to simplify the computation of these derivatives, the hypothesis of small Euler angles for θ and φ has been considered [11]. Therefore, the first order approximation of *arctan* function is done ($\arctan(x) \approx x$), and $\cos\theta^d \approx 1$. Finally, (5.11) becomes (5.12).

$$\begin{cases} \theta^d = \left(\cos\psi\ \ddot{X}^d + \sin\psi\ \ddot{Y}^d\right)/g \\ \varphi^d = \left(\sin\psi\ \ddot{X}^d - \cos\psi\ \ddot{Y}^d\right)/g \end{cases} \tag{5.12}$$

The desired angular velocities $\dot{\varphi}^d$ and $\dot{\theta}^d$ are

$$\begin{cases} \dot{\theta}^d = \left(\cos\psi\dot{\psi}\ddot{Y}^d - \sin\psi\dot{\psi}\ddot{X}^d + \dddot{X}^d\cos\psi + \dddot{Y}^d\sin\psi\right)/g \\ \dot{\varphi}^d = \left(\cos\psi\dot{\psi}\ddot{X}^d + \sin\psi\dot{\psi}\ddot{Y}^d + \dddot{X}^d\sin\psi - \dddot{Y}^d\cos\psi\right)/g \end{cases} \tag{5.13}$$

Note that we need to know how the acceleration $\ddot{X}^d$ given by the controller changes over time. This means that the third derivative $\dddot{X}^d$ must be computed by differentiating (5.7). Taking into account (5.6) and defining $\ddot{e}_X = \ddot{X}_T - \ddot{X}$, the result for $\dddot{X}^d$ (the result for $\dddot{Y}^d$ would be equivalent) is

$$\dddot{X}^d = (K_d + K_v)\ddot{e}_X + (K_p + K_vK_d + 1)\dot{e}_X + K_vK_p\ e_X + \dddot{X}_T \tag{5.14}$$

The desired angular accelerations $\ddot{\varphi}^d$ and $\ddot{\theta}^d$ are shown in (5.15).

$$\begin{cases} \ddot{\theta}^d = \Big(-(\dot{\psi}^2\cos\psi + \ddot{\psi}\sin\psi)\ddot{X}^d - 2\dot{\psi}\sin\psi\dddot{X}^d + \cos\psi\ddddot{X}^d + \\ \quad + (-\dot{\psi}^2\sin\psi + \ddot{\psi}\cos\psi)\ddot{Y}^d + 2\dot{\psi}\cos\psi\dddot{Y}^d + \sin\psi\ddddot{Y}^d\Big)/g \\ \ddot{\varphi}^d = \Big((-\dot{\psi}^2\sin\psi + \ddot{\psi}\cos\psi)\ddot{X}^d - 2\dot{\psi}\cos\psi\dddot{X}^d + \sin\psi\ddddot{X}^d + \\ \quad + (\dot{\psi}^2\cos\psi + \ddot{\psi}\sin\psi)\ddot{Y}^d + 2\dot{\psi}\sin\psi\dddot{Y}^d - \cos\psi\ddddot{Y}^d\Big)/g \end{cases} \tag{5.15}$$

As it can be seen, the fourth derivative $\ddddot{X}^d$ is needed, and it is obtained by differentiating (5.14) as

$$\ddddot{X}^d = (K_d + K_v)\ \dddot{e}_X + (K_p + K_vK_d + 1)\ \ddot{e}_X + K_vK_p\ \dot{e}_X + \ddddot{X}_T \tag{5.16}$$

where $\dddot{e}_X = \dddot{X}_T - \dddot{X}$.

Although in general the first four derivatives of X_T and Y_T are needed, this depends on the complexity of the trajectory. For smooth trajectories, third and fourth derivatives could be neglected. Also note that the second derivative of $\ddot{\psi}$ is needed in (5.15). This acceleration is not an state so it is not measured nor estimated. However it can approximated by the desired acceleration $\ddot{\psi}_T$.

5.4 Stability analysis

The AA controller does not guarantee that the quadrotor achieve the desired orientation instantaneously. As a consequence, the desired accelerations $\ddot{X}^d$ and $\ddot{Y}^d$ provided by the IB controller are also achieved after some time. Although the design of the state feedback controller and the IB controller is done so that they are stable, the whole control system shown in Figure 5.1 could be unstable. The goal of this section is to study how the dynamics of the AA controller affects stability of path following control. Additionally, the controller parameters K_p, K_d and K_v that ensure stability are determined.

Let consider the control scheme of Figure 5.2 which is the same than Figure 5.1 but regarding the movement in X coordinate only.

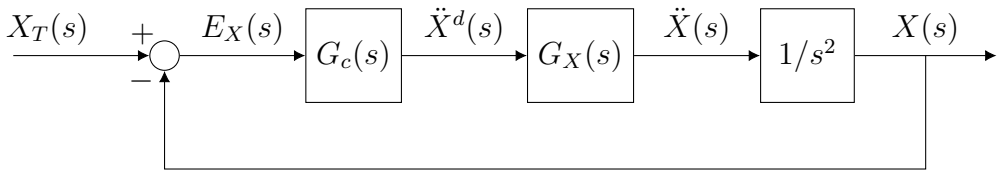


FIGURE 5.2: Path control scheme for X dynamics

Transfer function $G_c(s)$ represents the IB controller shown in (5.7). Transfer function $G_X(s)$ represents the combination of the roll-pitch reference computation block, the AA controller and the quadrotor equations for acceleration in X. As a result, $G_X(s)$ is a system with one input $\ddot{X}^d$ and one output $\ddot{X}$, and ideally an unitary static gain. Finally, the current acceleration is integrated such that the output is the position $X(s)$. The difference between the current position and the desired position in the trajectory is the tracking error $E_X(s) = X_T(s) - X(s)$.

Considering the definition of $e_V = V_C - \dot{X}$, the definition of $\dot{e}_X = \dot{X}_T - \dot{X}$ and (5.1), then (5.7) can be rewritten as

$$\ddot{X}^d = K_d\ \dot{e}_X + (K_p + 1)\ e_X + K_v\ \left(K_d\ e_X + K_p \int_0^t e_X(\tau)\ d\tau + \dot{e}_X \right) + \ddot{X}_T \tag{5.17}$$

The transfer function $G_c(s)$ is obtained by applying the Laplace transform into (5.17)

$$G_c(s) = \frac{\ddot{X}^d(s)}{E_X(s)} = \frac{(K_d + K_v)s^2 + (K_v K_d + K_p + 1)s + K_v K_p}{s} \tag{5.18}$$

The transfer function $G_X(s)$ has been approximated by a second order system as

$$G_X(s) = \frac{\ddot{X}(s)}{\ddot{X}^d(s)} = \frac{K_X}{\frac{1}{\omega_{nX}^2}s^2 + \frac{2\xi_X}{\omega_{nX}}s + 1} \tag{5.19}$$

The estimation of parameters K_X, ω_{nX} and ξ_X is done by applying a unitary step as input $\ddot{X}^d$ and looking at the response $\ddot{X}$. Figure 5.3 shows this response and the approximated second order system response.

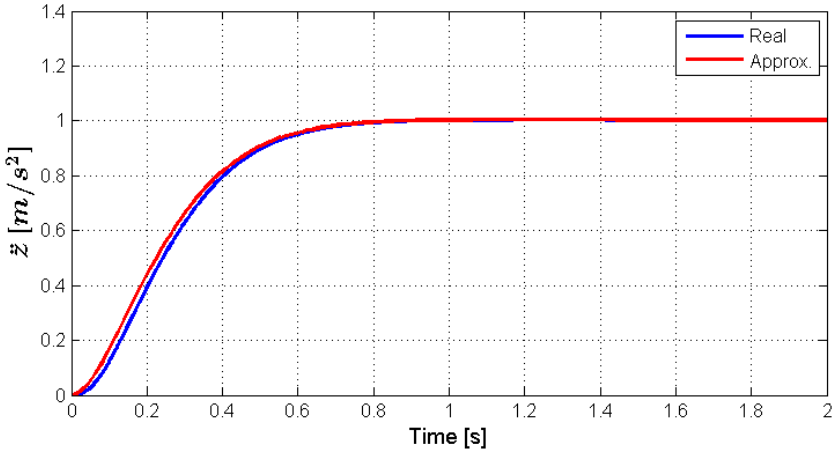


FIGURE 5.3: Step response of $G_X(s)$ and second order approximation

Remark 5.1. It is seen in simulation that in case $\ddot{Z}^d = 0$ the evolution of $\ddot{X}$ is the same for different $\ddot{Y}^d$. Although the states in the AA model are not decoupled, it is assumed that the evolution of $\ddot{X}$ is independent from $\ddot{Y}$ when $\ddot{Z}^d = 0$. □

The stability of the closed-loop system shown in Figure 5.2 has been studied using the Nyquist criterion applied to the open-loop system

$$G_{ol} = G_c(s)G_X(s)\frac{1}{s^2} = \frac{K\left(\frac{1}{\omega_{nc}^2}s^2 + \frac{2\xi_c}{\omega_{nc}}s + 1\right)}{\left(\frac{1}{\omega_{nX}^2}s^2 + \frac{2\xi_X}{\omega_{nX}}s + 1\right)s^3} \tag{5.20}$$

where

$$\omega_{nc} = \sqrt{\frac{K_v K_p}{K_d + K_v}}, \quad \xi_c = \frac{\omega_{nc}\,(K_v K_d + K_p + 1)}{2\;K_v K_p}, \quad K = K_X K_v K_p \tag{5.21}$$

By substituting $s = j\omega$ in (5.20), the gain $|G_{ol}(j\omega)| = M(j\omega)$ and phase $\angle G_{ol}(j\omega) = \Phi(j\omega)$ of the frequency response can be computed as

$$\begin{cases} M(j\omega) = \dfrac{|K|\;\left|\left(1 - \frac{\omega^2}{\omega_{nc}^2}\right) + j\;2\xi_c\frac{\omega}{\omega_{nc}}\right|}{\omega^3\;\left|\left(1 - \frac{\omega^2}{\omega_{nX}^2}\right) + j\;2\xi_X\frac{\omega}{\omega_{nX}}\right|} \\ \Phi(j\omega) = -\dfrac{3\pi}{2} - \arctan\left(\dfrac{2\xi_X\frac{\omega}{\omega_{nX}}}{1 - \frac{\omega^2}{\omega_{nX}^2}}\right) + \arctan\left(\dfrac{2\xi_c\frac{\omega}{\omega_{nc}}}{1 - \frac{\omega^2}{\omega_{nc}^2}}\right) \end{cases} \tag{5.22}$$

For any given positive frequency ω, a complex number with magnitude $M(\omega)$ and angle $\Phi(\omega)$ is obtained from (5.22). In case $\omega_{nX} > \omega_{nc}$ (that we can assume is true if we want the dynamics of the AA controller to be faster than the IB controller) then a qualitative representation of the Nyquist plot for $\omega > 0$ *rad* is shown in Figure 5.4. As it can be seen, the curve crosses the real axis at two different frequencies. In the special case where $\omega_{nX} >> \omega_{nc}$ (i.e. the AA control is instantaneous in comparison with the IB control), the curve only crosses once at frequency $\omega = \omega_{nc}$ and the phase tends to $-\pi/2$ rad when $\omega \to \infty$. If $\omega_{nX} < \omega_{nc}$ then the curve never crosses the real axis.

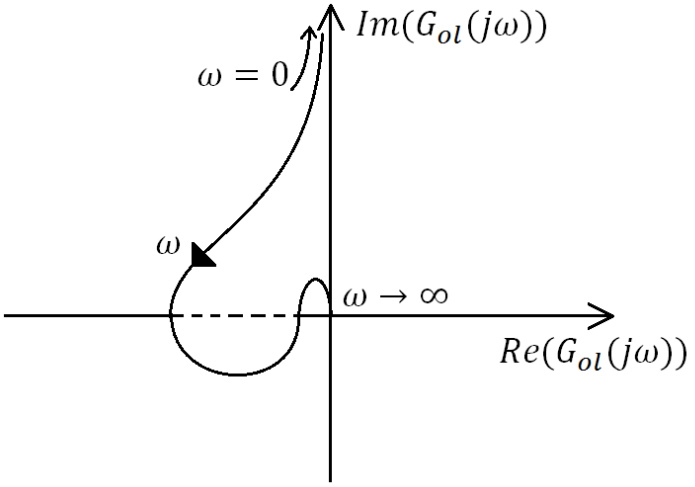


FIGURE 5.4: Qualitative Nyquist diagram of the open-loop system

Nyquist stability criterion: Let define the number of poles with positive real part of the closed-loop system as Z, the number of poles of the open-loop system as P, and the number of clockwise semi-encirclements (note only the half Nyquist plot is drawn) around -1 as N. In case the closed-loop system has poles at zero (three in our case) it must be added as many counter-clockwise quarters of circle as poles, starting at the initial phase (i.e. when $\omega = 0\ rad$). According to the Nyquist criterion the number of unstable poles of the closed-loop system is $Z = P + N$.

In our case $P = 0$ because $G_X(s)$ is stable (the closed-loop AA controller system is stable). Therefore, in order to have $Z = 0$, i.e. an stable closed-loop system, it must be $N = 0$. If the dashed segment shown in Figure 5.4 includes the complex point $-1 + j0$, then $N = 1 + (-1) = 0$ and the stability condition is fulfilled.

Note that all the parameters in (5.21) depend on two parameters: the real pole λ and IB controller's constant K_v. Constants K_d and K_p are computed from λ as shown in (5.4). Parameters K_X, ω_{nX} and ξ_X are known. Therefore the only tunable parameters are λ and K_v. Figure 5.5 shows four examples of bode diagrams for the following pairs of parameters: $\{K_v = 0.1,\ \lambda = -0.1\}$, $\{K_v = 10,\ \lambda = -0.1\}$, $\{K_v = 0.1,\ \lambda = -10\}$ and $\{K_v = 10,\ \lambda = -10\}$.

Note that there are two cases where the phase curve does not cross the -180 degrees. In the other two cases the closed loop system is stable. We want to know in detail which is

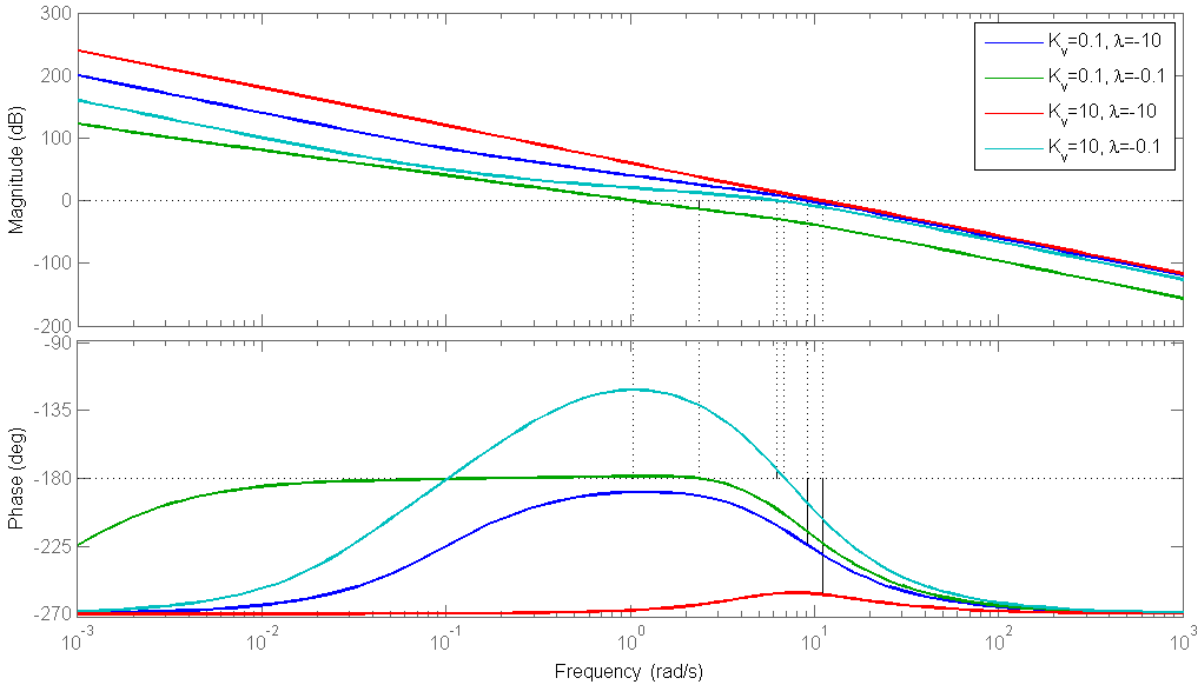


FIGURE 5.5: Bode diagram for four pairs of parameters K_v and λ

the region in the space of parameters $\{K_v, \lambda\}$ where the Nyquist stability condition holds. This is the same than check in which cases the Nyquist plot crosses the real axis twice in ω_{180_1} and in $\omega_{180\_2} > \omega_{180\_1}$, so that $M(\omega_{180\_1}) > 1$ and $M(\omega_{180\_2}) < 1$.

Let consider the intervals $\lambda \in [-10,\ 0]$ and $K_v \in [0,\ 20]$. By solving $\Phi(j\omega) = -\pi$ using the second equation of (5.22), ω_{180_1} and ω_{180_2} are found, and conditions $M(\omega_{180\_1}) > 1$ and $M(\omega_{180\_2}) < 1$ are checked using the first equation of (5.22). The parameters that satisfy the stability conditions are the one located in the shaded region of Figure 5.6. It can be seen as an example that the pairs of parameters that makes the closed-loop system stable in Figure 5.5 ($\{K_v = 0.1,\ \lambda = -0.1\}$ and $\{K_v = 10,\ \lambda = -0.1\}$) are inside the shaded region of Figure 5.6.

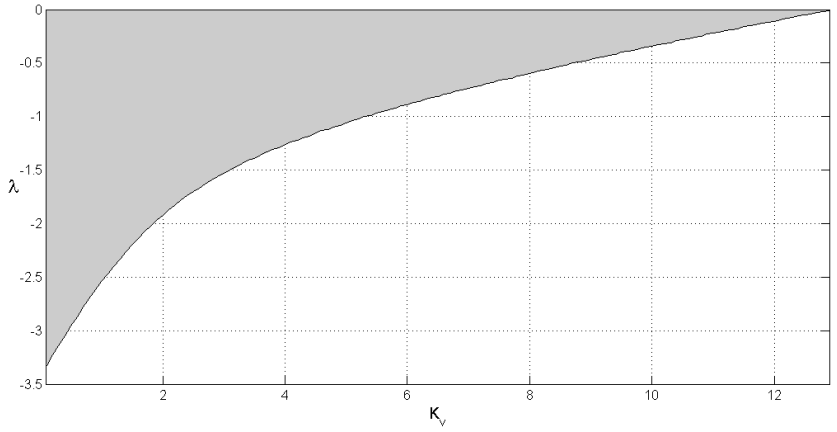


FIGURE 5.6: Region of parameters λ and K_v that satisfies stability condition

Chapter 6

Simulations and results

The different control techniques explained in this work has been implemented and simulated in Simulink®, and the results are discussed. First, three state-feedback controllers for altitude/attitude control have been designed and compared. Then, an state-observer has been designed. In the next section the FeedForward (FF) and Reference Model based FeddForward (RM-FF) control scheme have been implemented, simulated and compared. Finally, in the section dedicated to path tracking, an IB controller has been implemented in order to control X and Y position, and make the quadrotor to follow a 3D trajectory in simulation.

Before starting with the simulations, some general settings has been considered:

- The quadrotor system has been modeled and simulated using equations (3.9). The inputs are the angular speeds of the propellers. Since the design of the state-feedback controller is done including an Apkarian filter, this filter it is also added to the input of the quadrotor simulation model.
- The input to the quadrotor model (angular velocities $\mathbf{\Omega}$) has been saturated to make the model more realistic. The limits of the angular speeds are set to $\Omega_i \in [0,\ 600]\ rad\ s^{-1}$.

- Two Gaussian noise blocks has been added to the input $\boldsymbol{\Omega}$ and the states $\boldsymbol{x}$ to add realism to the model. The noise has zero mean and variances σ_x^2 for the states and $\sigma_{\boldsymbol{\Omega}}^2$ for the inputs. The matrices of covariances are diagonal.

$$\sigma_x^2 = \begin{bmatrix} 0.05^2 & 0.5^2 & 0.03^2 & 0.03^2 & 0.03^2 & 0.3^2 & 0.3^2 & 0.3^2 \end{bmatrix} \tag{6.1}$$

$$\sigma_{\boldsymbol{\Omega}}^2 = \begin{bmatrix} 0.01^2 & 0.01^2 & 0.01^2 & 0.01^2 \end{bmatrix} \tag{6.2}$$

- The values for general parameters and bounds of the TS model are summarized in Section A.2.

- Although the polytopic model used for the design of the state feedback controller does not include $Z = 0$, in most of the simulations the operation point for Z is zero or near to zero. This is done by simulating in other altitudes and then translating the result as it is explained in Remark 3.5.

- The sample time in all the simulations is 1 ms.

6.1 State feedback controller

Controllability should be checked before the design of the controller. Since the dynamics of the filter can be neglected, the (not extended) TS-LIA model has been considered in the controllability analysis.

Controllability analysis: It has been checked if all the states of the TS-LIA model are controllable by computing the controllability matrix for each subsystem. Given the subsystem matrices $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$, the controllability matrix $\mathcal{C}_i$ is computed as (6.3), which has maximum rank for each i-th subsystem.

$$\mathcal{C}_i = \begin{bmatrix} \boldsymbol{B_i} & \boldsymbol{A}_i\,\boldsymbol{B}_i & \boldsymbol{A}_i^2\,\boldsymbol{B}_i & \dots & \boldsymbol{A}_i^8\,\boldsymbol{B}_i \end{bmatrix} \tag{6.3}$$

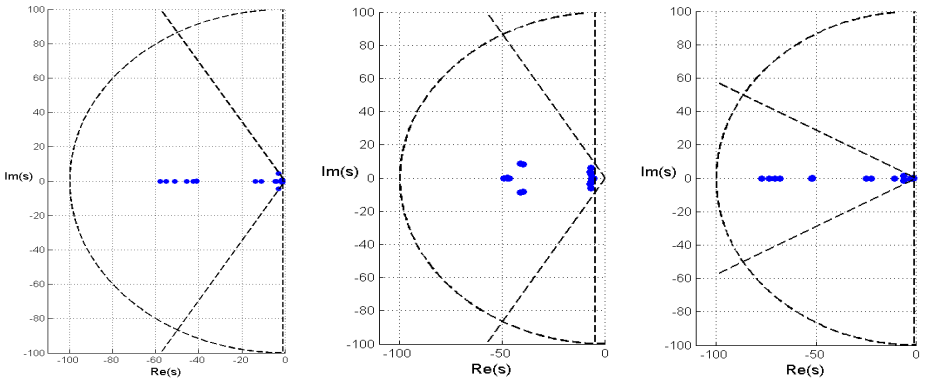
Several state feedback controllers has been designed following the procedure explained in Section 4.1.2 by specifying different LMI regions for the closed loop poles. The LMI

regions are defined by $\mathbb{S}_{\alpha,\rho,\gamma}$ (see Figure 4.1), where α is the minimum decay rate, ρ is the maximum natural frequency and γ is related with the minimum damping ratio. In this section, three examples of controllers with different performance requirements will be analysed. The parameters of the LMI regions are shown in Table 6.1.

LMI region	α	ρ	γ
R_1	1	100	$\pi/3$
R_2	5	100	$\pi/3$
R_3	1	100	$\pi/6$

TABLE 6.1: Three examples of LMI regions requirements for pole placement

The location of the closed loop poles at each LMI region is shown in Figure 6.1.



(a) R_1 (b) R_2 (c) R_3

FIGURE 6.1: Pole placement of closed loop system in different LMI regions

Figure 6.2 shows the evolution of position Z and angular positions φ, θ and ψ, starting at initial positions $Z(0) = 1\ m$, $\varphi(0) = \pi/4\ rad$, $\theta(0) = -\pi/4\ rad$ and $\psi(0) = \pi/2\ rad$. The initial velocities are zero. The initial states of the filter are the values at hovering condition Ω_H, so the net accelerations are zero. All three controllers stabilizes the quadrotor at the origin (see Remark 3.5). Note that the parameter that affects the performance more significantly is the value of α, which determines how fast the steady state is achieved. However, usually faster poles implies more "aggressive" control inputs. As a consequence the required angular speed of the propellers, which have physical limitations, are not able to follow the control signal which can be negative or very high. Therefore the saturation of the input $\mathbf{\Omega}$ includes a non-linearity that could yield into an undesired behaviour (like the peak shown in Figure 6.2 (b)) or even instability.

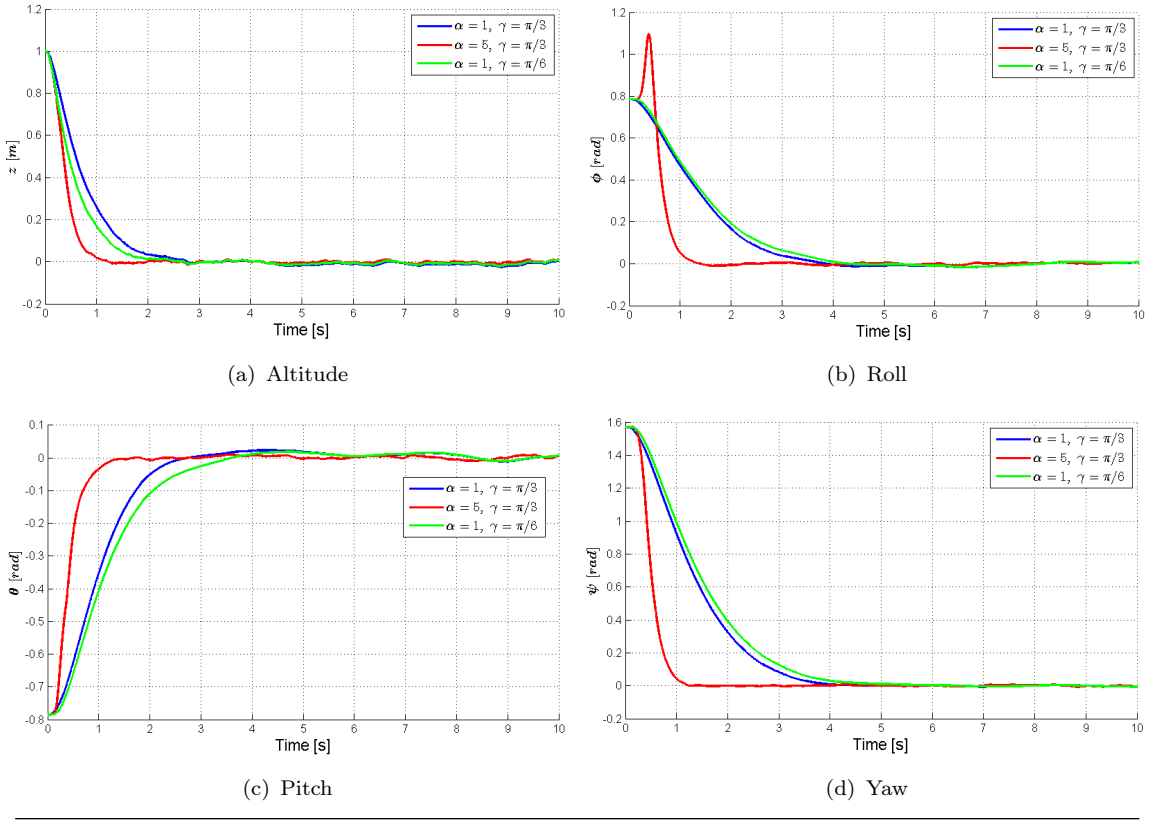


FIGURE 6.2: Comparison of different state feedback controllers

In Figure 6.3 the angular speed of the propellers are shown. Only the first two seconds of the simulation time has been plotted. As commented, in the case of having fast closed-loop poles ($\alpha = 5$) the input is saturated for long time. Note that the steady state value is $\Omega_H = 212.7\ rad$.

6.2 State observer

Observability analysis: It has been checked if all the states of the TS-LIA model are observable by computing the observability matrix for each subsystem. Given the subsystem matrices $\boldsymbol{A}_i$ and constant matrix $\boldsymbol{C}$ (6.4), the observability matrix $\mathcal{O}_i$ is computed as (6.5), which has maximum rank for each i-th subsystem.

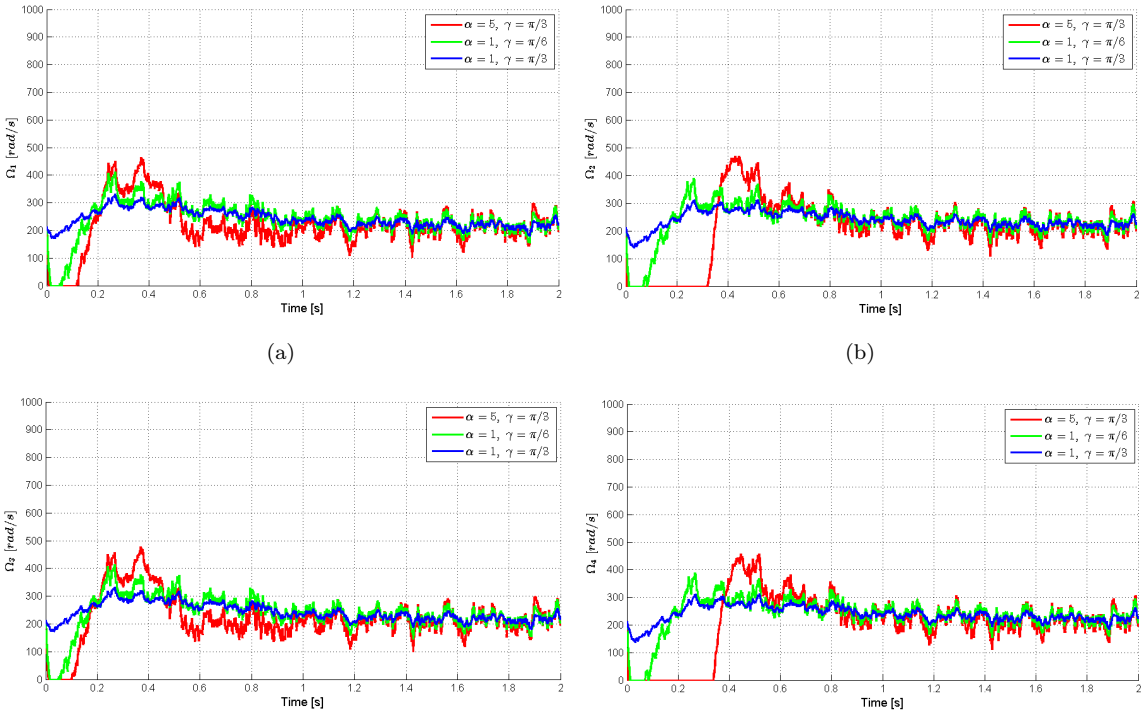


FIGURE 6.3: Angular speed of propellers Ω_1, Ω_2, Ω_3 and Ω_4

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{6.4}$$

$$\mathcal{O}_i = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{C}\ \boldsymbol{A}_i \\ \boldsymbol{C}\ \boldsymbol{A}_i^2 \\ \vdots \\ \boldsymbol{C}\ \boldsymbol{A}_i^8 \end{bmatrix} \tag{6.5}$$

The state observer has been designed following the procedure shown in Section 4.1.3. The poles of the observer should be faster than the closed-loop system poles. If the selected controller is the one that places the poles in the LMI region R_1 of Table 6.1, where $\alpha = 1$, then the poles of the observer should be about 10 times faster, i.e. $\alpha = 10$.

It is assumed that only the position Z and angles φ, θ, ψ are measured. Figure 6.4 shows the estimation error, which is the difference between the actual state and the state estimated by the observer. Note that only the first half second of simulation has been plotted. It is shown the states of the AA model but actually the complete extended state (which includes the states of the filter) is estimated.

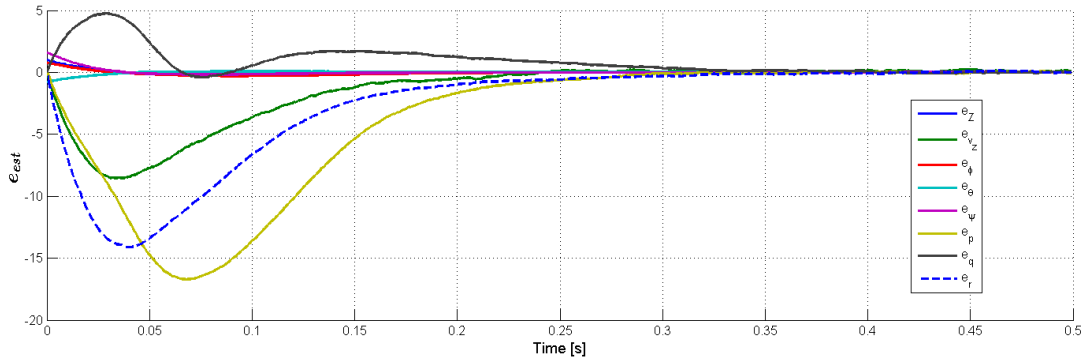


FIGURE 6.4: State estimation error

6.3 Reference tracking

Two different control schemes has been proposed for the case when the desired position Z^d and desired orientations φ^d, θ^d, ψ^d are not constant: the Feedforward (FF) control scheme (see Section 4.2.1) and the Reference Model based Feedforward (RM-FF) control scheme (see Section 4.2.2). Figure 6.5 shows the evolution of altitude and orientation of the quadrotor when each control scheme is applied. The variable references are the sinusoidal functions (6.6), where $A_Z = 1\ m$, $A_\varphi = A_\theta = A_\psi = 0.5\ rad$ are the amplitude of oscillation. The periods are $N_Z = N_\varphi = N_\theta = N_\psi = 10\ s$. Initial conditions are the same than in the previous simulations. The state feedback controller used in these simulations is again the one that set the poles of the closed-loop system in R_1.

$$
\begin{aligned}
Z^d &= A_Z \sin\left(\frac{2\pi t}{N_Z}\right), \quad \dot{Z}^d = A_Z \frac{2\pi}{N_Z} \cos\left(\frac{2\pi t}{N_Z}\right), \quad \ddot{Z}^d = -A_Z \left(\frac{2\pi}{N_Z}\right)^2 \sin\left(\frac{2\pi t}{N_Z}\right) \\
\varphi^d &= A_\varphi \sin\left(\frac{2\pi t}{N_\varphi}\right), \quad \dot{\varphi}^d = A_\varphi \frac{2\pi}{N_\varphi} \cos\left(\frac{2\pi t}{N_\varphi}\right), \quad \ddot{\varphi}^d = -A_\varphi \left(\frac{2\pi}{N_\varphi}\right)^2 \sin\left(\frac{2\pi t}{N_\varphi}\right) \\
\theta^d &= A_\theta \sin\left(\frac{2\pi t}{N_\theta}\right), \quad \dot{\theta}^d = A_\theta \frac{2\pi}{N_\theta} \cos\left(\frac{2\pi t}{N_\theta}\right), \quad \ddot{\theta}^d = -A_\theta \left(\frac{2\pi}{N_\theta}\right)^2 \sin\left(\frac{2\pi t}{N_\theta}\right) \\
\psi^d &= A_\psi \sin\left(\frac{2\pi t}{N_\psi}\right), \quad \dot{\psi}^d = A_\psi \frac{2\pi}{N_\psi} \cos\left(\frac{2\pi t}{N_\psi}\right), \quad \ddot{\psi}^d = -A_\psi \left(\frac{2\pi}{N_\psi}\right)^2 \sin\left(\frac{2\pi t}{N_\psi}\right)
\end{aligned}
\tag{6.6}
$$

The FF control scheme, only requires the first column of (6.6), i.e. the desired position and orientation. Since the FF scheme assumes the steady state behaviour of the system, it works properly when the references are constant values. But when the references are variable like in Figure 6.5 the output is delayed in phase and reduced in magnitude. Another drawback is that the FF control requires the computation of an inverse matrix in (4.29), which can result into numerical problems. Therefore, the FF control scheme also introduce, a limitation in the desired performance.

In the case of the RM-FF control scheme it is seen that there is no steady state error and it makes the position and orientation to follow the sinusoidal references. The computation of an inverse matrix is not needed in this case (see Remark 4.1). The main drawback of the RM-FF control scheme is that it needs the knowledge of the reference velocity and acceleration, not only the position, and that could not be provided nor computed. Also it could be difficult to provide the velocities and accelerations if the references are the output of another controller, like in the computation of roll and pitch references from the IB controller (see Section 5.3).

6.4 Path tracking

As it is explained in Chapter 5, the control in X and Y is done assuming that the desired positions X_T, Y_T, velocities $\dot{X}_T$, $\dot{Y}_T$ and accelerations $\ddot{X}_T$, $\ddot{Y}_T$ are known. The IB control provides the required acceleration in X and Y, and then the references for roll and pitch

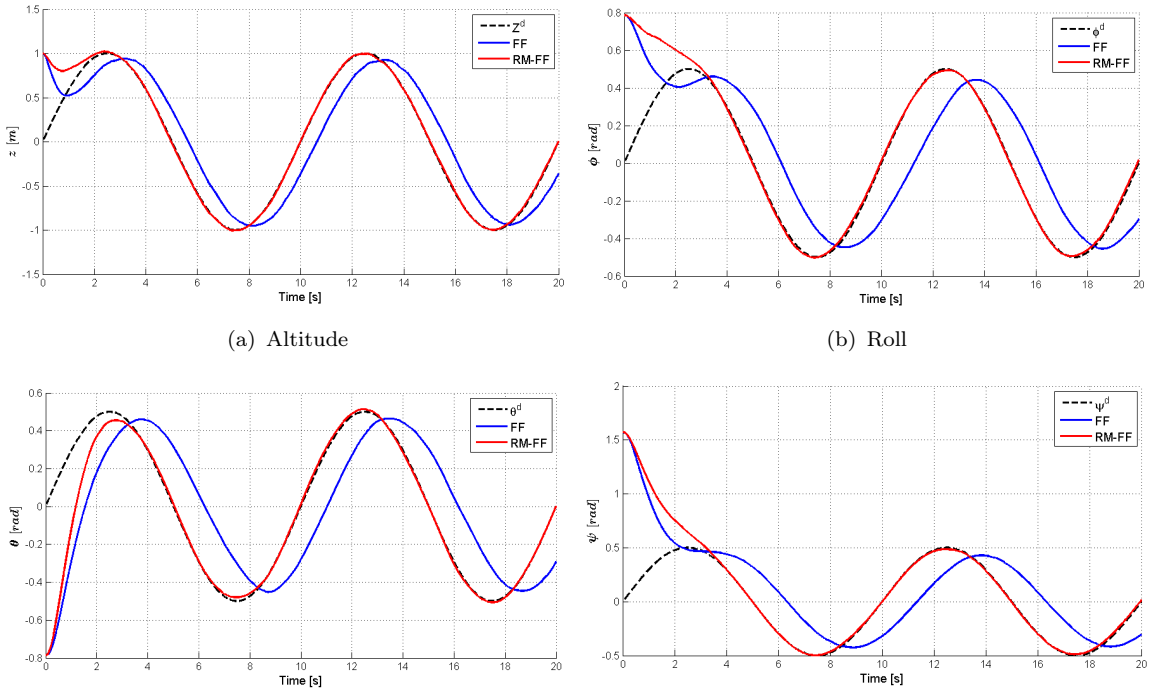


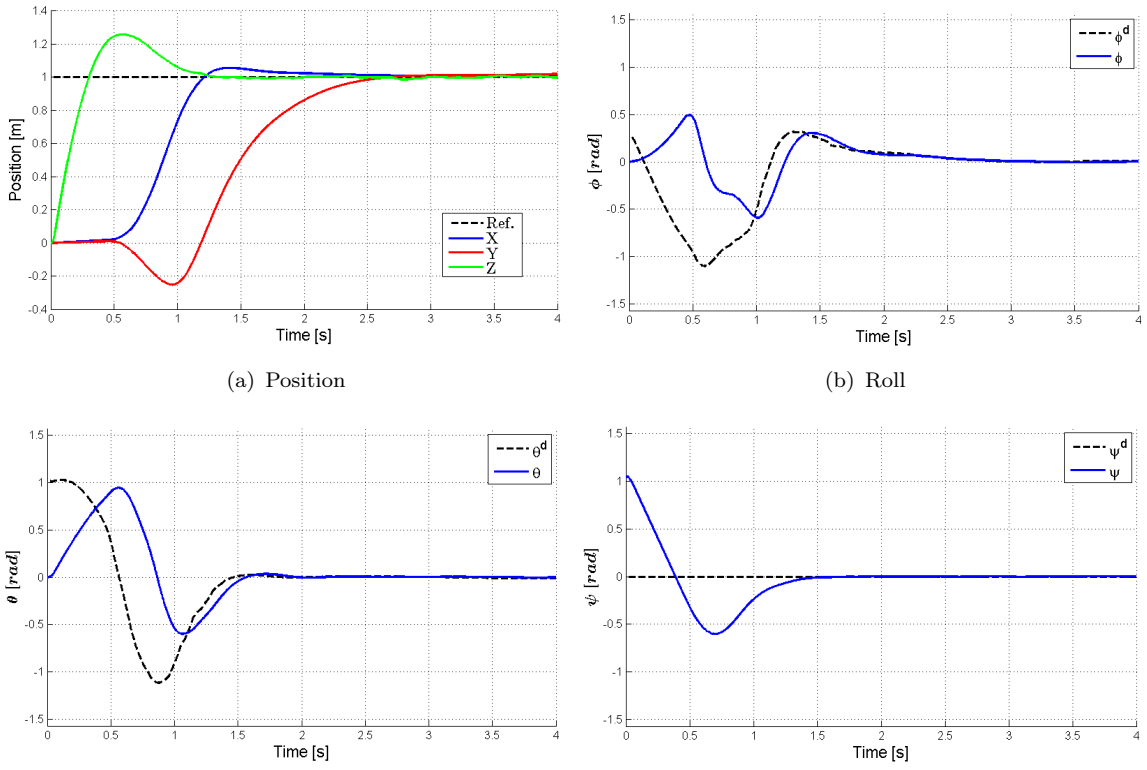
FIGURE 6.5: Variable reference tracking using Feedforward (FF) control scheme and Reference Model based Feedforward (RM-FF) control scheme

orientations are computed from these accelerations (see Figure 5.1). These roll and pitch references, along with the reference in altitude and yaw, provides the complete reference to the altitude and attitude control. In particular, the RM-FF control scheme is applied because it is able to track variable references.

The state feedback controller used in the AA control is the one that places the poles of the closed-loop system in R_2 (see Table 6.1). This controller will determine which is the estimated dynamics of $G_X(s)$ (see Figure 5.3), i.e. the parameters of its second order approximation (5.19). The estimated parameters of $G_X(s)$ are $K_X = 1.0035$, $\omega_{nX} = 7.1638\ rad\ s^{-1}$ and $\xi_X = 0.9287$. The state feedback controller also determines the parameters of the IB controller λ and K_v, i.e. the region of these parameters where the Nyquist stability condition is satisfied. This region for the controller used is the one shown in Figure 5.6. The parameters chosen are $\lambda = -2.5\ s^{-1}$ and $K_v = 0.5\ s^{-1}$. The other two parameters K_p and K_d are computed from (5.4).

Let consider first a trajectory composed by one point $[X_T\ Y_T\ Z_T]^T$. This is equivalent to say that references for X, Y and Z are constant. The coordinates of the point are

$X_T(t) = 1\ m$, $Y_T(t) = 1\ m$ and $Z_T(t) = 1\ m$ for all t. The desired velocity and acceleration at this point is zero. The desired yaw orientation is $\psi_T = 0\ rad$. The quadrotor is initially at origin position with initial yaw $\psi(0) = \pi/3\ rad$ and zero angular velocities and accelerations. Figure 6.6 shows the simulation results.



(a) Position (b) Roll

(c) Pitch (d) Yaw

FIGURE 6.6: 3D Position and yaw orientation control with constant references

In order to have a good performance regarding the settling time, a fast state-feedback controller is needed and appropriates constants for the IB controller should be chosen. However, fast controllers cause the drawback mentioned in the previous section: the saturation of the propeller speeds could produce undesired long-term saturated inputs, which could yield into an unstable behaviour of the quadrotor. Fortunately, the magnitude of the control input not only depends on the controller. It also depends on how big is the reference compared with the initial state. In the example of Figure 6.6, the gap between initial and desired position for each coordinate is 1 m. However, a new trajectory could be defined that connects initial and the same final point but considering more points in between, so the gaps between trajectory points are smaller. As a consequence, the

required control input will be smaller, and also the time the input is saturated is reduced, improving the overall performance.

Let define a new trajectory $T(t) = [X_T(t)\ Y_T(t)\ Z_T(t)]^T$ where the parametrized coordinates are (6.7).

$$T(t) = \begin{cases} X_T(t) = \dot{X}_T\ t \\ Y_T(t) = \dot{Y}_T\ t \\ Z_T(t) = \dot{Z}_T\ t \end{cases} \tag{6.7}$$

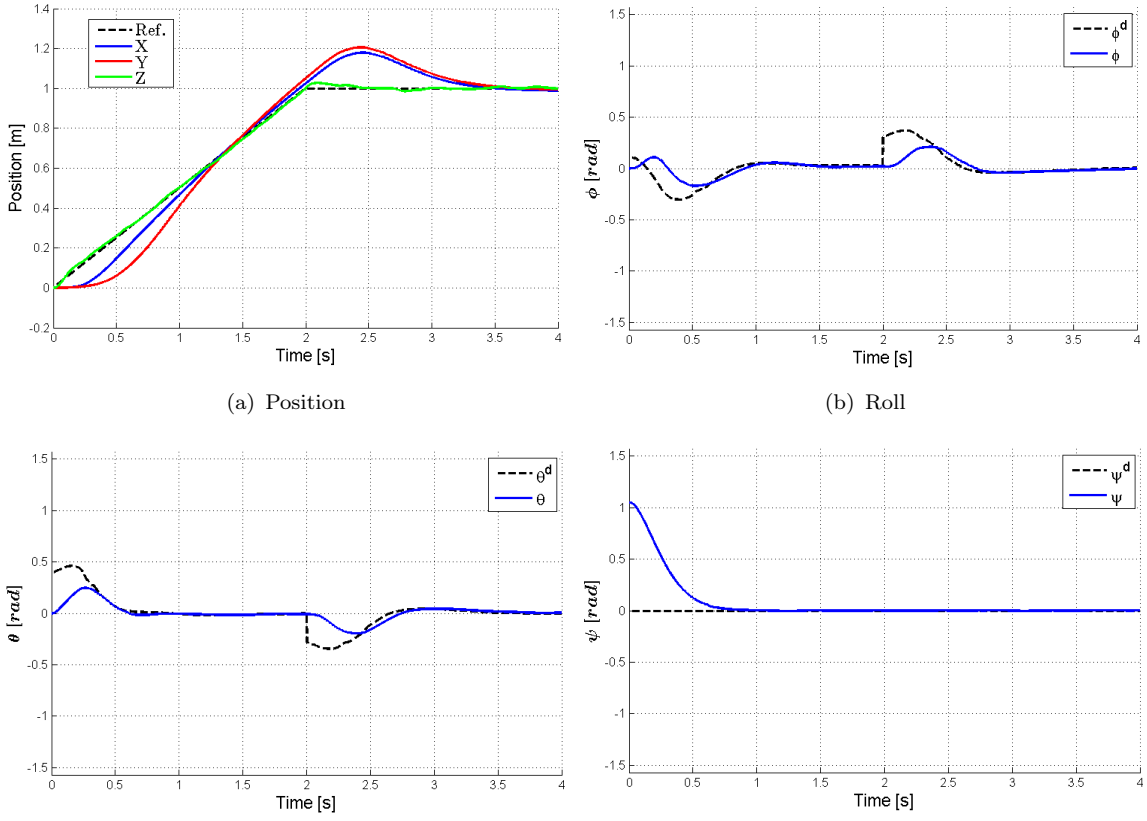
If the desired velocities $\dot{X}_T$, $\dot{Y}_T$ and $\dot{Z}_T$ are constant the trajectory (6.7) is a straight line. In order to connect the initial point (origin) with the desired final point $[1\ 1\ 1]^T$ the velocities must be equal. These velocities has been set to $\dot{X}_T = \dot{Y}_T = \dot{Z}_T = 0.5\ m\ s^{-1}$ so that the results are similar to the previous simulation, but any other value would be valid. The state feedback controller and the constants of the IB controller are the same than in the previous simulation, as well as the initial conditions.

The results of the simulation with this new trajectory are shown in Figure 6.7. Since the velocity in each coordinate is considered constant, the references for X, Y and Z are now ramps instead of steps. The slope of the ramps are the desired velocities $\dot{X}_T$, $\dot{Y}_T$ and $\dot{Z}_T$. Once these references reach the desired final position, then they are set to a constant value at that position. Note as the the coordinates X, Y and Z follow the references. Note also as the roll and pitch angles are small (i.e. near the hovering condition) when the quadrotor is following the ramp references (before the abrupt change of slope).

In the last simulation, an helicoidal trajectory is considered. It is defined by equations (6.8), where $\dot{Z}_T = 1\ m\ s^{-1}$ is constant.

$$T(t) = \begin{cases} X_T(t) = A_X \cos\left(\dfrac{2\pi t}{N_X}\right) \\ Y_T(t) = A_Y \sin\left(\dfrac{2\pi t}{N_Y}\right) \\ Z_T(t) = \dot{Z}_T\ t \end{cases} \tag{6.8}$$

The position, velocity and acceleration references for X and Y directions are sinusoidal signals as in (6.6). The yaw reference is set to $\psi_T = 0\ rad$ for all the simulation time, so the



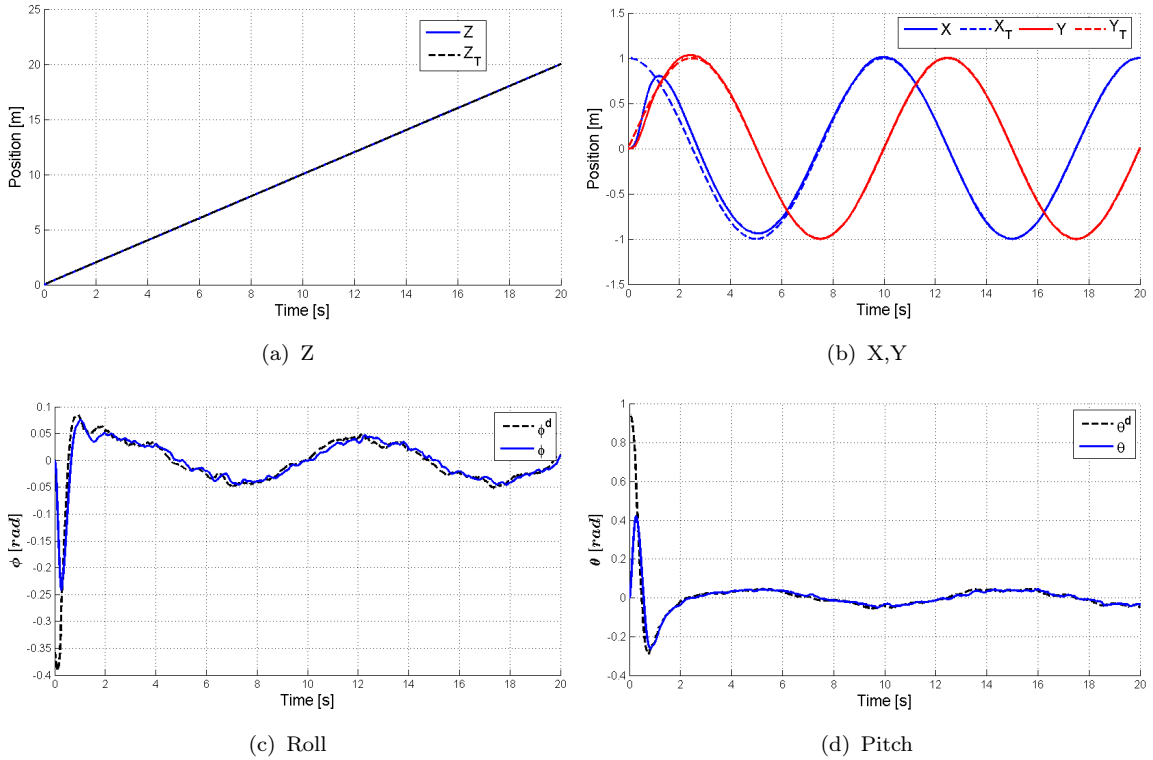
(a) Position (b) Roll

(c) Pitch (d) Yaw

FIGURE 6.7: 3D straight line reference tracking

quadrotor is performing a translation over the trajectory. The initial position is $[0\ 0\ 0]^T$, with zero velocities and accelerations. The amplitude of oscillations are $A_X = A_Y = 1\ m$, and the periods are $N_X = N_Y = 10\ s$. The result of the simulation is plotted in Figure 6.8.

Position Z evolves following a straight line whereas X and Y follow the sinusoidal references with zero steady state error. Note that the assumption of small Euler angles can be verified if the slope of the references changes smoothly (Figure 6.8 (c) and (d)), in contrast to the previous simulation (Figure 6.7 (b) and (c)). Figure 6.9 shows the evolution of 3D position in the same simulation.



(a) Z (b) X,Y

(c) Roll (d) Pitch

FIGURE 6.8: 3D helicoidal line reference tracking

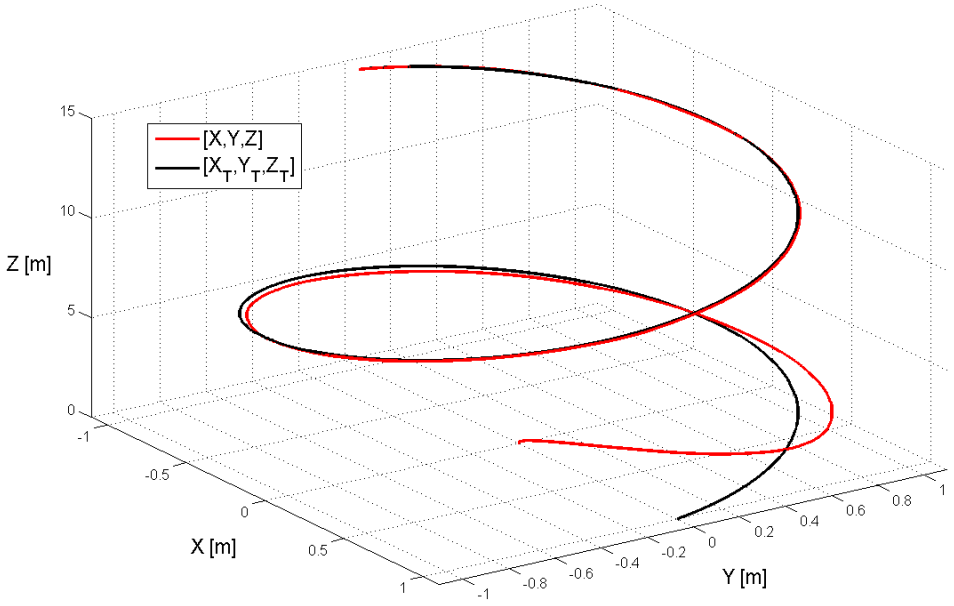


FIGURE 6.9: 3D position in helicoidal tracking

Chapter 7

Conclusions and future work

In this chapter, the conclusions of the thesis are discussed, as well as some additional work that could be done in future projects. Although some of them has been already commented as remarks in previous chapters, the main conclusions are the following:

- On the generation of a Takagi-Sugeno model, it has been seen that there is a trade-off between how accurate the model is and how big the dimension of the model is (number of rules/subsystems). The Takagi-Sugeno model obtained from sector non-linearity approach has been proven to be the most accurate (see Figure 3.7) but also the one with more rules/subsystems. In the best case (when the triangular polytopic is considered for some premise variables), there are 1296 rules/subsystems. However the TS-LIA model provides a handy Takagi-Sugeno model of 32 rules/subsystems, despite the lost of accuracy in the linear approximation of input torques and forces.

- One of the main advantages of the TS-LIA model developed in this work is, as it is said above, the reduction on the dimension of the model. Other advantage is the fact that the premise variables are independent form the input which makes the defuzzification process of the controller easier. It has also the advantage of being linear with respect the input, so the computation of input references in the Reference Model based FeedForward control scheme does not require the computation of an inverse matrix, so it can be computed explicitly instead (Remark 4.1).

- The Reference Model based FeedForward control scheme is suitable for tracking variable references in the altitude and orientation control with zero steady state error. However, it has limitations related with references of angles that are far from zero, where the assumption of small Euler angles is not fulfilled. Another drawback is that not only positions should be provided as references. The knowledge of desired velocities and accelerations are also necessary.

- There is a physical limitation on the required performance (for example on how fast the closed-loop system might be) when the state feedback controller is designed. This limitation is related with the fact that the speed of the propellers could be saturated. If the input applied is not the control input for too long due to the saturation, this could produce an unstable system behaviour.

- The tracking of trajectories in 3D space generates better results if the trajectory points are close one another, so the required control input does not force the saturation. In addition, the references for roll and pitch angles do not violate the assumption of small Euler angles.

The following are some proposed ideas for future works:

- The quadrotor models shown in this work have as input the angular speed of the propellers. As a consequence, it is assumed that when an input control signal is applied, the desired angular speed is achieved instantaneously. In a more realistic model the input would be the voltage to the actuators, and therefore the dynamics that relates this voltage with the angular speed would be considered. In that case the input would not be the noisy signal shown in Figure 6.3 because a limitation on the rate of change of angular speeds would be introduced.

- The approximation of the input done in the TS-LIA model is valid under the assumption of small changes of propellers speed with respect hovering condition. However, an important property of the physical system (the quadratic relation between angular speed and forces/torques) is neglected. It would be interesting to explore which are the limitations, if any, of this assumption.

- As commented in the conclusions, the saturation in the propellers speed could cause instability. However, the system could remain stable for short time saturated input signals as it is shown in Figure 6.2 and Figure 6.3. It would be interesting to study the stability of the system taking into account the non-linearity introduced by the saturation.

- The stability analysis of the path tracking control is done by approximating the inner Altitude/Attitude control to a linear second order system. This provides a general idea on how the parameters of the IB controller should be set in order to make the system stable. A more general result could be obtained by taking into account the complete AA control instead of an approximation.

- Implementation of the ideas developed in this work in a real environment.

Appendix A

Quadrotor models

A.1 Newton-Euler model

The Newton-Euler model derived in [7] has the following form:

$$\boldsymbol{M}\ \dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\boldsymbol{\zeta})\ \boldsymbol{\zeta} = \boldsymbol{G} + \boldsymbol{O}(\boldsymbol{\zeta})\ \boldsymbol{\Omega} + \boldsymbol{E}(\boldsymbol{\xi})\ \boldsymbol{\Omega}^2 \tag{A.1}$$

Each one of the matrices are explained below:

- **Inertia matrix**: Constant matrix that contains the inertial mass (m) and the inertia moments in the principal directions (I_X, I_Y and I_Z).

$$\boldsymbol{M} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_X & 0 & 0 \\ 0 & 0 & 0 & 0 & I_Y & 0 \\ 0 & 0 & 0 & 0 & 0 & I_Z \end{bmatrix} \tag{A.2}$$

- **Coriolis-centripetal matrix**: Matrix that considers the Coriolis and centripetal accelerations. It depends on the inertia moments and the angular velocity $\boldsymbol{\omega}$.

$$\boldsymbol{C}(\boldsymbol{\zeta}) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_Z\ r & -I_Y\ q \\ 0 & 0 & 0 & -I_Z\ r & 0 & I_X\ p \\ 0 & 0 & 0 & I_Y\ q & -I_X\ p & 0 \end{bmatrix} \tag{A.3}$$

- **Gravitational vector**: External force/torque due to gravity. It is constant.

$$\boldsymbol{G} = \begin{bmatrix} 0 \\ 0 \\ -m\ g \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{A.4}$$

- **Gyroscopic propeller matrix**: Matrix that represents the gyroscopic effect. It depends on the angular velocity $\boldsymbol{\omega}$ and the rotational moment of inertia J_{TP} around the propeller axis. As it is shown in (A.1), this matrix is multiplied by the propellers speed vector $\boldsymbol{\Omega}$, shown in (A.6), which contains the angular speed Ω_i of each propeller $i \in \{1, 2, 3, 4\}$.

$$\boldsymbol{O}(\boldsymbol{\zeta}) = J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{A.5}$$

$$\boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 & \Omega_2 & \Omega_3 & \Omega_4 \end{bmatrix}^T \tag{A.6}$$

- **Movement matrix**: In the B-frame there are four basic forces/torques which are related with the basic movements explained in the previous section: a lift force in

Z_B named U_1, a torque which produces the roll rotation around axis X_B named U_2, a torque which produces the pitch rotation around axis Y_B named U_3, and the counter-torque which produces the yaw rotation around axis Z_B named U_4:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -b\,l & 0 & b\,l \\ -b\,l & 0 & b\,l & 0 \\ -d & d & -d & d \end{bmatrix} \boldsymbol{\Omega}^2 \tag{A.7}$$

Here b is the thrust factor, d is the drag factor, l is the distance between the center of the quadrotor and the center of any propeller and $\boldsymbol{\Omega}^2 = [\Omega_1^2\ \Omega_2^2\ \Omega_3^2\ \Omega_4^2]^T$.

The last term in (A.1) is the product of the movement matrix $\boldsymbol{E}(\boldsymbol{\xi})$ and the vector $\boldsymbol{\Omega}^2$, and can be written in terms of the forces/torques (U_1, U_2, U_3, U_4) as:

$$\boldsymbol{E}(\boldsymbol{\xi})\ \boldsymbol{\Omega}^2 = \begin{bmatrix} (s_\psi s_\varphi + c_\psi s_\theta c_\varphi)\ U_1 \\ (-c_\psi s_\varphi + s_\psi s_\theta c_\varphi)\ U_1 \\ (c_\theta c_\varphi)\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{A.8}$$

A.2 System and models parameters

This section summarizes all the values of parameters and constants needed to run the simulations of the AA models and TS models.

A.2.1 Quadrotor system parameters

The parameters used in this thesis (Table A.1) are based on the estimations done by [7] and used later in [9]. Note that ideally the quadrotor is symmetric with respect x-axis and y-axis, so the moments of inertia I_X and I_Y are equal. As a consequence, the equation

of $\dot{r}$ in the AA model could have been simplified, but it is not in order to maintain the generality of the model.

Parameter	Description	Value
m	Mass of the quadrotor	$1\ kg$
I_X	Body moment of inertia around x-axis	$8.1 \times 10^{-3}\ Nms^2$
I_Y	Body moment of inertia around y-axis	$8.1 \times 10^{-3}\ Nms^2$
I_Z	Body moment of inertia around z-axis	$14.2 \times 10^{-3}\ Nms^2$
d	Drag factor	$1.1 \times 10^{-6}\ Nms^2$
b	Thrust factor	$54.2 \times 10^{-6}\ Ns^2$
l	Distance from the center of the quadrotor to a propeller	$0.24\ m$
J_{TP}	Total rotational moment of inertia around the propeller axis	$104 \times 10^{-6}\ Nms^2$

TABLE A.1: Quadrotor system parameters

From (3.16) the input needed in hovering condition can be computed, and its value is $\Omega_H = 212.7\ rad\ s^{-1}$.

A.2.2 Bounds of input, state and premise variables

The generation of a Takagi-Sugeno model requires the definition of bounds for the premise variables. The premise variables not depend on all the states, so only the bounds for the states and inputs that appears in the premise variables will be defined as follows

$$\begin{aligned} Z &\in [1,\ 20]\ m & \varphi &\in [-\pi/4,\ \pi/4]\ rad \\ \theta &\in [-\pi/4,\ \pi/4]\ rad & p &\in [-0.25,\ 0.25]\ rad\ s^{-1} \\ q &\in [-0.25,\ 0.25]\ rad\ s^{-1} & r &\in [-0.25,\ 0.25]\ rad\ s^{-1} \\ \Omega_1 &\in [100,\ 500]\ rad\ s^{-1} & \Omega_2 &\in [100,\ 500]\ rad\ s^{-1} \\ \Omega_3 &\in [100,\ 500]\ rad\ s^{-1} & \Omega_4 &\in [100,\ 500]\ rad\ s^{-1} \end{aligned} \tag{A.9}$$

The selection of bounds for Z has been chosen arbitrarily and the only aim is to avoid the zero in the interval (see Remark 3.5). The other bounds has been selected as in [9].

The bounds of the premise variables for the TS model are:

$$\begin{array}{llll} \underline{z}_1 = 1/20, & \overline{z}_1 = 1, & \underline{z}_2 = -0.25, & \overline{z}_2 = 0.25, \\ \underline{z}_3 = -0.25, & \overline{z}_3 = 0.25, & \underline{z}_4 = -0.25, & \overline{z}_4 = 0.25 \\ \underline{z}_5 = 50, & \overline{z}_5 = 500, & \underline{z}_6 = 50, & \overline{z}_6 = 500 \\ \underline{z}_7 = 50, & \overline{z}_7 = 500, & \underline{z}_8 = 50, & \overline{z}_8 = 500 \\ \underline{z}_9 = 100, & \overline{z}_9 = 500, & \underline{z}_{10} = 100, & \overline{z}_{10} = 500 \\ \underline{z}_{11} = 100, & \overline{z}_{11} = 500, & \underline{z}_{12} = 100, & \overline{z}_{12} = 500 \end{array} \tag{A.10}$$

And the bounds of the premise variables for the TS-LIA model are:

$$\begin{array}{llll} \underline{z}_1 = 1.5/20, & \overline{z}_1 = 2, & \underline{z}_2 = -0.25, & \overline{z}_2 = 0.25 \\ \underline{z}_3 = -0.25, & \overline{z}_3 = 0.25, & \underline{z}_4 = -0.25, & \overline{z}_4 = 0.25, \\ \underline{z}_5 = 0.5, & \overline{z}_5 = 1 & & \end{array} \tag{A.11}$$

A.3 Reduction on the number of rules in a TS model

One of the main problems on generating a Takagi-Sugeno model is the number of rules and subsystems that could arise. Assuming that the subsystems are found by combination of bounds of the premise variables and that all the combinations are considered, then 2^k rules are needed (where k is the number of premise variables). However, is not always mandatory to consider all the combinations. In many cases some rules/subsystems can be neglected so that the dimensionality of the TS model is reduced at expense of loosing conservativeness.

The general idea is try to reduce the polytopic space of premise variables defined by the combinations of bounds without excluding any realizable subsystem. For the examples below, it will be assumed that there exist at least two premise variables z_1 and z_2 that could be independent or not, and maybe other $p-2$ premise variables which are independent from the first two. The bounds for the premise variables are: $z_1 \in [\underline{z}_1\ \overline{z}_1]$ and $z_2 \in [\underline{z}_2\ \overline{z}_2]$.

- **Case 1: Rectangular polytopes**

When two variables are independent (see Figure A.1(a)), or the dependency is such that the realizable values cannot be constraint by three vertices (see Figure A.1(b)(c)), then the rules/subsystems related with all the combinations of upper and lower bounds must be considered. In this case we have two membership functions for each premise variable (see Section A.4).

All the possible combinations of fuzzy subsets ("big" and "small") appears in the fuzzy rules:

Rule 1: z_1 is "small" and z_2 is "small" and ...

Rule 2: z_1 is "small" and z_2 is "big" and ...

Rule 3: z_1 is "big" and z_2 is "small" and ...

Rule 4: z_1 is "big" and z_2 is "big" and ...

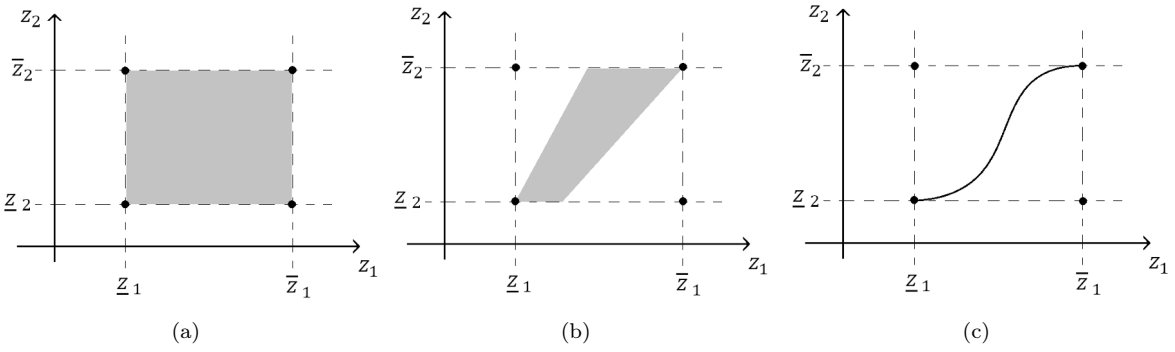


FIGURE A.1: Example of premise variables such that all four vertices should be considered

- **Case 2: Triangular polytope**

When the set of realizable values for z_1 and z_2 can be constrained by a triangle with three combinations of upper and lower bounds, then the fourth combination can be omitted. In Figure A.2, two examples of this are shown. In this case, there are three membership functions for each pair of dependent premise variables (see Section A.4).

The number of fuzzy rules is 3^k, where k is the number of pairs of premise variables. For the examples in Figure A.2, the combinations of the fuzzy subsets in the rules would be:

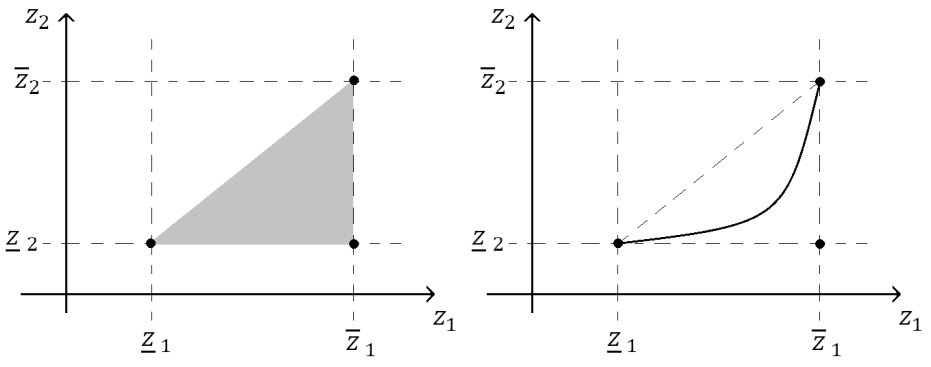


FIGURE A.2: Example of premise variables such that only three vertices are needed

Rule 1: z_1 is "small" and z_2 is "small" and ...

Rule 2: z_1 is "big" and z_2 is "small" and ...

Rule 3: z_1 is "big" and z_2 is "big" and ...

- **Case 3: Straight line**

Let consider the case when a premise variable z_{n+1} is a linear combination of n premise variables $[z_1\ z_2\ \dots z_n]$. In other words, z_{n+1} can be written as:

$$z_{n+1} = a_0 + a_1\ z_1 + a_2\ z_2 + \cdots + a_n\ z_n \tag{A.12}$$

where $a_0, a_1, \dots, a_n$ are constants. In this case, all the realizable values can be constrained by a straight line between two vertices (i.e. two of the combinations of upper and lower bounds). Figure A.3 shows the particular example of two premise variables such that $z_2 = a_0 + a_1\ z_1$, where $a_1 > 0$.

In that example, the combinations of fuzzy sets are reduced to two rules:

Rule 1: z_1 is "small" and z_2 is "small" and ...

Rule 2: z_1 is "big" and z_2 is "big" and ...

In fact, one of the premise variables becomes irrelevant in the computation of how many rules are.

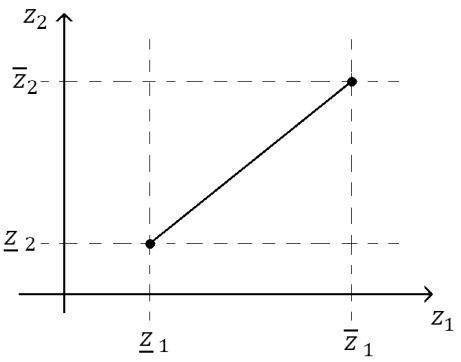


FIGURE A.3: Example of premise variables whit linear relationship

A.4 Membership functions on rectangular and triangular polytopes

Membership functions can be defined in many ways depending on how the fuzzy subset (the subjective label "big", "small", etc.) is related with the premise variables. In this work, the membership functions are linear functions of the premise variables and bounds, so the method of obtaining the membership functions can be generalized for the case of rectangular and triangular polytopes (see Section A.3).

- **Rectangular polytope**

Let define the subset of premise variables $[z_1, z_2, \ldots, z_n]$ which are classified in the "case 1" shown in Appendix A.3, i.e. all the combinations of lower and upper bounds of the premise variables should be considered. In this case, each premise variable z_i has two membership functions: $M_{i1}(z_i(t))$ measures the degree of membership (a number between 0 and 1) to the lower bound value $\underline{z}_i$ and $M_{i2}(z_i(t))$ is related with the upper bound value $\overline{z}_i$.

We want $M_{i1}(z_i(t))$ and $M_{i2}(z_i(t))$ be such the following equations are satisfied [3]:

$$\begin{cases} z_i = M_{i1}(z_i(t))\ \underline{z}_i + M_{i2}(z_i(t))\ \overline{z}_i \\ M_{i1}(z_i(t)) + M_{i2}(z_i(t)) = 1 \end{cases} \tag{A.13}$$

Solving (A.13), the following two membership functions are obtained:

$$M_{i1}(z_i(t)) = \frac{\overline{z}_i - z_i}{\overline{z}_i - \underline{z}_i}, \quad M_{i2}(z_i(t)) = \frac{z_i - \underline{z}_i}{\overline{z}_i - \underline{z}_i} \tag{A.14}$$

In Figure A.4 functions (A.14) are plotted. Note that $M_{i1} = 1$ and $M_{i2} = 0$ when $z_i = \underline{z}_i$, and otherwise when $z_i = \overline{z}_i$.

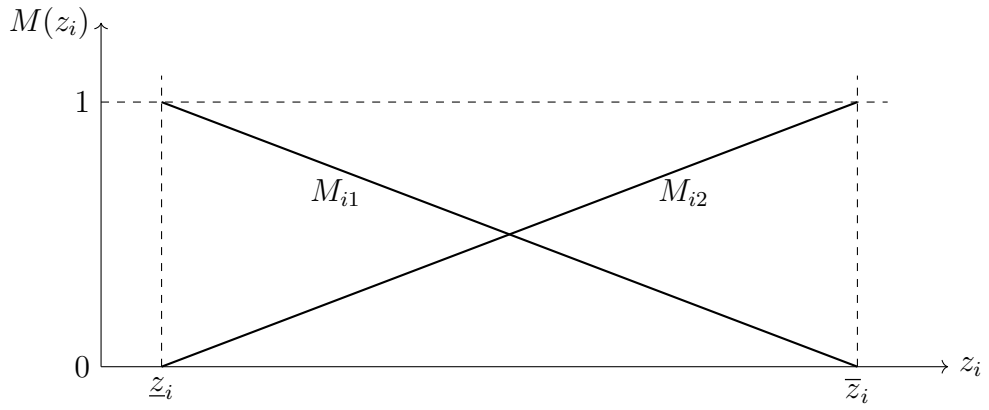


FIGURE A.4: Membership functions of a premise variable. Rectangular polytope approach

- **Triangular polytope**

In the case when two premise variables are dependent and the dependency can be constrained in a triangle as it is shown in Figure A.2, one combination of bounds (vertex) can be omitted. In the case of a rectangle polytope the premise variables are computed independently from two membership functions (A.13). Now, instead of having two membership functions for each premise variable, there are three membership functions for each pair of premise variables.

Let define a pair of premise variables (z_1,z_2) which are dependent, and the dependency has one of the forms shown in Figure A.2(a) or Figure A.2(b). Similarly to the case of a rectangle polytope, the vector of premise variables $[z_1 \ z_2]^T$ is obtained by a linear combination of three vertices: $[\underline{z}_1 \ \underline{z}_2]^T$, $[\overline{z}_1 \ \underline{z}_2]^T$ and $[\overline{z}_1 \ \overline{z}_2]^T$. The combination is done multiplying each vertex by a membership function, that can be named as N_{j1}, N_{j2} and

N_{j3}, where j is a number associated with the pair (z_1, z_2). Then the system of equations in this case is:

$$\begin{cases} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = N_{j1} \begin{bmatrix} \underline{z}_1 \\ \underline{z}_2 \end{bmatrix} + N_{j2} \begin{bmatrix} \overline{z}_1 \\ \underline{z}_2 \end{bmatrix} + N_{j3} \begin{bmatrix} \overline{z}_1 \\ \overline{z}_2 \end{bmatrix} \\ N_{j1} + N_{j2} + N_{j3} = 1 \end{cases} \tag{A.15}$$

The solution of (A.15) for N_{j1}, N_{j2} and N_{j3} is

$$\begin{cases} N_{j1} = M_{11} = \dfrac{\overline{z}_1 - z_1}{\overline{z}_1 - \underline{z}_1} \\ N_{j2} = M_{21} - M_{11} = \dfrac{\overline{z}_2 - z_2}{\overline{z}_2 - \underline{z}_2} - \dfrac{\overline{z}_1 - z_1}{\overline{z}_1 - \underline{z}_1} \\ N_{j3} = M_{22} = \dfrac{z_2 - \underline{z}_2}{\overline{z}_2 - \underline{z}_2} \end{cases} \tag{A.16}$$

Note that they can be written as a function of membership functions like (A.14).

Appendix B

Costs/Sustainability

B.1 Costs

This project has two types of costs associated: the one related with the human resources, which takes into account the working time spent by the engineer, and the costs related with the use of material or software.

For computing the first cost it has been taken into account how much time has been spent in the development of the project, the writing and the use of the software for simulations.

Concept	**Hours**	**€/h**	**Cost**
Development	200 h	30€/h	6.000 €
Writing	100 h	30€/h	3.000 €
Simulation	50 h	30€/h	1.500 €
Total	350 h		10.500 €

Table B.1: Human Resources costs

The main resource used in this thesis is Matlab® and Simulink® software. The license for students for academic purposes is 35 €.

B.2 Environmental impact

In this section some considerations about sustainability regarding the use of UAV's are explained.

- **Use of batteries**. Besides the specifications explained in this work for the design of the controllers, one additional criteria could be the reduction of batteries usage. Extending life of batteries could reduce the amount of waste produced as a result of activities based on UAV's.
- **UAV's and Wildlife**. One important usage of UAV's is the surveillance of wildlife areas. The designer should consider the effect of noise pollution and the potential interaction of the quadrotor with birds or other animals.
- **Safety**: In an urban environment it should also be considered the interaction of the quadrotor with people. From the point of view of the design process some limitations regarding the accelerations or velocities could be applied for safety reasons.

Bibliography

[1] Damiano Rotondo. *Advances in Gain-Scheduling ans Fault Tolerant Control Techniques*. PhD thesis, Universitat Politècnica de Catalunya, September 2015.

[2] Pierpaolo Murrieri Samir Bouabdallah and Roland Siegwart. Design and Control of an Indoor Micro Quadrotor. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 5:4393–4398, 2004.

[3] Kazuo Tanaka and Hua O. Wang. *Fuzzy Control Systems Design and Analysis. A Linear Matrix Inequality Approach*. John Wiley & sons, Inc., 2001.

[4] K.M.Passino and S. Yurkovich. *Fuzzy Control*, chapter 2, pages 61–77. Addison Wesley, 1998.

[5] Hicham Khebbache Fouad Yacef, Omar Bouhali and Fares Boudjema. Takagi-Sugeno Model for a Quadrotor Modelling and Control using Nonlinear State Feedback Controller. *International Journal of Control Theory and Computer Modelling (IJCTCM)*, 2(3):9–24, May 2012.

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

[7] Tommaso Bresciani. Modelling, Identification and Control of a Quadrotor Helicopter. Master's thesis, Lund University, October 2008.

[8] Yucai Zhu. *Multivariable System Identification For Process Control*, chapter 3, pages 41–45. Elsevier, 2001.

[9] Vicenç Puig Damiano Rotondo, Fatiha Nejjari. Robust Quasi-LPV Model Reference FTC of a Quadrotor UAV Subject to Actuator Faults. *International Journal of Applied Mathematics and Computer Science*, 25(1):7–22, March 2015.

[10] Samir Bouabdallah. *Design and control of quadrotors with application to autonomous flying*. PhD thesis, École Polytechnique Fédérale de Lausanne, February 2007.

[11] Daniel Warren Mellinger. Trajectory Generation and Control for Quadrotors. *Publicly Accessible Penn Dissertations*, Paper 547:10–12, 2012.