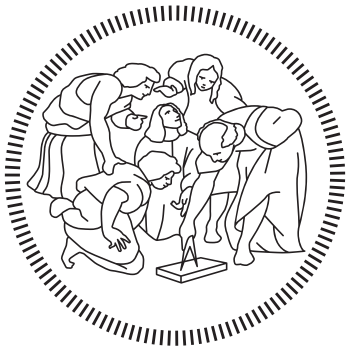
Politecnico di Milano
SCHOOL OF INDUSTRIAL AND INFORMATION ENGINEERING
Master of Science in Automation and Control Engineering



Optimization of reference models for data-driven controller tuning

Advisor: Prof. Marco LOVERA

Thesis by:
David E. Luna S. **Matr.** 913300

Academic Year 2020–2021

Ai miei cari, vires acquirit eundo...

Acknowledgments

First of all I would like to thank my parents, who have always been a guide for me and have accompanied me on this long journey. ¡Gracias por permitirme cumplir mis sueños! To my brother who has been a pillar in my life, you can always count on me Billy.

A big thanks to my friends, with whom I have been able to share unforgettable moments, frustrations and successes, I hope to continue shaping my life with you.

To Politecnico di Milano and its contributors, for opening its doors to international students and for providing quality education in each of us.

Finally, to Professor Marco Lovera for the possibility to work on this project, for his enthusiasm in the topic, and for the guidance during the development of this work.

Abstract

The latest technological advances in the field of unmanned aerial vehicles (UAV) have promoted a great interest in the development of aircraft due to its wide range of applicability. Therefore, controllability and automation problems have become one of the first lines of research in this area.

The basis of the control of these aircraft is usually defined by attitude control, where theoretical or experimental model approaches are frequently used for the development and tuning of the controllers. However, this approach entails many limitations corresponding to the fidelity of the model.

To solve these limitations, data-driven control techniques have been developed dealing with the uncertainty of the system to be controlled, from which a reference model is used for the system to replicate its dynamics. Although this methodology is functional for a large part of systems, in the case of underactuated non-linear systems, the choice of a suitable reference model turns out to be critical for the stability and performance of the aircraft.

In view of this problem, the introduction of optimization algorithms within the framework of data-driven controllers turns out to be considered as an alternative approach for the design and synthesis of the same. In this thesis, a two-level optimization method has been implemented within the virtual reference feedback adjustment algorithm to generate a free model alternative for application within the UAV quadcopter, considering a cascade structure of the controller.

The algorithm has been tested using different simulation scenarios to validate the operation and the results generated. Likewise, an experimental test has been conducted with measurement data of an aircraft in controlled flight conditions where an algorithm performance metric is produced within this frame of reference.

Sommario

Gli ultimi progressi tecnologici nel campo dei veicoli aerei senza equipaggio (UAV) hanno promosso un grande interesse nello sviluppo dei velivoli a causa della sua vasta gamma di applicabilità. Pertanto, i problemi di controllabilità e automazione sono diventati una delle prime linee di ricerca in questo settore.

La base del controllo di questi velivoli è solitamente definita dal controllo d'assetto, dove approcci teorici o sperimentali del modello sono frequentemente utilizzati per lo sviluppo e la messa a punto dei controllori. Tuttavia, questo approccio comporta molte limitazioni corrispondenti alla fedeltà del modello.

Per risolvere queste limitazioni, sono state sviluppate tecniche di controllo data-driven che trattano l'incertezza del sistema da controllare, da cui si utilizza un modello di riferimento del sistema per replicarne la dinamica. Sebbene questa metodologia sia funzionale per gran parte dei sistemi, nel caso di sistemi non lineari sottoattuati, la scelta di un modello di riferimento adeguato si rivela critica per la stabilità e le prestazioni dell'aereo.

Alla luce di questo problema, l'introduzione di algoritmi di ottimizzazione nell'ambito dei controllori data-driven risulta essere considerato un approccio alternativo per la progettazione e la sintesi degli stessi. In questa tesi è stato implementato un metodo di ottimizzazione a due livelli all'interno dell'algoritmo di regolazione del feedback di riferimento virtuale per generare un modello alternativo libero da applicare all'interno del quadcopter UAV, considerando una struttura a cascata del controller.

L'algoritmo è stato testato utilizzando diversi scenari di simulazione per convalidare il funzionamento e i risultati generati. Allo stesso modo, un test sperimentale è stato condotto con i dati di misurazione di un velivolo in condizioni di volo controllate, dove una metrica delle prestazioni dell'algoritmo è prodotta in questo quadro di riferimento.

Contents

List of Figures

List of Tables

List of Acronyms

UAV	Unmanned Aerial Vehicles
PID	Proportional-Integral-Derivative
NED	North-East-Down
VRFT	Virtual Reference Feedback Tuning
SISO	Single-Input Single-Output
MIMO	Multi-Input Multi-Output
LTI	Linear Time-Invariant
cbT	Correlation-based Tuning
IFT	Iterative Feedback Tuning
LQ	Linear Quadratic
MPC	Model Predictive Control
IV	Instrumental Variable
EIV	Extended Instrumental Variable
PSO	Particle Swarm Optimization
CAM-ES	Covariance Matrix Adaptation Evolution Strategy
PRBS	Pseudo-Random Binary Sequence
UC	Unfalsified Control
PEM	Prediction Error Methods
PBSID	Predictor-Based Subspace Identification
FCU	Flight Control Unit
MEMS	Electro-Mechanical Systems
KF	Kalman Filter
GPS	Global Positioning System
PWM	Pulse-Width Modulation
ESC	Electronic Speed controllers
BLDC	Brushless DC
MOCAP	Motion Capture System

Introduction

The growing development of unmanned aerial vehicles (UAV) in recent years promoted by technological and theoretical advances in terms of dynamics and control has led aircraft such as quadcopters to be a tool that offers a great variety of alternatives in their applications ranging from military or soil recognition for buildings through cinematography, or recreational use. This great variety of applications requires the aircraft to have specific characteristics in terms of safety and flight handling, hence, the control designed must be optimal for each type of aircraft.

The attitude control paradigm is the main basis for the synthesis of controllers in charge of mastering the dynamics of the quadrotors such as position, linear or angular velocity which can be achieved through the generation of torque in the desired axis. The construction and design of the quadrotors provides different benefits in terms of dynamics, the symmetry of the aircraft is able to decouple each component of the inertia while the blade structure which turn in opposite directions, deals with the gyroscopic effect while provides more unstable responses due to its increase in maneuverability. The variation of the thrust generated by each of the rotors is the way to control the attitude itself, this is achieved through the angular velocity of pitch and roll from which all the other dynamics can be derived.

At the same time, the safety features of the aircraft rely on the stabilization, disturbance, and noise rejection as well as robustness of the implemented controllers which in combination with the attitude control specifications represents a significant effort on the control design framework to provide the necessary and sufficient conditions so the systems in control can achieve stable flight conditions.

Nowadays, the tendency to build smaller UAVs is greater, which implies that the dynamic models of each aircraft must be more precise as the external excitations have more influence, making the complexity greater, and thus, the application within the control framework becomes increasingly difficult. The mathematical models established in the theory are based mainly on physical principles or statistical methods in charge of identifying the parameters related to the dynamic system through the usage of measured data. However, this type of modeling tends to have certain drawbacks as the complexity of the system increases, within these drawbacks related to physical models, there is the sub-modeling of the system where dynamics that affect the real model are not taken into account due to lack

of information, approximations, linearization, among others. This causes that at the time of implementing the controllers an adequate or optimal performance is not obtained. Moreover, this type of modeling does not usually consider wear and operating conditions.

As far as statistical models are concerned, the unrepresentative excitation of the system can generate under modeling or misidentification of the system. The introduction of uncertainty factors in the measured data such as noise, disturbances, and uncertainty propagation within the parameters is another drawback of the statistical models.

Improper modeling of the system has become one of the main limitations of control strategies, and data-driven control techniques have been developed to overcome these limitations. These techniques use optimization methodologies to synthesize model-based controllers through the usage of measured data from operating systems where uncertainty is considered. However, those control methodologies are not able to surpass all the limitations since the necessity of reference models is still a problem that leads to sub-modeling limitations, and therefore, sub-optimal performance.

Another research trend that has been reinforced during the last decades is related to optimization methods, where highly complex problems are solved using new techniques that not only optimize the parameters but also have a high processing capacity that allows the implementation in the UAVs themselves. The use of these optimization algorithms within the control framework generates the possibility of synthesizing controllers without the need to establish a model to refer to, that being used as another layer of the control methodologies based on data, generates a further enhancement on the synthesized controllers.

Thesis structure

The scope of this thesis is to implement and validate optimization techniques for the reference models for data-driven controller tuning in order to get a model-free data-driven methodology. The work is organized as follows:

- **Chapter 1 – Attitude control – UAV:** The principles of the unmanned aerial vehicles dynamics are presented, establishing the proper derivation of the model of a quadrotor. Furthermore, the logics of the attitude control are introduced showing the importance of the actuator dynamics giving a final definition of the dynamic model. An overview of PID controllers is shown providing motivations and limitations to finally introduce the control structure for the attitude control.

- **Chapter 2 – Data-driven methods:** An introduction of data-driven control framework is given, focusing in the Virtual-Reference-Feedback-Tunning and its motivations related to the model reference control paradigm. The

proper mathematical derivation is fully detailed giving an emphasis on noisy realizations and how to the deal with them, additionally, the VRFT algorithm is extended to cascade control structures. To conclude, a description of the main drawbacks of the choice of reference models is presented.

- **Chapter 3 – Optimization methods:** The two optimization algorithms (Particle swarm optimization and Covariance matrix adaptation strategy) are formally presented showing its implementation within the VRFT algorithm and its extension for the cascade VRFT structure.

- **Chapter 4 – Simulation:** Multiple simulation scenarios are used to validate the algorithm results showing its performance capabilities with respect to the nominal algorithms. A simulation of a real quadrotor is presented where the algorithm is tested to give an overview of the algorithm within UAVs framework.

- **Chapter 5 – Drone platforms:** A general overview of the experimental conditions is fully depicted showing the main features of the drone, its software and hardware as well as the facilities where its measured.

- **Chapter 6 – Experimental testing and results:** The experimental tests using the Cascade model-free VRFT algorithm into the ANT-X drone platform using the in-flight experimental data are presented, portraying the range of implementation and its limitations.

- **Chapter 7 – Conclusions:** The conclusions are formulated following the results given by the simulations and the experimental testing where the analysis is cited to give some insights on the future research direction.

Chapter 1

Attitude control - UAV

This chapter introduces the principles of the quadrotor unmanned aerial vehicles (UAV) dynamics, which serve as a basis for generating control methodologies to obtain the desired flight behavior. Hence, preliminary notions within aerial mobile robotics are introduced to derivate the quadrotor dynamics through a mathematical model. Furthermore, an overview of the main goal of the attitude control is given and the control design paradigm is discussed within the PID control structure.

1.1 Mobile robotics

The unmanned aerial vehicles can be classified as mobile robots, understanding robots as machines capable of carrying out a complex series of actions guided or automatically. Within this field of study, the basic problem formulation starts from the representation of a rigid body in the space. To achieve this goal, cartesian frames are introduced, and the most common technique is to consider a reference frame and attach a second frame to the body, such that the problem resides on the characterization of the position and orientation of a frame with respect to the other one.

For the UAVs, there are several choices for the reference frames that depend on the application used, and each frame has its characteristics useful for simplifications of the model. In this case, two reference systems are commonly used, the first one called *North-East-Down* (NED) -also called the inertial reference frame, since the first Newton's law is considered valid due to the fixed coordinate system- the reference frame O_{NED} originates as a fixed point on the surface of the earth and its orientation is given by its name (NED). The second reference frame is called the *Aircraft body center* O_{ABC}, and it is characterized to be a mobile reference system where its origin is aligned with the center of mass of the aircraft and its axes X_{ABC}, Y_{ABC} & Z_{ABC} points towards the front of the aircraft, right propeller and downwards; a detailed illustration is shown in Fig 1.1.

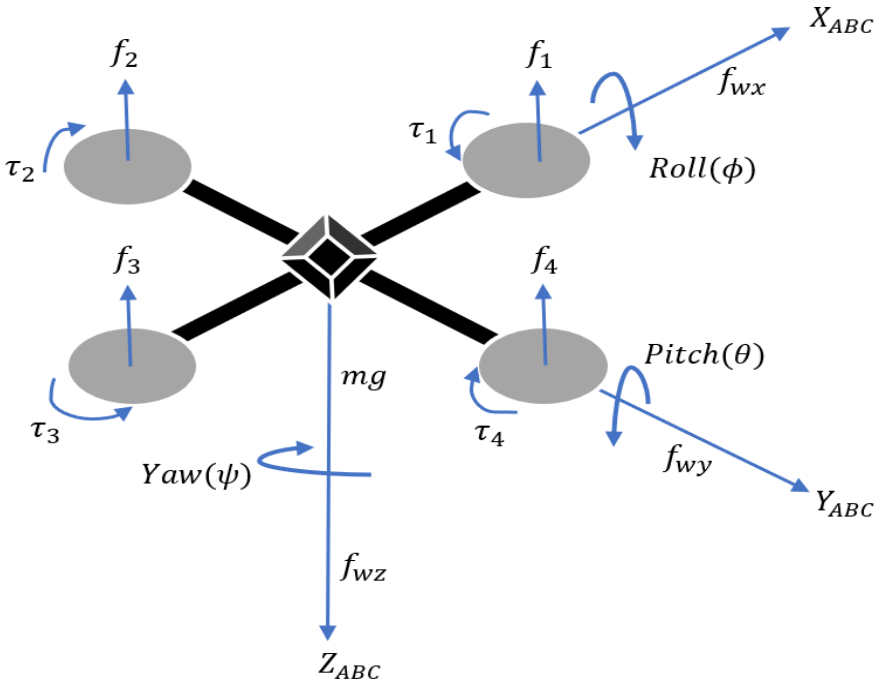


Figure 1.1: Aircraft body center frame

Given the respective reference frame, the representation of the position is made with the components of the origin of the body frame for the fixed frame, while the representation of the orientation can be made considering unit length vectors along the axes of the rotating frame and evaluating them in the reference frame. The gathering of the three-unit vectors is known as rotation matrix R of the frame $\{x', y', z'\}$ with respect to the frame $\{x, y, z\}$ since the following relations holds:

$$x'^T x' = 1, y'^T y' = 1, z'^T z' = 1 \tag{1.1}$$

$$x'^T y' = 0, y'^T z' = 0, z'^T x' = 0 \tag{1.2}$$

Thus, the rotational matrix is orthogonal since:

$$R^T R = I(R^T = R^{-1}) \tag{1.3}$$

Note that the initial formulation of the rotation matrix uses nine parameters to represent the orientation of a frame with respect to another one from which six constraints exist, adding complexity to the general problem, conversely, another minimal representation of the orientation known as ZYX Euler angles [3] described by three independent parameters $\phi \in \theta \in \&\ \psi \in$ is used.

Euler angles represent a sequence of three elemental rotations shown in Equations (1.4)(1.5)(1.6) from which any possible orientation can be described from the combination of each element.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{1.4}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\phi) \end{bmatrix} \tag{1.5}$$

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.6}$$

Additionally, the position coordinates and the mobile body reference can be defined with the Euler ZYX rotation matrix as:

$$R_{zyx}(\phi, \theta, \psi) = R_z(\psi) \cdot R_y(\theta) \cdot R_x(\phi) \tag{1.7}$$

Moreover, by using the fixed and the mobile frames, the resulting Euler angle is rewritten as:

$$R_{NED}(\phi, \theta, \psi) = \begin{bmatrix} c(\theta)c(\psi) & s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) \\ c(\theta)s(\psi) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) \\ -s(\theta) & s(\phi)c(\theta) & c(\phi)c(\theta) \end{bmatrix} \tag{1.8}$$

Where $s(\cdot)$ stands for $\sin(\cdot)$ and $c(\cdot)$ for the $\cos(\cdot)$

1.2 Model of the quadrotor

The mathematical model of the quadrotor is derived using Newton and Euler equations for the motion of a rigid body in a 3D space. Thus, the definition of the position for both reference frames is given by:

$$P_{O_{NED}} = \begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \psi \end{bmatrix} \tag{1.9}$$

$$P_{ABC} = \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} \tag{1.10}$$

Where the first three terms of Equation (1.9) represent the linear position, and the other three, the angular positions. Likewise, for Equation (1.10), the first three terms represent the linear velocity and the other three, the angular velocity linked to the body frame. Following the body dynamics, a relationship between the two frames can be described as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R_{NED}\left(\phi, \theta, \psi\right) \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{1.11}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = T_{NED}\left(\phi, \theta, \psi\right) \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{1.12}$$

In which T_{NED} is the transformation matrix for the angular positions that is constrained by the so-called *Gimbal lock problem,* where one degree of freedom is lost when the axes of two of the three gimbals are placed in parallel configuration, generating a two-dimensional space. To deal with this problem, a fourth rotational axis can be considered as shown in [4].

$$T_{NED}\left(\phi, \theta, \psi\right) = \begin{bmatrix} 1 & s\left(\phi\right) t(\theta) & c\left(\phi\right) t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & \frac{s(\phi)}{c(\theta)} & \frac{c(\phi)}{c(\theta)} \end{bmatrix} \tag{1.13}$$

Where $t\left(\cdot\right) is the \tan\left(\cdot\right)$

To apply Newton and Euler equations, it is necessary to study the forces and moments acting on the body frame. Furthermore, Fig 1.1 shows that three external forces are actuating on the system, the first one being the gravitational force F_g, which is related to the fixed axis; the second one is the total thrust generated by rotors F_{Mt} , and finally the wind forces F_w actuating on each axis of the quadrotor. Similarly, the external moments acting on the quadrotor are the torques generated by the differential rotor speeds τ_{Mt}, the gyroscopic moments τ_G -which is a term related to the rotation of the rotors and the vehicle body-, and finally, the wind torques τ_w.

Furthermore, the external forces are defined as:

$$F_{ABC} = F_g R^T \cdot \widehat{z}_{NED} - F_{Mt} \cdot \widehat{z}_{ABC} + F_w \tag{1.14}$$

While the external moments are given by:

$$m_{ABC} = \tau_{Mt} - \tau_G + \tau_w \tag{1.15}$$

Finally, the dynamic model of the quadrotor in the body frame is derived from the second Newton's law, as follows:

$$m\left(\begin{bmatrix} p \\ q \\ r \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}\right) = F_{ABC} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} \tag{1.16}$$

Considering the symmetry of the aircraft, the inertial tensor is:

$$J_q = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix} \tag{1.17}$$

Thus, the external moment equation can be formulated using Euler's equations:

$$J_q \cdot \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \left(J_q \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix}\right) = m_{ABC} = \begin{bmatrix} M_{roll} \\ M_{pitch} \\ M_{yaw} \end{bmatrix} \tag{1.18}$$

Consequently,

$$\begin{bmatrix} F_x \\ F_y \\ F_z \\ M_{roll} \\ M_{pitch} \\ M_{yaw} \end{bmatrix} = \begin{bmatrix} -mgs\left(\theta\right) + f_{wx} \\ mg\left[c\left(\theta\right)s\left(\phi\right)\right] + f_{wy} \\ mg\left[c\left(\theta\right)c\left(\phi\right)\right] + f_{wz} - F_{Mt} \\ \tau_{Mtx} - \tau_{Gx} + \tau_{wx} \\ \tau_{Mty} - \tau_{Gy} + \tau_{wy} \\ \tau_{Mtz} - \tau_{Gz} + \tau_{wz} \end{bmatrix} = \begin{bmatrix} m(\dot{u} + qw - rv) \\ m(\dot{v} - pw + ru) \\ m(\dot{w} + pv - qu) \\ \dot{p}J_{xx} - qrJ_{yy} + qrJ_{zz} \\ \dot{q}J_{yy} + prJ_{xx} - prJ_{zz} \\ \dot{r}J_{zz} - pqJ_{xx} + pqJ_{yy} \end{bmatrix} \tag{1.19}$$

1.3 Control design framework

The attitude control paradigm on quadrotors is ruled by the dynamics of the propellers present in the aircraft. The speed control of the motors allows the generation of moments and forces that are related to the thrust given by the rotors, the pitching and rolling moments produced by differential speed on the rotors while the yawing moment is counteracted when two rotors rotate in opposite directions, as shown in Fig 1.1. Additionally, the motion of the quadrotors in the space is described by six degrees of freedom, divided into three translational degrees, namely, forward and backward, lateral and vertical movements, and three angular degrees, described as roll, pitch, and yaw movements. Observe that a quadrotor is an underactuated nonlinear complex system [5] due to the difference between inputs (Rotor speeds) and six outputs (Forces and Moments). Usually, the quadrotor is manipulated using four basic movements, specifically, according to the notation of the propeller in Fig 1.1 the movements are defined as:

- **Thrust movement** occurs when all four rotors have the same speed.
- **Pitch movement** takes place when rotor 1 speed is lower than the rest, rotor 3 speed is higher and rotor 2 & 4 is equivalent.
- **Roll movement** uses the same dynamics as the pitch movement and instead of rotor 1 being the lower and rotor 3 being the higher, the rotors 2 and 4 take those actions respectively.
- **Yaw movement** is generated by a reactive torque produced by the different speeds on the four rotors, to get a counterclockwise yaw movement, the rotor speed of 1 and 3 have the same magnitude and are lower than the 2 and 4 that also have the same magnitude.

1.3.1 Actuator dynamics

According to the previous motivation, it is essential to define the proper models of the propellers to give the dynamic model of the quadrotor the necessary inputs to control the motion of the system. Therefore, let the control inputs be defined by the speed of the rotors Ω that consider four degrees of freedom, which account for the vertical thrust and the 3D angular motions, providing the following formulation:

$$\begin{bmatrix} F_{Mt} \\ \tau_{Mtx} \\ \tau_{Mty} \\ \tau_{Mtz} \end{bmatrix} = \begin{bmatrix} t_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ t_f l(\Omega_3^2 - \Omega_1^2) \\ t_f l(\Omega_4^2 - \Omega_2^2) \\ d_f(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 + \Omega_3^2) \end{bmatrix} \tag{1.20}$$

Where $t_f \wedge d_f$ are the thrust and drag factors respectively, while l is the distance between the propeller and the center of the aircraft. Introducing the actuator dynamics on Equation (1.20) into (1.19), the dynamic model of the quadrotor becomes:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \\ M_{roll} \\ M_{pitch} \\ M_{yaw} \end{bmatrix} = \begin{bmatrix} -mgs(\theta) + f_{wx} \\ mg[c(\theta)s(\varphi)] + f_{wy} \\ mg[c(\theta)c(\varphi)] + f_{wz} - t_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ t_f l(\Omega_3^2 - \Omega_1^2) - \tau_{Gx} + \tau_{wx} \\ t_f l(\Omega_4^2 - \Omega_2^2) - \tau_{Gy} + \tau_{wy} \\ d_f(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 + \Omega_3^2) - \tau_{Gz} + \tau_{wz} \end{bmatrix} = \begin{bmatrix} m(\dot{u} + qw - rv) \\ m(\dot{v} - pw + ru) \\ m(\dot{w} + pv - qu) \\ \dot{p}J_{xx} - qrJ_{yy} + qrJ_{zz} \\ \dot{q}J_{yy} + prJ_{xx} - prJ_{zz} \\ \dot{r}J_{zz} - pqJ_{xx} + pqJ_{yy} \end{bmatrix} \tag{1.21}$$

As for the control framework, Equation (1.21) can be rewritten into space-state model described in [6] to formulate the proper analysis and design which are outside the focus of this work, the reader is referred to [6][7][8] for further details on this topic.

1.3.2 Control configuration

The controller used on the quadcopter is the widely known PID controller, given that nowadays more than 95% of all industrial control problems are solved by PID control [9], due to the ease of implementation and performance capabilities. According to the extensive implementation in the industrial sector, the knowledge about this type of controller in terms of design, tuning, extension, advantages and disadvantages is already well-documented, the reader is referred to the literature in [10] & [11] for a more detailed description of this type of controllers.

PID controllers get their name according to their main characteristics of construction and operation, P stands for a proportional -term that provides inputs that correct the "current" errors-, the I is the integral term, which ensures steady state zero error, and finally, the derivative term D that provides "anticipation of upcoming changes". More formally, the PID controller is defined as:

$$u\left(t\right) = k_p e\left(t\right) + k_i \int_0^t e\left(\tau\right) d\tau + k_d \frac{de\left(t\right)}{dt} \tag{1.22}$$

Where $k_p, k_i \wedge k_d$ represent the proportional, integral, and derivative gains, while $e(t)$ is the signal of error at time t, which is the difference between the desired signal and the output signal of the plant; In contrast, $u(t)$ is the output signal of the controller or as usually used, the input signal for the plant. The PID controller is commonly transformed into the Laplace-domain as:

$$H_{ue}\left(s\right) = K_P + K_I \frac{1}{s} + K_D s = k\left(1 + \frac{1}{T_i s} + T_d s\right) = \frac{kT_D}{T_i} \frac{\left(s + \frac{1}{T_i}\right)\left(s + \frac{1}{T_D}\right)}{s} \tag{1.23}$$

At the same time, the controller can be discretized using several methods such as the Forward Euler Discretization defined by:

$$z = e^{sT_s} \simeq 1 + sT_s \tag{1.24}$$

Hence,

$$H_{ue}\left(z\right) = K_p + K_I \frac{T_s}{z-1} + K_d \frac{z-1}{T_s} \tag{1.25}$$

Note that the controller is capable to give high gain at low frequency and phase margin at high frequency, also it can be designed to cover a specific bandwidth placing the poles of the transfer function in (1.23).

The formulation previously described accounts for the so-called parallel structure of PID controllers shown in Fig 1.2. However, the *set-point kick phenomenon* happens when the reference signal is a step function, where the presence of an approximation of a pure derivative action makes the variable $u\left(t\right)$ to involve a sharp pulse function, instead of an impulse function which is not acceptable in the UAV framework. To overcome this phenomenon, the idea of a PI-D control

shown in Fig 1.3 has been developed, where the derivative action is fed only by the feedback signal $Y(s)$ and not the reference signal $R(s)$. Thus, the output signal of the controller becomes:

$$U(s) = K_p\left(1 + \frac{1}{T_i s}\right) R(s) - K_p(1 + \frac{1}{T_i s} + T_d s)Y(s) \tag{1.26}$$

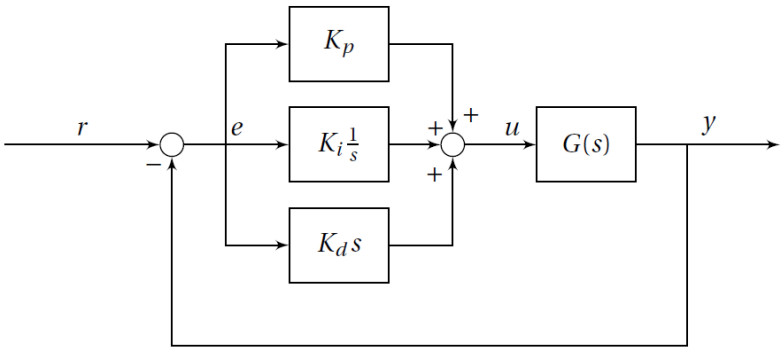


Figure 1.2: PID Controller

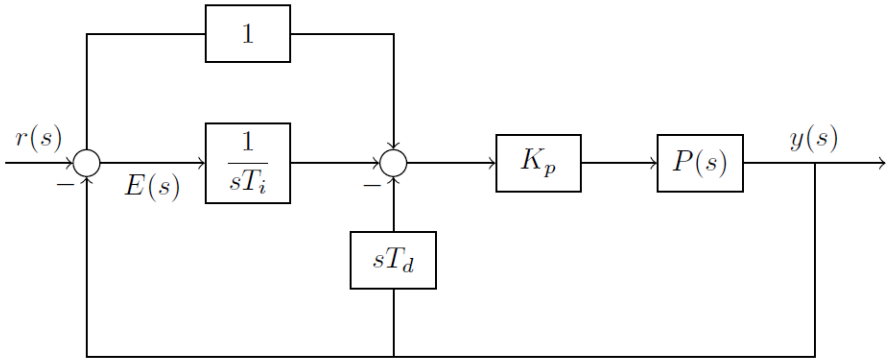


Figure 1.3: PI-D Controller

The introduction of PID controllers into the attitude control framework can be achieved using a cascade controller structure, where the process variables can be the roll and pitch angles depending on the scope of the problem. Furthermore, it can be a single-input single-output (SISO) scope due to decoupled attitude dynamics from Equation (1.17), while a multi-input multi-output (MIMO) extension is also valid especially when coupled dynamics diminish the performance of the system. The control variables are the angular rates and the external moments generated by the propellers, while the cascade control loop is divided into two feedback loops; the outer loop, which is fed by the angular positions and uses a P controller to provide the reference signal for the inner loop -which in this case

are the angular rates- and the inner loop reference signal is processed by a PI-D controller to feed the angular rate dynamics of the quadrotor.

The cascade control structure is particularly useful for this application according to two intrinsic characteristics of the quadrotor control paradigm. The first one is related to the difference between the attitude rate dynamics and angular rate dynamics which, as can be seen from Equation (1.21), determines the attitude of the quadrotor, swhilst the second characteristic is the availability of the data, where measurement for both angular and attitude rates are reliably evaluated. Conclusively, the final control structure used is presented in Fig 1.4.

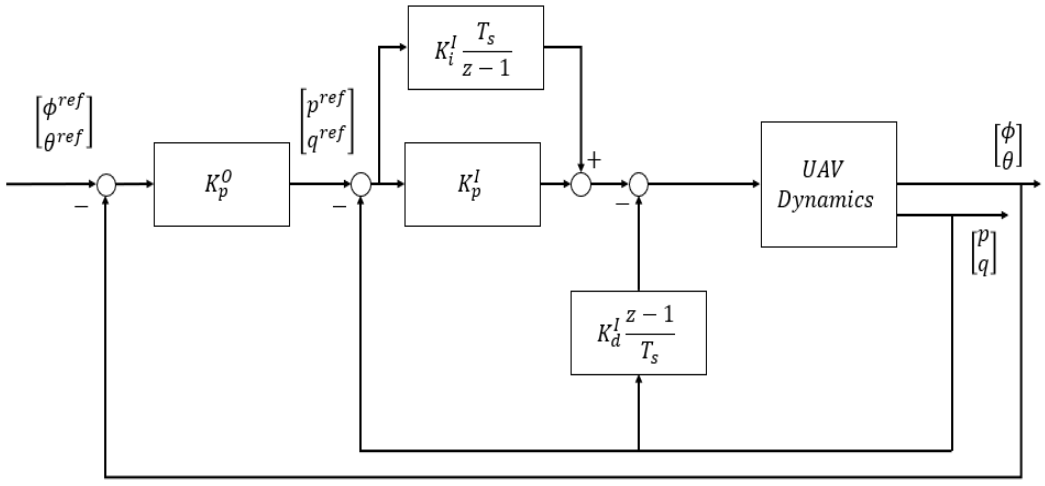


Figure 1.4: Quadrotor cascade P, PI-D controller

Chapter 2

Data-driven methods

This chapter is dedicated to the introduction of data-driven methods, showing the latest developments, features, and advantages that they have concerning the commonly used methods such as model-based control synthesis.

Furthermore, the Virtual Reference Feedback Tuning (VRFT) is meticulously derived for Single-Input Single-Output (SISO) Linear Time-Invariant (LTI) problems starting from its motivations related to the model reference scope, followed by its mathematical derivation, the different setup for the experimental realization, and finishing with a generalization of the algorithm and its usage on the cascade control framework.

2.1 Introduction

The data-driven methodology in the framework of control can be thought of as a robust controller designed according to an approach having two levels of abstractions, as shown in Fig 2.1. The upper level prescribes that the controller is formed by two units with different aims: one unit (control unit) selects the control action to apply to the system. This unit is not determined though as it contains parameters that must be tuned; a second unit (tuning unit) is on duty to perform the parameter tuning. This fundamental split is at the basis of the concept itself of an adaptive system. The lower level consists instead in selecting specific algorithms for the two units.

Nonetheless, the data-based methodology covers a very large spectrum of applicability, so it is necessary to define or classify the type of controllers that can be generated. Consequently, different aspects are considered, such as the variability concerning the time in which the controller estimates are tuned. In this particular case, there is the online type of adaptability, where the tuning unit provides updated estimates of the system at each instant of time, and these new estimates are incorporated and used in the controller unit for the selection of control actions. In contrast, when the controller is only sporadically updated, or the controller

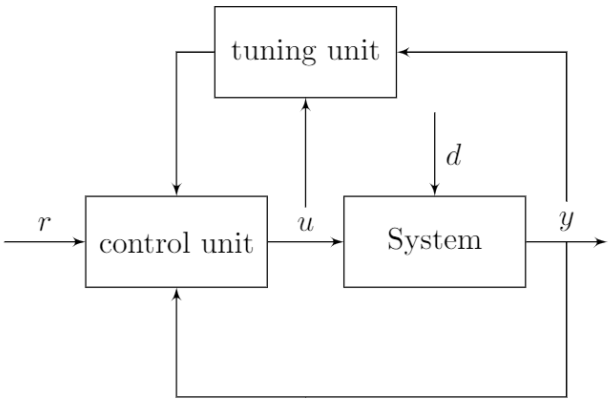


Figure 2.1: Data-Driven control topology

implementation and data acquisition phases are widely separated, it is considered offline adaptation: after data collection, the operation of the control system stops, and the controller is readjusted according to the tuning system parameters.

Off-line adaptation gains an important advantage over online adaptation: an online adaptive controller is a truly complex nonlinear device whose behavior can be hardly predicted in non-standard operating conditions. Moreover, even stability is difficult to ascertain under general circumstances. In off-line adaptation, the re-tuned controller can be tested before it is implemented so that it can be used more safely.

At the same time, there is another way of classifying the methods that depend on the procedure in which the controller parameters are estimated; the first approach is based on generating an estimate $\widehat{S}_t$ of the system S to optimize the performance measure $J(C, \widehat{S}_t)$, which aims to give the optimizer of $J(C, S)$, this is called the indirect method. On the other hand, the direct method tries to estimate directly from the data the cost $J(C, S)$ as a function of the controller C, and the parameters are obtained by direct optimization of $\widehat{J}_t(C, S)$ with respect to C. Note that the estimate $\widehat{J}_t(C, S)$ can be updated over time. Thus, the direct method has the advantage that the parts of S that do not affect the cost of control $J(C, S)$ are automatically ignored.

The state-of-the-art shows that the research approach is taking place within the group of direct and offline methods due to the advantages previously described and their relative similarity with conventional control methods. Table 2.1 shows the classification of the most used algorithms where VRFT is taken into account as the main methodology for this thesis work due to its non-iterative nature compared to CbT or IFT, as well as the saving of an identification step necessary to obtain results in unfalsified control.

	ONLINE	OFFLINE
DIRECT	Q-Learning	Virtual Reference Feddback Tuning (VRFT) Correlation-based Tuning (cbT) Data-driven unfalsified control
INDIRECT		Recursive Least Squares

Table 2.1: State of the Art Data-driven techniques

2.2 Model reference control

Model reference control is a development mainly based on aircraft control; it is used to specify the desired response of a control system for command input, and consequently, it is considered fundamentally as a command shaping filter that obtains the desired command following. Typically, the model contemplated is a Linear Time-Invariant (LTI) model, even though nonlinear models could be used, the level of complexity in terms of practical considerations and analytical tractability among others making them less appealing to work with. Notice that the chosen model should capture all performance & robustness specifications such as rise time, settling time, phase, and gain margins, so it is assumed that the designer is sufficiently familiar with the system under consideration since the structure, desired response, and parameters of the reference model are chosen concerning the model outputs. [12][13]

The problem formulation regarding the model reference control is based on linear-quadratic (LQ) theory, in an attempt to minimize the error transients between the responses of the actual plant and those of the desired target model. To do so, an H_2 or LQ minimization criterion is used to provide a controller capable of minimizing the mismatch in the frequency domain between the plant and the reference model responses. Thus, the model following control problem is detailed in the following *structured* approach:

Given the model $M(z)$, that is the designed transfer function of the closed-loop from R to Y, the system plant P(z), a linear combination of basis transfer functions $C(z,\theta)$ parametrized with the vector of parameters $\theta \in R^n$, such that $C(\theta) = \{C(z,\theta) = \beta^T(z)\,\theta\}$, and a frequency weight $W(z)$ used to free the mismatch between the *closedloop* and M at frequencies that are important for the control goal. The resulting metric is:

$$J_{MR}(\theta) = \left\| \left(\frac{P(z)\,C(z,\theta)}{1 + P(z)\,c(z,\theta)} - M(z) \right) W(z) \right\|_2^2 \tag{2.1}$$

$$J_{MR}(\theta) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| \frac{P(e^{j\omega})\,C(e^{j\omega},\theta)}{1 + P(e^{j\omega})\,C(e^{j\omega},\theta)} - M\left(e^{j\omega}\right) \right|^2 \left| W\left(e^{j\omega}\right) \right|^2 d\omega \tag{2.2}$$

Hence, the model following, or model reference control problem results in:

$$\theta^*_{MR} = \underset{\theta}{argmin} J_{MR}(\theta) \tag{2.3}$$

Note that the assumption in which a linear combination of transfer functions is used for the controller $C(z, \theta)$ and that the sensitivity function $S(z) = I - M(z)$ is close to the closed-loop sensitivity function for $\theta = \theta^*$, which ensures a convex approximation to the optimization problem. However, this formulation does not guarantee any stability criteria, since the problem focuses solely on the mismatch between the output complementary sensitivity and the reference model.

Moreover, there is another formulation which is the *unstructured* formulation, where the target of optimization is $C(z)$. Nevertheless, this approach carries several disadvantages concerning the structured problem because the solution might not be feasible to implement caused by factors such as: the relative degree of the transfer function showing an anticipative improper system, a high control effort and the presence of high-frequency dynamics. Even if a model reduction could be obtained on the system or the controller, other problems arise in terms of performance and stability.

In essence, the solution of the model reference control problem using the structured approach relies on the *a priori* knowledge of the plant $P(z)$, which in the framework of data-driven and adaptive control remains unknown due to the presence of uncertainty on the system, providing a not well-defined optimization metric $J_{MR}(\theta)$. Therefore, another metric should be entirely constructed based on data to overcome the uncertainty.

2.3 Virtual Reference Feedback Tuning (VRFT)

Virtual reference feedback tuning is a direct and offline data-driven method for controller design, and is based on the framework of the model reference control and attempts to solve the problem related to the uncertain systems through input/output measurements of an unknown plant. This approach was introduced in [14] for a generic nonlinear system, and then the first approach mainly focused on the design of one degree of freedom controllers of a closed-loop linear time-invariant system on [15]. Afterward, it was studied and developed in [16], [17], [18] for several case studies such as procedures for nonlinear control, MIMO LTI control or non-minimum phase systems.

The main feature that VRFT provides among the different approaches is that there is no need for model identification of the plant, and it can be used applying only a single data set generated by the plant. This data set can be generated from a closed or open loop configuration of the experiment given certain conditions that will be discussed later in this work. Additionally, this approach does not require any type of iteration either for the data collection or for the control design. Lastly,

since the VRFT works within the framework of the reference model, it is important to emphasize that prior knowledge of the dynamics of the system is important to design an adequate reference model that provides suitable responses for the system itself.

This method relies on the so-called *virtual reference,* which is a signal that would produce an output equal to measured data output if a perfect controller $C^*(z, \theta)$ is placed in the control loop. Then, the idea is to choose the parameter vector θ in such a way that this behavior is followed at best by the chosen controller and that Equation (2.1) is minimized.

2.3.1 Mathematical derivation

The SISO formulation of the VRFT will be shown motivating a pre-filter of the data, the treatment of the noise and its implementation on cascade control framework. The source of the problem given by Equation(2.1), and its impossibility to get the prior knowledge of the plant $P(z)$, makes this method introduce a different cost function that depends on a collection of input/output data from de plant $D_N = \{(u_t, y_t), t = 1, \ldots, N\}$. The so-called virtual reference $\overline{R}_r$ is the set-point for when the desired response of the closed-loop ($M(z)\overline{R}_r$) is proper for the measured values of the output of $P(z)$, during the experiment y_t. The whole setup is illustrated as:

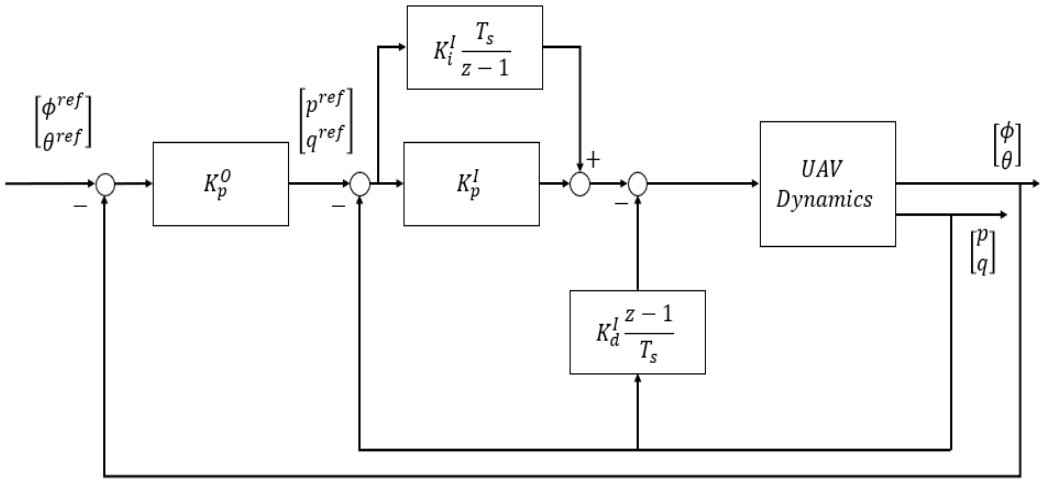


Figure 2.2: VRFT block diagram

Where:

$$\overline{R}_r(i) = M(z)^{-1}\,\overline{y}(i)\; i = 1, \ldots, N \tag{2.4}$$

$$\overline{y}\,(i) = M(z)\overline{R}_r\,(i) \tag{2.5}$$

Introducing the virtual error:

$$\overline{e}_v\,(i) = \overline{R}_r\,(i) - \overline{y}\,(i) \tag{2.6}$$

Observe that the VRFT is an offline method so $\overline{R}_r$ can be always computed even if $M\,(z)^{-1}$ is not proper. A **Necessary condition** for the method is that an optimal controller $C^*(z,\theta)$ is such that:

$$\frac{(P\,(z)\,C^*\,(z,\theta))}{(1+P\,(z)\,C^*\,(z,\theta))} = M(z) \tag{2.7}$$

If this is so:

$$\frac{(P\,(z)\,C^*\,(z,\theta))}{(1+P\,(z)\,C^*\,(z,\theta))}\overline{R}_r\,(i) = M\,(z)\,\overline{R}_r\,(i) = \overline{y}\,(i) \tag{2.8}$$

$$C^*\,(z,\theta)\,\overline{e}_v\,(i) = P\,(z)^{-1}\,\overline{y}\,(i) = \overline{u}\,(i) \tag{2.9}$$

Thus, the optimization cost function constructed based on data becomes:

$$J_{VR}\,(\theta) = \frac{1}{N}\sum_{i=1}^{N} \|\overline{u}\,(i) - C(z,\theta)\overline{e}_v\,(i)\|_2^2 \tag{2.10}$$

However, in most of the cases $C^*\,(z,\theta)$ does not belong to the class of controllers C producing a severe mismatch between Equation(2.1) and Equation(2.10). The idea of introducing a pre-filter on the data is to exploit a new degree of freedom so VRFT will behave close as a model reference control design, a more detailed description is shown in [14]. In order to compare both cost functions, the asymptotic frequency domain expression of Equation (2.10) is described as:

$$\overline{J_{VR}}(\theta) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{|P(e^{j\omega})|^2\,|C^*\,(e^{j\omega}) - C\,(e^{j\omega},\theta)|^2}{|1+P(e^{j\omega})C^*\,(e^{j\omega})|^2}\frac{|L(e^{j\omega})|^2}{|M(e^{j\omega})|^2}\Phi_u\,d\omega \tag{2.11}$$

Where $L(z)$ is the pre-filter and Φ_u is the spectral density of the input u_t which is chosen by the user. Note that an optimal choice of $L(z)$ minimizes the mismatch between Equation(2.11) & Equation(2.12).

$$L^*\left(e^{j\omega}\right) = \frac{|W\,(e^{j\omega})|^2\,|M\,(e^{j\omega})|^2\,|(1-M\,(e^{j\omega}))|^2}{\Phi_u}, \forall\omega\in[-\pi,+\pi] \tag{2.12}$$

With this implementation, the data of Equation(2.10) becomes:

$$\overline{e}_{vL}\,(i) = L\,(z)\,\overline{e}_v\,(i) \tag{2.13}$$

$$\overline{u}_L(i) = L(z)\overline{u}(i) \tag{2.14}$$

Finally, considering a structured controller case with a linear combination of basis transfer functions $C(z,\theta)$, parametrized with the vector of parameters $\theta \in R^n$, such that $C(\theta) = \{C(z,\theta) = \beta^T(z)\,\theta\}$, provides the Equation(2.10) to be:

$$J_{VR}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left\|\overline{u}_L(i) - \beta^T(z)\,\theta\overline{e}_{vL}(i)\right\|_2^2 = \frac{1}{N}\sum_{i=1}^{N}\left\|\overline{u}_L(i) - \theta^T\varphi_L(i)\right\|_2^2 \tag{2.15}$$

Thus, the minimization problem of the cost function turns into a Least Squares minimal equation:

$$\theta^*_{VR} = \underset{\theta}{argmin} J_{VR}(\theta) = \left[\sum_{i=1}^{N}\varphi_L^T(i)\,\varphi_L(i)\right]^{-1}\left[\sum_{i=1}^{N}\varphi_L^T(i)\,\overline{u}_L(i)\right] \tag{2.16}$$

2.3.2 Experiment Setup

The collection of the input/output data from the plant $D_N = \{(u_t, y_t), t = 1, \ldots, N\}$ can be done employing open or closed-loop experiments. Each of one of them can be affected by additive noise $d(t)$ in the output, leading to a biased parameter vector in the case of an open-loop experiment and a correlation between the input and the noise if the experiment is carried out with a closed-loop experiment. This results in a significant deterioration of the performance of the VRFT algorithm, since the cost function of Equation (2.11) may no longer approximate to Equation (2.2). Such effect is described as:

$$\widetilde{y}(t) = P(z)\,u(t) + d(t) \tag{2.17}$$

Additionally for closed loop experiments:

$$\widetilde{e}_L(t) = (I - M(z))\,P(z)\,u(t) + (I - M(z))\,d(t) \tag{2.18}$$

To deal with noisy data the concept of Instrumental Variables (IV) is introduced. The main idea behind the IVs is that "the instruments must be correlated with the regression variables but uncorrelated with the noise" [19]. Providing a reduction in the impact of the noise in the dataset. In the case of VRFT, there are two principal ways to implement the IVs:

- **Repeated experiment method**: Performing a second experiment on the plant using the same input sequence $u(t)$ as in the first experiment provides a secondary output sequence $\widetilde{y}'(t)$ and, assuming that the noise signals on the different realizations are uncorrelated, then, with the instrumental variable and for a dataset D_N with $N \to \infty$, the resulting estimate $\widehat{\theta}$ will be consistent with the noiseless estimate.

- **System identification method:** Identifying a model of the plant using the dataset D_N and retrieving output data from a simulated experiment provides uncorrelated sequences to be used as IVs. Even though the identification procedure has its contradiction related to the direct data-driven methods, it is worth remarking that the only objective of this model is to generate de instrumental variable and, consequently, does not interfere with the design of the controller.

Furthermore, the actual implementation and derivation of the instrumental variables on SISO closed-loop experiments are in [20] and [21]; as for the MIMO implementation, another further step must be made by introducing the Extended Instrumental Variable (EIV) to accomplish the same goal. The reader is referred to [17] for details and comments.

2.3.3 Cascade controller structure

As previously stated, the cascade control framework is commonly used to increase the dynamic performance of systems and specifically in the case of study to control the kinematics related to velocity and position of the system. To transpose this framework into the VRFT **it is necessary** to have at least a minimal prior knowledge of the dynamics of both the inner and the outer feedback loops, since the main approach to deal with this problem is to tune each loop separately by the means of separate model references. The method proposed in [22] states that the dataset $D_N = \{(u_{it}, y_{it}, y_{ot}), t = 1, \ldots, N\}$ is a collection of input and output signals, where u_{it} is the excitation input on the inner loop, y_{it} is the output of the inner loop and y_{ot} is the output of the outer loop. Under the VRFT formulation, the inner loop controller parameters $\widehat{\theta}_i$ can be estimated from D_N, while for the outer loop other further steps are needed considering that the input data of the outer loop is affected by the controller of the inner loop, this relationship is shown in Fig 2.3 and described as:

$$u_{ot} = r_1 = e_1(t) + y_{it}(\widehat{\theta}_i) \tag{2.19}$$

$$e_1(t) = C_2^{-1}\left(\widehat{\theta}_i\right) u_{it}(\widehat{\theta}_i) \tag{2.20}$$

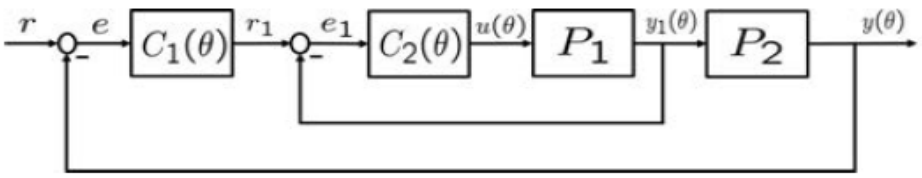


Figure 2.3: Cascade VRFT block diagram

Note that the calculation of Equation (2.20) requires an inversion of the inner controller, implying that a **necessary condition** for the inner controller is that the result is minimum phase. Once this condition is verified, another execution of the VRFT algorithm is made using the calculated input data u_{ot} and the outer error e to generate the outer cost function:

$$J_{VRo}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \|\overline{u}_{ot}(i) - C_1(z,\theta)e(i)\|_2^2 \tag{2.21}$$

To sum up, a generalization of the complete VRFT cascade algorithm is shown in the following pseudo-code.

Cascade SISO VRFT controller

1. Design Φ_{ui}, $W(z)$, $M_i(z)$ and $M_o(z)$
2. Compute the optimal pre-filter $L_i^*(e^{j\omega}) = \frac{|W(e^{j\omega})|^2 |M_i(e^{j\omega})|^2 |(1-M_i(e^{j\omega}))|^2}{\Phi_u}$
3. Compute $\overline{u}_{Li}(i) = L_i(z)\overline{u}(i)$
4. Compute $\overline{e}_{vLi}(i) = L_i(z)\,\overline{e}_{vi}(i)$
5. Identify the plant $\widetilde{P}(z)$ and compute the output $\widetilde{y}_{it} = \widetilde{P}(z)\,u_{it}$
6. Compute the instrumental variables $\xi_{i,t} = C_{2i}(z)\left(M_i(z)^{-1} - I\right)L_i(z)\,\widetilde{y}_t$
7. Compute the parameter vector $\widehat{\theta}_i =? \left[\sum_{i=1}^{N} \xi_{i,t}(i)\,\widetilde{\varphi}_{Li}(i)^T\right]^{-1}\left[\sum_{i=1}^{N} \xi_{i,t}(i)\,\overline{u}_L(i)\right]$
8. Check if minimum-phase, otherwise return to 1
9. Compute $e_1(t) = C_2^{-1}\left(\widehat{\theta}_i\right) u_{it}(\widehat{\theta}_i)$
10. Compute $u_{ot} = r_1 = e_1(t) + y_{it}(\widehat{\theta}_i)$
11. Compute the optimal pre-filter $L_o^*(e^{j\omega}) = \frac{|W(e^{j\omega})|^2 |M_o(e^{j\omega})|^2 |(1-M_o(e^{j\omega}))|^2}{\Phi_u}$
12. Compute $\overline{u}_{Lo}(i) = L_o(z)\overline{u}(i)$
13. Compute $\overline{e}_{vLo}(i) = L_o(z)\,\overline{e}_{vo}(i)$
14. Identify the plant $\widetilde{P}_o(z)$ and compute the output $\widetilde{y}_{ot} = \widetilde{P}(z)\,u_{ot}$
15. Compute the instrumental variables $\xi_{o,t} = C_{1i}(z)\left(M_o(z)^{-1} - I\right)L_o(z)\,\widetilde{y}_t$
16. Compute the parameter vector $\widehat{\theta}_o =? \left[\sum_{i=1}^{N} \xi_{o,t}(i)\,\widetilde{\varphi}_{Lo}(i)^T\right]^{-1}\left[\sum_{i=1}^{N} \xi_{o,t}(i)\,\overline{u}_{Lo}(i)\right]$

Observe that steps 5 and 14 can be substituted for the repeated experiment method only if the data set experiment is carried out in an open loop configuration. In addition, this implementation is made with a structured control design so the structure of the controller $\beta(z)$ must be given before running the code. Finally, the dynamics of the designed inner loop model reference must be faster than the dynamics of the outer loop model reference otherwise the system will never converge.

2.4 Reference models

Within the paradigm of data-driven methods, one of the main characteristics is that there is no need to have prior knowledge about the plant to generate a stabilizing controller. Nevertheless, the substantial influence of the reference model within the VRFT framework makes it mandatory to obtain a minimum of knowledge about the dynamics of the system to be able to generate an adequate reference, where problems such as performance, stability, non-minimum phase plant, among others, can be avoided.

As the main countermeasure for this problem, first or second-order reference models are often implemented depending on the performance restrictions, where metrics such as crossover frequency, damping ratio, or settling time are considered. Even so, when the complexity of the system is higher, and particularly when nonlinear dynamics exist, the reference models are not usually accurate enough to obtain a controller capable of stabilizing the system, and therefore, data-based methods do not perform as intended. This is expected, since there may be optimal reference models that allow an improvement in the operation of the system in which the algorithm is implemented.

Chapter 3

Optimization methods

Within the paradigm of data-driven methods, one of the main characteristics is that there is no need to have prior knowledge about the plant to generate a stabilizing controller. Nevertheless, the substantial influence of the reference model within the VRFT framework makes it mandatory to obtain a minimum of knowledge about the dynamics of the system to be able to generate an adequate reference, where problems such as performance, stability, non-minimum phase plant, among others, can be avoided.

As the main countermeasure for this problem, first or second-order reference models are often implemented depending on the performance restrictions, where metrics such as crossover frequency, damping ratio, or settling time are considered. Even so, when the complexity of the system is higher, and particularly when nonlinear dynamics exist, the reference models are not usually accurate enough to obtain a controller capable of stabilizing the system, and therefore, data-based methods do not perform as intended. This is expected, since there may be optimal reference models that allow an improvement in the operation of the system in which the algorithm is implemented.

PSO is a stochastic population-based optimization method proposed in [23]; it has been widely used due to its ease of implementation and fast convergence. In PSO, the population is composed of several sample entities that are called particles, each one of them is defined by three $D - dimensional$ vectors:

- Current position $\overrightarrow{x}_i$: Contains the position of the particle. The current position is evaluated as a problem solution and determines the quality of the particle.

- Personal best $\overrightarrow{p}_{best}$: Consist of the best position that the particle has achieved during its life span.

- Velocity $\overrightarrow{V}_i$: Represents the direction and length of movement of the particle in the space and it can be seen as a step size for exploration.

Each particle is placed randomly in a search space of a problem or function S, defined by $\{\overrightarrow{y} : l_i \leq y_i \leq u_i\}$ where y_i is the i^{th} dimension of the vector $\overrightarrow{y}$; $u_i \wedge l_i$ are the upper and lower boundaries of the i^{th} dimension, respectively. After evaluating the objective function at its current location, each particle defines the velocity vector depending on its previous positions, the best location found so far, some random perturbations, and most importantly, collecting information of its neighborhood through the interaction between particles that determines the best position explored $\overrightarrow{g}_{best}$. The definition of all velocities is done after all particles evaluate the problem solution, and eventually, all particles converge to an optimum of the cost function. This procedure is illustrated in Fig 3.1.

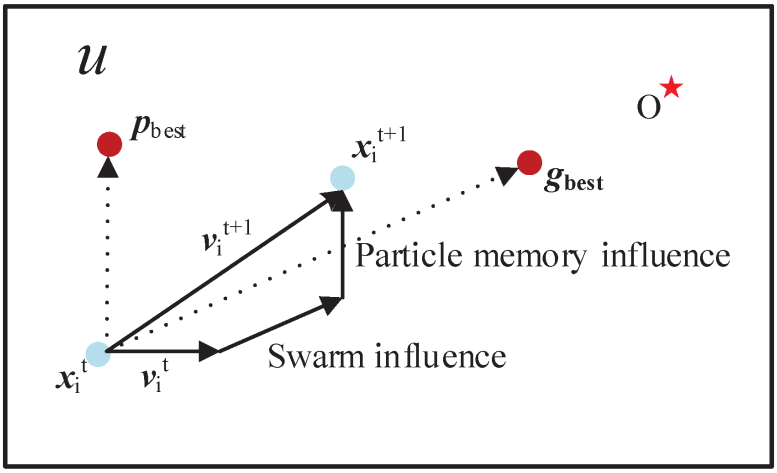


Figure 3.1: PSO particle update [1]

Furthermore, a formal generalization of the algorithm is shown in the following pseudo-code:

PSO algorithm

1. Populate the particle swarm with random initial values on D dimensions
2. Loop:
3. Evaluate the cost function for each particle and set its current position $\overrightarrow{x}_i$
4. Compare the evaluation of $\overrightarrow{x}_i$ with c_{best}, if the current value is better, then set c_{best} as the $eval(\overrightarrow{x}_i)$, and set $\overrightarrow{p}_{best}$ equal to $\overrightarrow{x}_i$
5. Search the best cost c_{best} and assign it into $\overrightarrow{g}_{best}$
6. Update the velocity and position of the particle with this formulation:

$$\begin{cases} \overrightarrow{v}_i \leftarrow \overrightarrow{v}_i + \overrightarrow{U}(0, \varphi_1) \otimes (\overrightarrow{p}_i - \overrightarrow{x}_i) + \overrightarrow{U}(0, \varphi_2) \otimes (\overrightarrow{g}_{best} - \overrightarrow{x}_i) \\ \overrightarrow{x}_i \leftarrow \overrightarrow{x}_i + \overrightarrow{v}_i \end{cases}$$

- $\overrightarrow{U}(0, \varphi_i)$ represents a vector of random number uniformly distributed in $[0, \varphi_i]$ generated at each iteration
- $\otimes$ is component-wise multiplication

7. If a criterion is met, exit loop
8. End Loop

Note that the global optimum is not guaranteed since this application is ruled by the exploration-exploitation dilemma. A more detailed description of the main features, drawbacks, and constraints of the PSO algorithm is illustrated in [24][25].

3.1 Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

CMA-ES is a stochastic method for optimization of non-linear, non-convex, continuous domain functions. It belongs to the class of evolutionary algorithms that are based on biological evolution from which the idea is to try to mimic this process by implementing an iterative procedure of selection, mutation, and recombination of the population of candidate solutions and its fitness represents the value of the used cost function [26]. The selection process is made by comparing all the individuals' values, and the most suitable is used to develop a new generation that is a recombined and mutated version of the best individual of the previous generation, through the usage of Gaussian noise. CMA-ES is characterized by its generation of the population of the individuals, which is achieved by sampling a multivariable Gaussian probability distribution described as:

$$x \sim \mathcal{N}(m, C) \tag{3.1}$$

Where the modal value corresponds to the distribution mean, $m \in R^n$ and its covariance matrix $C \in R^{nxn}$, which is symmetric and positive definite that can be interpreted as an ellipsoid shown in Fig 3.2. As the figure shows, the principal axes correspond to the eigenvector of C, while the squared axes lengths correspond to its eigenvalues, the surface of the ellipsoid is depicted as the density of the distribution. Thus, the covariance matrix C can be Eigen-decomposed as:

$$C = B\left(D^2\right)B^T \tag{3.2}$$

From which Equation (3.1) can be written as:

$$\mathcal{N}(m, C) \sim m + BD\mathcal{N}(0, I) \tag{3.3}$$

Where D scales the spherical distribution while B defines the orientation of the ellipsoid.

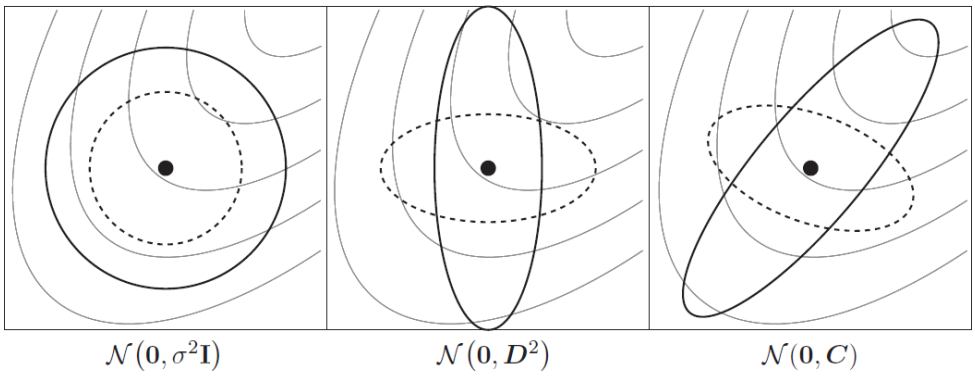


Figure 3.2: Ellipsoid decomposition for normal distribution

If this is so, the sampled search points for individuals of each generation g can be determined as:

$$x_k^{g+1}\ m^{(g)} + \sigma^{(g)}\mathcal{N}\left(0, C^{(g)}\right)\ m^{(g)} + \sigma^{(g)} BDZ_k for k = 1, \ldots, \lambda \tag{3.4}$$

Where σ is the step-size, λ is the population size, and Z_k is a vector of random numbers.

After the evaluation of the candidate solutions on each iteration, the parameters of Equation (3.4) are updated based on the fitness of the individuals. The new mean $m^{(g+1)}$ is calculated as a weighted average ω of the best μ individuals also called parents and its formulation is given by:

$$m^{(g+1)} = \sum_{i=1}^{\mu} \omega_i x_{i:\lambda}^{(g+1)} \tag{3.5}$$

As for the update of the covariance matrix $C^{(g+1)}$ and the step size $\sigma^{(g+1)}$, the reader is referred to [27] since the complexity, motivation, description, and analysis are beyond the scope of this work. However, the mathematical results of each of the variables will be presented to give a complete vision of the method.

$$C^{(g+1)} = (1 - c_1 - c_\mu)\, C^g + c_1 \left(p_c^{(g+1)} p_c^{(g+1)^T}\right) + c_\mu \sum_{i=1}^{\lambda} \omega_i y_{i:\lambda}^{(g+1)} y_{i:\lambda}^{(g+1)^T} \tag{3.6}$$

Equation (3.6) is a combination of different update methodologies that target separate information on each term:

1. Weighted historical realization of the Covariance matrix (previous generation $C^{(g)}$)

2. Efficient information from the entire population ($rank - \mu$ update)

3. Exploitation of the evolution path given by information on the correlation between generations (rank-one update)

As for the update of the step size, an empirical length of the evolution path is used:

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\left\|p_\sigma^{(g+1)}\right\|}{\mathbb{E}\left\|\mathcal{N}(0,I)\right\|} - 1\right)\right) \tag{3.7}$$

The CMA-ES algorithm results to be an advantageous method due to usage of the covariance matrix C, where the dependencies between variables are modeled; moreover, the update methodology provides robust adaptability on the search distribution, as well as the prevention on the degeneration of the population, while the updates on the step size prevent a premature convergence of the population. Furthermore, the CMA-ES is designed to have invariant and unbiased properties that are highly desirable for evolution strategies, since it gives a uniform performance on classes of functions considering variations of the parameters to be tackled by the usage of the multivariable distribution.

To conclude, a general overview of the CMA-ES is shown in the following diagram:

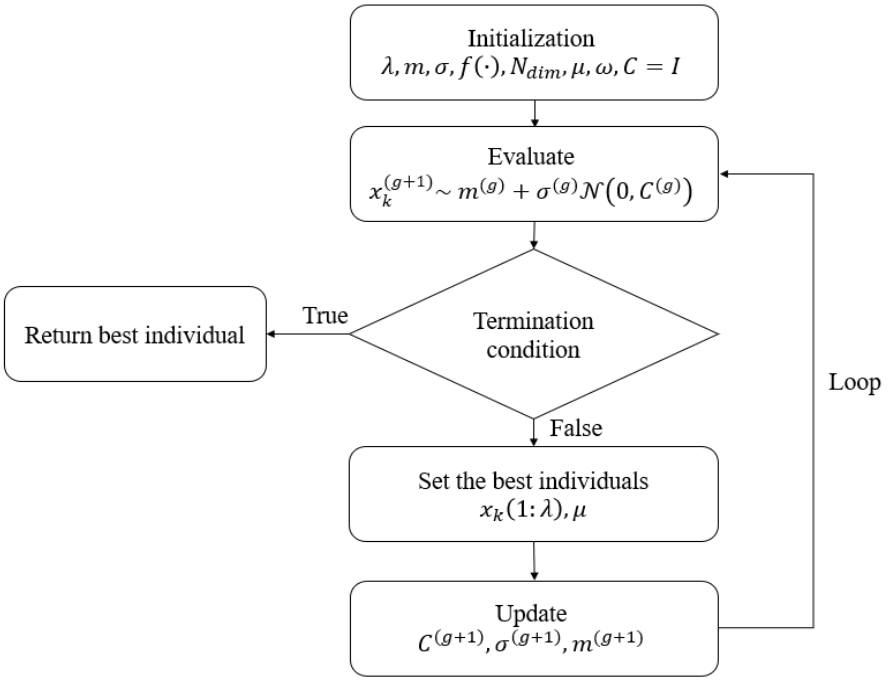


Figure 3.3: CMA-ES Flowchart algorithm [2]

3.2 Model-free VRFT optimization implementation

Due to the setbacks previously described in (VRFT model-reference) and the fast-tuning characteristics provided by the VRFT algorithm, the idea of implementing iterative processes within the method becomes appealing. For instance, it is possible to develop strategies to improve certain aspects regarding the performance and stability of the resulting closed-loop systems targeting the choice of the model reference. To do so, it is necessary to introduce a reference signal $r\left(0\right)\ldots r(N-1)$ that must supply a representative informative dataset of the system dynamics, which in the study case a Pseudo-Random Binary Sequence (PRBS), is used due to several characteristics in terms of implementation, scalability and harmonic content given its binary nature and reproducibility due to its pseudo-randomness. Note that the choice of $r(i)$ depends on the application of interest, $r(t)$ is not restricted to PRBS signals. As in VRFT let the optimization problem be defined as structured with respect to the model reference; where the number of real zeros (n_{zr}), complex conjugate zeros ($n?_{zc}$), real poles (n_{pr}), and complex conjugate poles (n_{pc}) are previously defined, if this is so, let the model reference $M(z)$ depends on a parameter vector θ to be optimized so the model reference becomes:

$$M\left(z\right)=M\left(\theta,z\right)=K\frac{\prod_{l=1}^{n_{zr}}(z-z_l)\prod_{l=n_{zr}+1}^{n_{zr}+n_{zc}}(z-z_l)(z-z_l^*)}{\prod_{l=1}^{n_{pr}}(z-p_l)\prod_{l=n_{pr}+1}^{n_{pr}+n_{pc}}(z-p_l)(z-p_l^*)} \tag{3.8}$$

Where K enforces $M(\theta,1)=1$ and z_l,p_l stands for the l^{th} zero and pole, respectively, whilst the parameter vector θ is defined by:

$$\theta=\begin{bmatrix} z_l & l=1,\ldots,n_{zr} \\ \Re\left\{z_l\right\},\Im\{z_l\} & l=n_{zr}+1,\ldots,n_{zr}+n_{zc} \\ p_l & l=1,\ldots,n_{pr} \\ \Re\left\{p_l\right\},\Im\{p_l\} & l=n_{pr}+1,\ldots,n_{pr}+n_{pc} \end{bmatrix} \tag{3.9}$$

So, the VRFT Equation (2.16) will be equivalent to:

$$\varphi^*\left(\theta\right)=\underset{\varphi}{argmin}\sum_{t=0}^{N-1}\left(u\left(t\right)-C\left(\varphi,z\right)e_v\left(\theta,t\right)\right)^2 \tag{3.10}$$

In addition, the cost function of the parameter vector θ is:

$$J\left(\theta\right)=\frac{1}{N}\sum_{t=0}^{N-1}W_y\left(r\left(t\right)-y_p\left(\theta,t\right)\right)^2+W_{\Delta u}\Delta u_p^2\left(\theta,t\right)+W_{fit}(u\left(t\right)-C\left(\varphi^*(\theta),z\right)e_v\left(\theta,t\right)) \tag{3.11}$$

Where the first two terms account for error tracking and control effort, which are the reflection of the performance of the reference model $M(\theta,z)$ is completely

matched by the closed-loop system. This is described as:

$$y_p(\theta, t) = M(\theta, z)\, r(t) \tag{3.12}$$

$$u_p(\theta, t) = C(\varphi^*(\theta), z)(r(t) - y_p(\theta, t)) \tag{3.13}$$

And

$$\Delta u_p(\theta, t) = u_p(\theta, t) - u_p(\theta, t-1) \tag{3.14}$$

The third term is related to the model matching when $C(\varphi^*(\theta), z)$ is used while $W_y, W_{\Delta u} \wedge W_{fit}$ are nonnegative weights that are left as tuning parameters just like Model Predictive Control (MPC) penalties. Finally, the optimal parameter vector θ^* is selected as:

$$\theta^* = \underset{\theta}{argmin} J(\theta) \tag{3.15}$$

As can be seen, the optimization problem becomes in a nonlinear, nonconvex one (bi-level programming problem) which can be solved using PSO, the reader is referred to [28] for details on motivations and derivation of the algorithm, while a general pseudo-code of the implementation will be depicted below:

Model-free VRFT algorithm

1. Populate the particle swarm $\theta^i, i = 1, \ldots, N$ with random initial values constrained by the Routh-Hurwitz stability criterion for both poles and zeros
2. Loop: k
3. Loop: i
4. Set θ^i into $M(\theta, z)$
5. Fix K such that $M(\theta^i, z) = 1$
6. Compute the controller $C(\varphi^*(\theta^i, z)$ through VRFT
7. Set the cost function $J(\theta^i)$ adding a penalty function for poles and zeros out or in the limits of the Routh-Hurwitz stability criterion
8. End Loop i
9. Set the global best particle position $\overrightarrow{g}_{ibest}$ based on the cost function $J(\theta^i)$
10. Update the velocity and position of the particle using the formulation on the **PSO algorithm**
11. End Loop k

This algorithm provides a controller, which is fully data-driven, since the need for prior knowledge of the plant $P(z)$ is tackled by the model-free approach; it is meaningful to mention that this method does not provide theoretical guarantees of stability of the resulting closed loop. However, further developments of this algorithm show that using the unfalsified control (UC) approach [29], it's possible to guarantee stability as shown in [28], despite the benefits of the guaranteed stability of the resulting closed-loop system. Furthermore, the implementation of the unfalsified control is less appealing for the study case considering that UC is much slower tuning method than VRFT due to the necessity of estimation through Prediction Error Methods (PEM) as well as a-posterior stability tests.

3.2.1 Cascade model-free VRFT optimization implementation

The interpolation of the algorithm described in the previous section can be made using two different approaches. The first one, related to the cascade VRFT controller design, previously described in section 2.3.3 which can be implemented by using the Model-free VRFT algorithm first for the inner controller and after computing the outer input signal shown in the Equation (2.19) another execution of the algorithm can be done to provide the outer controller. However, this approach, which is fully data-driven and Model-free, does not account for cascade loop theory by the fact that the outer loop must be at least one decade slower than the inner loop to see a unitary closed-loop transfer function for the inner one. Even more, this implies that for the resulting cutoff frequency of the model references must be in accordance with the following inequality:

$$\omega_o \leq \frac{\omega_i}{10} \rightarrow \frac{P_i C_2(\theta)}{1 + P_i C_2(\theta)} = 1 \ \textit{seen from the outer loop} \tag{3.16}$$

Equation (3.16) is not compliant with the optimal value that the method is capable to provide, making the resulting closed-loop system to be divergent. To overcome this situation different scenarios had been tested, such as the introduction of another constraint on the cost function on Equation (3.11) to provide information of cascade controller restrictions into the optimization method.

$$W_{cascade} = \begin{cases} 0 & \textit{if } \omega_o(\theta) \leq \frac{\omega_i}{10} \\ inf & \textit{if } \omega_o(\theta) > \frac{\omega_i}{10} \end{cases} \tag{3.17}$$

Despite this, the condition does not provide any noticeable enhancement on the resulting closed-loop and the only scenario from which the algorithm was able to provide a convergent response was to impose a simulated output dataset using a transfer function that follows the inequality on Equation (3.16). Nevertheless, this requires prior knowledge of the system and thus goes in contradiction with the model-free framework. The third scenario proposed is focused on the elimination

of one calculation of an optimal model reference; in order to do so, the proposed methodology described in 2.3.3 must be redesigned in the following way:

Consider the cascade control system shown in Fig 2.3, taking the same assumptions of the VRFT formulation, so it is considered LTI SISO system in the structured case with the cost function of the inner loop with the form of Equation (36), thus, the function can be rewritten as:

$$J_{VRi}(\varphi) = \frac{1}{N}\sum_{i=1}^{N} \|u_{ini}(i) - C_2(z,\varphi)\,\overline{e}_1(i)\|_2^2 \tag{3.18}$$

$$\overline{e}_1(i) = \overline{r}_1(i) - y_{ini}(i) \tag{3.19}$$

Similarly, the outer loop virtual error and reference are given by:

$$\overline{e}(i) = \overline{r}(i) - y_{ini}(i) \tag{3.20}$$

$$y_{ini}(i) = M_o(\theta, z)\overline{r}(i) \tag{3.21}$$

As shown in Fig 2.3 the reference of the inner loop is also the output of the outer controller:

$$\overline{r}_1(i) = C_1(\varphi, z)\overline{e}(i) \tag{3.22}$$

Substituting into Equation (3.18) the performance index of the cascade control is obtained:

$$J_{VRi}(\varphi) = \frac{1}{N}\sum_{i=1}^{N} \left\| u_{ini}(i) + C_2(\varphi,z)\,y_{1ini}(i) - C_1(\varphi,z)C_2(\varphi,z)\left(\frac{1}{M_o(\theta,z)} - 1\right) y_{ini}(i)\right\|_2^2 \tag{3.23}$$

In addition, the optimal parameter vector φ is a result of:

$$\varphi^* = \underset{\varphi}{argmin} J_{VRi}(\varphi) \tag{3.24}$$

The reader is referred to [30] for the full description and analysis of the derivation of the cascade problem. As can be seen, Equation (3.23) is a non-linear optimization problem that can be solved solidly using CMA-ES. This scenario provides a solution for the model-free scope due to its one-time nature to provide both inner and outer controllers just using one optimized model reference.

To sum up, the complete Cascade model-free VRFT can be summarized by presenting a pseudo-code:

Cascade model-free VRFT algorithm

1. Populate the particle swarm $\theta^i, i = 1, \ldots, N$ with random initial values constrained by the Routh-Hurwitz stability criterion for both poles and zeros
2. Execute a one-shot experiment to get a dataset of the form $\{u_{ini}, y_{1ini}, y_{ini}, \varphi_{ini}\}$

3. Loop: k
4. Loop: i
5. Set θ^i into $M_o(\theta, z)$
6. Fix K such that $M_o(\theta^i, z) = 1$
7. Calculate the virtual reference signal $\overline{r}(i) = \frac{1}{M_o(\theta^i, z)} y(\varphi_{ini})$
8. Construct $J_{VRi}(\varphi)$ as Eq.(3.23)
9. Compute the controller $C(\varphi^*(\theta^i, z)$ through **CMA-ES**
10. Set the cost function $J(\theta^i)$ adding a penalty function for poles and zeros out or in the limits of the Routh-Hurwitz stability criterion
11. End Loop i
12. Set the global best particle position $\overrightarrow{g}_{ibest}$ based on the cost function $J(\theta^i)$
13. Update the velocity and position of the particle using the formulation on the **PSO algorithm**
14. End Loop k

Note that the performance of the PSO algorithm is highly influenced by the choice of the weights in Equation (3.11), as well as the reference signal $r(i)$, which, if possible, should be adapted from the dynamics of the outer loop.

Chapter 4

Simulation

This chapter is dedicated to the execution and validation of the previously described algorithms. The validation is achieved by using different well-known benchmarks that will cover most of the scenarios from which the algorithms will be tested within the UAVs framework. Three different benchmarks will be carried out providing information about the performance, decoupling, and MIMO extension.

4.1 Numerical simulation

This benchmark has been built to show the incidence of coupling effects with respect to the controller synthesis and it is set to guarantee perfect tracking with an adequate PI controller. The simulation has been used in [17] and [31] showing the performance of different VRFT realizations.

Experiment setup

A discrete system of first order transfer functions with a sampling rate of 1Hz is introduced:

$$G(z) = \begin{bmatrix} \frac{0.09516}{z-0.9048} & \frac{0.03807}{z-0.9048} \\ \frac{-0.02974}{z-0.9048} & \frac{0.04758}{z-0.9048} \end{bmatrix} \tag{4.1}$$

With a reference model $M(z)$ of the form:

$$M(z) = \begin{bmatrix} \frac{0.1}{z-0.9} & 0 \\ 0 & \frac{0.1}{z-0.9} \end{bmatrix} \tag{4.2}$$

And its respective controller using standard VRFT method is given by:

SISO	MIMO
$\begin{bmatrix} \frac{1.082z-1.023}{z-1} & 0 \\ 0 & \frac{2.223z-2.057}{z-1} \end{bmatrix}$	$\begin{bmatrix} \frac{0.08406z-0.7606}{z-1} & \frac{0.5254z-0.4754}{z-1} \\ \frac{-0.6726z+0.6086}{z-1} & \frac{1.681z-1.521}{z-1} \end{bmatrix}$

Table 4.1: VRFT Controller parameters

4.1.1 Results

The experiment was tested implementing different configurations of the PSO to give a clear overview of the incidence of the coupling effects between SISO and MIMO configurations, and how the optimization is able to provide different responses according to the enforced performance action through the usage of the weights described in Equation (3.11).

SISO Open loop

As can be seen in Fig 4.1, the decoupling in the SISO case is dealt as a disturbance rejection property of the controller. Moreover, note that the performance of the closed-loop system is better in terms of settling time and disturbance rejection, as the enforcing weight w_y is increased due to the resulting optimal model reference detailed in Table 4.2. However, the algorithm is not capable to satisfy the tracking requirements.

W_y	$M^*(\theta, z)$	$C^*(\varphi, z)$
30	$\begin{bmatrix} \frac{0.08224}{z-0.9178} & 0 \\ 0 & \frac{0.08224}{z-0.9178} \end{bmatrix}$	$\begin{bmatrix} \frac{0.8592z-0.777}{z-1} & 0 \\ 0 & \frac{1.767z-1.601}{z-1} \end{bmatrix}$
100	$\begin{bmatrix} \frac{0.1252}{z-0.8748} & 0 \\ 0 & \frac{0.1252}{z-0.8748} \end{bmatrix}$	$\begin{bmatrix} \frac{1.12z-1.186}{z-1} & 0 \\ 0 & \frac{2.688z-2.437}{z-1} \end{bmatrix}$

Table 4.2: Numerical SISO Mode-Free VRFT Parameters

MIMO Closed loop with additive noise

A MIMO regulation approach for these specific case results in a significant improvement, related to the tracking requirements since the off-diagonal terms of the synthesized controller are now used as can be seen in Table 4.3. This means that the coupling effect is mitigated from this approach, and also the PSO algorithm is able to provide both a controller which follows the model reference described in the setup of the experiment, and an enhanced controller that provides better tracking response to the step reference as can be seen in Fig 4.2, this is due to the optimal choice of the model reference.

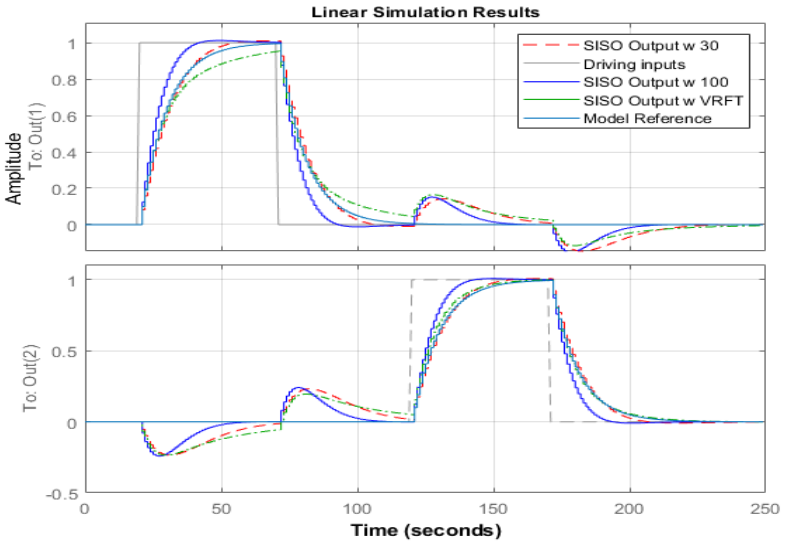


Figure 4.1: Numerical SISO step response

W_y	$M^*(\theta, z)$	$C^*(\varphi, z)$
30	$\begin{bmatrix} \frac{0.09884}{z-0.9012} & 0 \\ 0 & \frac{0.09884}{z-0.9012} \end{bmatrix}$	$\begin{bmatrix} \frac{0.8232z-0.7445}{z-1} & \frac{0.523z-0.4733}{z-1} \\ \frac{-0.6846z+0.6214}{z-1} & \frac{1.678z-1.517}{z-1} \end{bmatrix}$
100	$\begin{bmatrix} \frac{0.1474}{z-0.8526} & 0 \\ 0 & \frac{0.1474}{z-0.8526} \end{bmatrix}$	$\begin{bmatrix} \frac{1.237z-1.12}{z-1} & \frac{0.4502z-0.7049}{z-1} \\ \frac{-0.9921z+0.898}{z-1} & \frac{2.474z-2.235}{z-1} \end{bmatrix}$

Table 4.3: Numerical MIMO Mode-Free VRFT Parameters

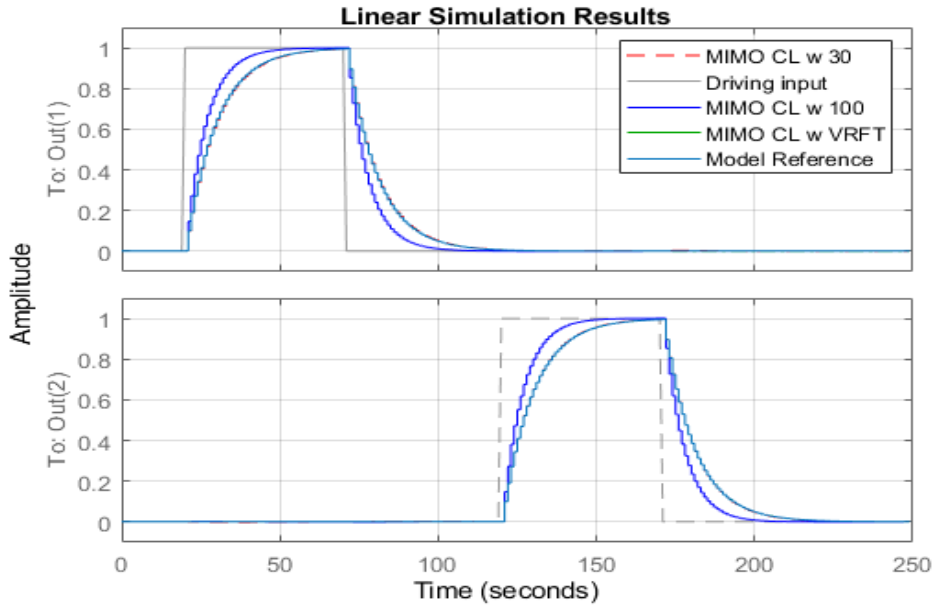


Figure 4.2: Numerical MIMO step response

4.2 LVL100 gas turbine

This experiment is a well-known MIMO benchmark [32], which is a Gas Turbine Engine that is represented by a linear continuous-time state-space model with a two-input, two-output, five states, minimum phase system. To work in the VRFT framework, the model is discretized using Tustin's approximation with a sampling period of 0.1 seconds, and the main objective of the experiment is to tune a multivariable PI controller as has been done with several data-driven methods, described in [17], [31], [33], [34], [33]. Additionally, 200 Monte-Carlo simulations are performed to denote the variance of the closed-loop system and the robustness of the algorithms.

4.2.1 Experiment setup

The goal of the experiment is to control the spool speed and its temperature with respect to the fuel flow and the area of the turbine nozzle. The data generation is done using a closed-loop experiment where an initial stabilizing controller defined in Equation (4.3) has been used.

$$C_0(z) = \begin{bmatrix} \frac{z-0.99}{z-1} & \frac{0.1z-0.099}{z-1} \\ \frac{-z+0.99}{z-1} & \frac{z-0.99}{z-1} \end{bmatrix} \tag{4.3}$$

The system implemented is given by:

$$\dot{x} = Ax + Bu \qquad y = Cx + Du$$

$$A = \begin{bmatrix} -1.4 & -0.055 & 0 & 43.0 & 6.3 \\ 0.093 & -0.11 & 0 & 4.2 & -0.76 \\ -7.8 & -0.26 & -3.3 & 300.0 & -4.5 \\ 0 & 0 & 0 & -25.0 & 0 \\ 0 & 0 & 0 & 0 & -33.0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.4}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The input provided is as previously discussed PRBS signal with a frequency bandwidth spanning from 0 to 30 rad/s for each channel of the system as shown in Fig 4.3.

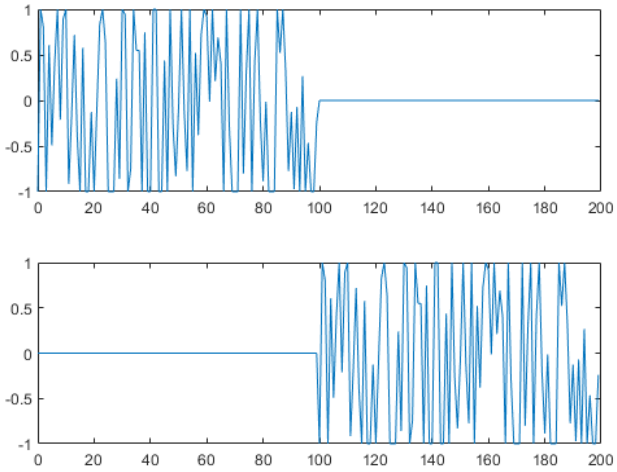


Figure 4.3: PBRS signal

Finally, the proper model reference has been chosen, its design considers decoupling effects that in the case of study are a relevant characteristic to deal with since those effects can lead the turbine to malfunction, which in terms of safeness is not desirable.

$$M(z) = \begin{bmatrix} \frac{0.4}{z-0.6} & 0 \\ 0 & \frac{0.4}{z-0.6} \end{bmatrix} \tag{4.5}$$

4.2.2 Results

Once again, the experiment was separated into different cases depending on the used data-acquisition method. The first and second experiments are used to ex-

plore the behavior of the designed closed-loop when the dataset is affected by noise, while the third case focuses on the robustness of the algorithm with respect to the variability on the noise realizations such that the variance of the closed-loop systems becomes visible.

MIMO Open loop

In this example, it can be seen in Fig 4.4 that the choice of the weight does not provide any further enhancement of what has been achieved using nominal VRFT as it was shown in Fig 4.2. Nevertheless, it can be seen that the resulting optimal model reference functions described in Table 4.4 reside in the neighborhood of the proposed model reference.

W_y	$M^*(\theta, z)$	$C^8(\varphi, z)$
3000	$\begin{bmatrix} \frac{0.2257}{z-0.7743} & 0 \\ 0 & \frac{0.2257}{z-0.7743} \end{bmatrix}$	$\begin{bmatrix} \frac{0.1893z-0.03358}{z-1} & \frac{11.19z-10.85}{z-1} \\ \frac{0.1998z-0.1428}{z-1} & \frac{-1.834z+1.302}{z-1} \end{bmatrix}$
10000	$\begin{bmatrix} \frac{0.3426}{z-0.6574} & 0 \\ 0 & \frac{0.3426}{z-0.6574} \end{bmatrix}$	$\begin{bmatrix} \frac{0.2993z-0.06062}{z-1} & \frac{17.49z-16.96}{z-1} \\ \frac{0.3081z-0.2207}{z-1} & \frac{-2.868z+2.036}{z-1} \end{bmatrix}$

Table 4.4: LVL100 MIMO Open Loop Step Parameters

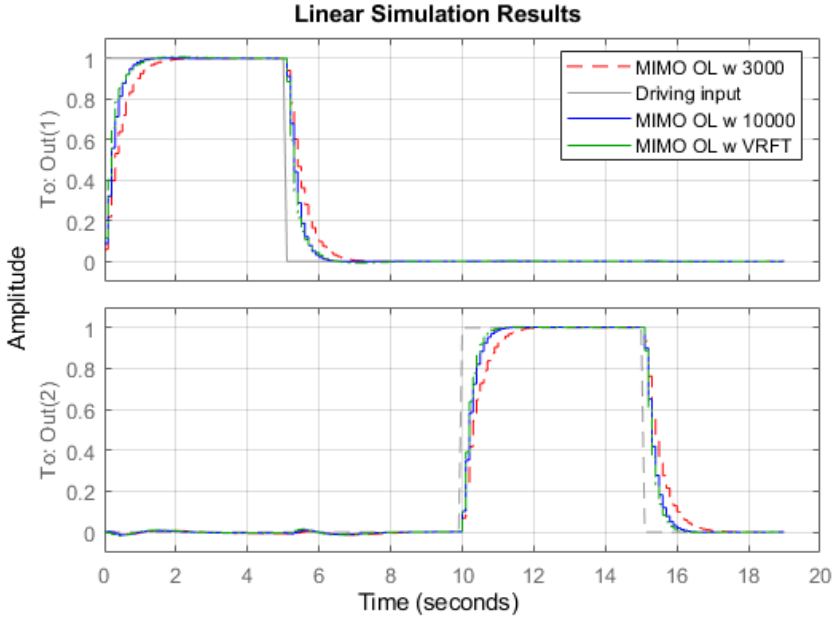


Figure 4.4: LVL100 MIMO Open Loop Step Response

MIMO Closed loop with additive noise

The introduction of additive noise into the dataset generation reduces the performance of the data-driven control techniques, as can be seen in Fig 4.5, due to the presence of coupled effects that the MIMO-designed controller is not able to effectively counteract as it was done in the second channel of Fig 4.4.

However, the Model-Free approach is able to get slightly better tracking capabilities than the initial configuration using VRFT, meaning that the optimization method provides an advantage in noisy conditions. This condition is reflected in the remoteness of the models in Table 4.5 with respect to the initial model on Equation (4.5). On the contrary, the computational effort required to execute the PSO algorithm due to its iterative nature is not comparable with the nominal VRFT.

W_y	$M^*(\theta, z)$	$C^*(\varphi, z)$
3000	$\begin{bmatrix} \frac{0.5154}{z-0.4846} & 0 \\ 0 & \frac{0.5154}{z-0.4846} \end{bmatrix}$	$\begin{bmatrix} \frac{0.4802z-0.1019}{z-1} & \frac{25.79z-24.98}{z-1} \\ \frac{0.5412z-0.3973}{z-1} & \frac{-4.105z+2.89}{z-1} \end{bmatrix}$
10000	$\begin{bmatrix} \frac{0.6758}{z-0.3242} & 0 \\ 0 & \frac{0.6758}{z-0.3242} \end{bmatrix}$	$\begin{bmatrix} \frac{0.6739z-0.1456}{z-1} & \frac{34.66z-33.56}{z-1} \\ \frac{0.7622z-0.5633}{z-1} & \frac{-5.517z+3.893}{z-1} \end{bmatrix}$

Table 4.5: LVL100 MIMO Closed Loop Mode-Free VRFT Parameters

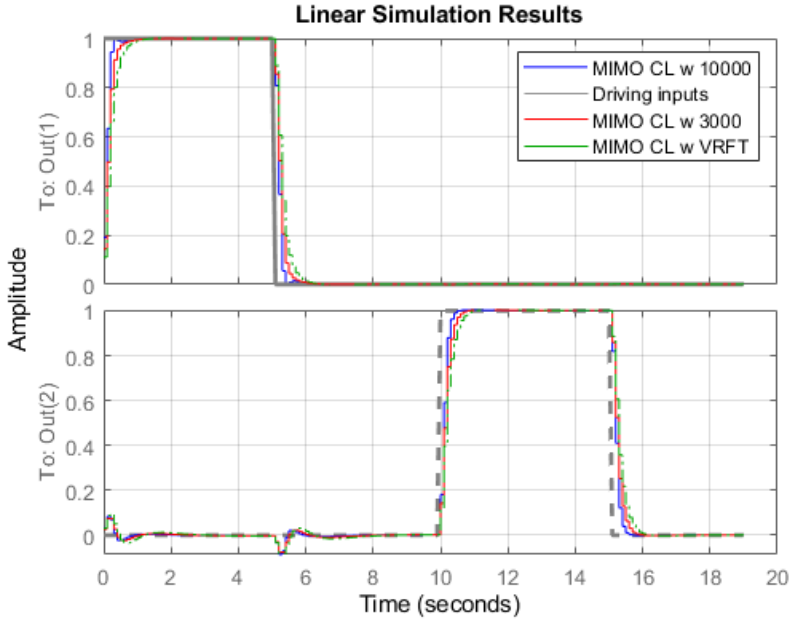


Figure 4.5: LVL100 MIMO Closed Loop Step Response

Monte Carlo Simulation

A statistical analysis has been carried out to determine the robustness of the algorithm for different noise realizations using a dataset of 1000 data points while the other input conditions are maintained, and the same step reference is used to simulate 200 cycles of control design and its respective response. Table 4.6 shows the mean values, standard deviation, and the ratio between them. Moreover, it can be seen that the variance of the Model-Free PSO method is higher than the VRFT for each of the components of the resulting controller. This behavior is due to the performance of the algorithm, the number of data-points simulated motivated by the fact that the results on VRFT are related to its convergence which is given when $N \to \infty$, and finally, the optimization of each iteration gives an optimal model-reference causing the resulting controller to have more variability when compared between them.

		μ_{PSO}	σ_{PSO}	σ/μ_{PSO}	μ_{VRFT}	σ_{VRFT}	σ/μ_{VRFT}
K_p	(1,1)	0.0197425	0.220617	11.1747	0.074188	0.09133	1.2311
	(2,1)	11.3402	1.2051	0.106268	15.8591	1.10127	0.0694412
	(1,2)	0.17882	0.1482	0.82876	0.29936	0.018815	0.062849
	(2,2)	-1.2907	0.22865	-0.17715	-1.7365	0.16102	-0.0092724
K_i	(1,1)	1.7759	1.2597	0.70935	2.8343	0.16206	0.057177
	(2,1)	3.8349	1.7725	0.46219	6.2614	0.77747	0.12417
	(1,2)	0.66797	0.33379	0.4997	1.1262	0.056882	0.050506
	(2,2)	-5.751	1.1936	-0.20754	-8.8524	0.4944	-0.055849

Table 4.6: Statistical analysis of the controllers using PSO & VRFT

Fig 4.6 and Fig 4.7 show the results to a step response highlighting the mean value of the whole execution. As can be seen, the change of the model reference through the iteration achieves a smoother response, particularly on the first channel, where not only the variability is diminished but also the decoupling is more effective than the VRFT due to decreased tracking capabilities that can be related to the sensitivity of the parameter $K_p(1,1)$, where exists sign variations during the experiment. As for the second channel, the peak values of the coupling effect are reduced but the uncertainty is higher.

4.3 ANT-R Simulation

Complementing the previous benchmarks, a final experiment has been developed to test the algorithm simulating a multirotor platform. This simulator is created from real flying data using predictor-based subspace identification (PBSID) techniques to provide an identified model of the quadrotor. The dataset used is simulated using the same controller conditions $C_0(z)$ and the identified plant model $\widehat{P}$.

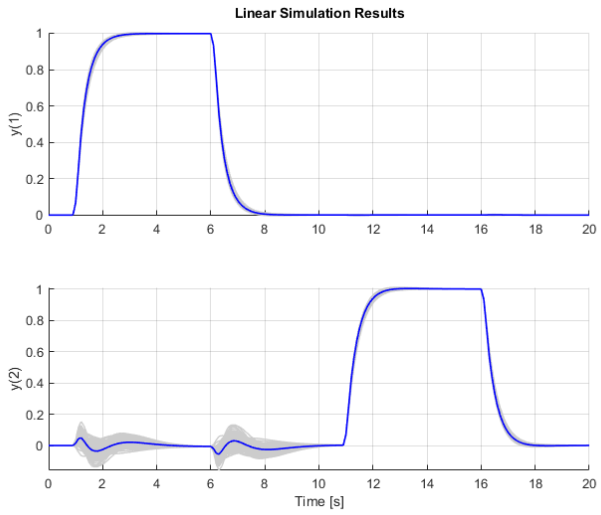


Figure 4.6: 200 Monte-Carlo simulations using PSO

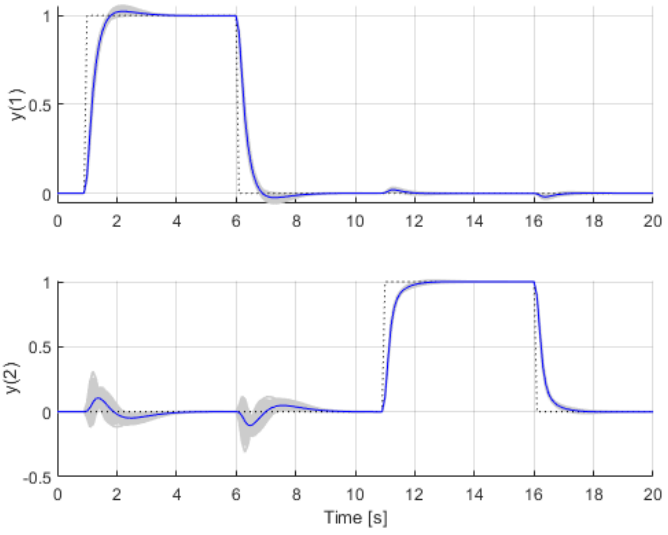


Figure 4.7: 200 Monte-Carlo simulations using VRFT

Figure 4.8: ANT-R quadrotor

The simulation is then settled to test the weighting values on Equation (3.11) and synthesize a SISO cascade P/PI-D controller for the pitch dynamics.

4.3.1 Experiment setup

For the first experiment, a set of increasing values is proposed as follows:

$$\begin{bmatrix} Wy & W_{\Delta u} & W_{fit} \\ 1 & 1 & 1 \\ 10 & 10 & 10 \\ 100 & 100 & 100 \\ 1000 & 1000 & 1000 \\ 10000 & 10000 & 10000 \end{bmatrix} \tag{4.6}$$

The purpose is to test every possible combination of the arrange into the Model-Free PSO algorithm, using as a base the inner control loop of the quadrotor to synthesize a PI-D controller and evaluate the performance of the response depending on the value of each weight.

The second experiment is based on the following cascade control loop:

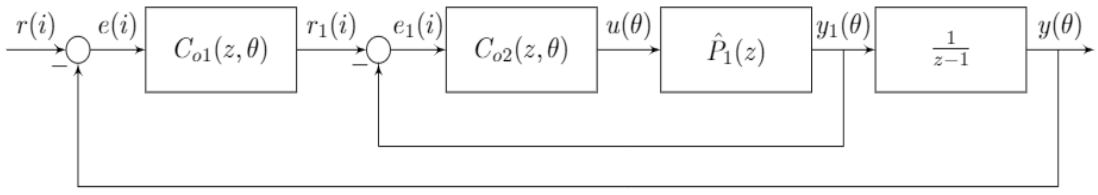


Figure 4.9: Block diagram of the simulation

In addition, the dataset is collected exciting the input with a PRBS signal,

while the output data is then collected from the identified plant model $\widehat{P}_1$ and the output of the known integrator that closes the outer loop.

4.3.2 Results

Sensitivity analysis of the PSO cost function (3.11)

The weighting factors are used to impose a relative performance goal (Tracking Error, Control Effort, Model Matching) on the resulting closed-loop system. As in MPC control design, the weights of the cost function must be tuned according to the specifications of the problem. In this case, inaccurate tuning of the weights will lead to divergent unstable closed-loop systems, and thus, the exploration of a vast spectrum of incidence in the synthesis of the controller provides a metric of usability for the algorithm in the UAV context.

Fig 4.10 exhibits on the left the stable responses of the closed-loop system, while on the right, the selection of the best performing controllers. Note that Fig 4.10 does not have more than the 30% of systems evaluated, the more complex is the problem, narrower is the spectrum of the weighting factors. In counterpart, the arrange of (3.18) shows the weights of the right part of Fig 4.10 where it can be highlighted a narrower spectrum of variables.

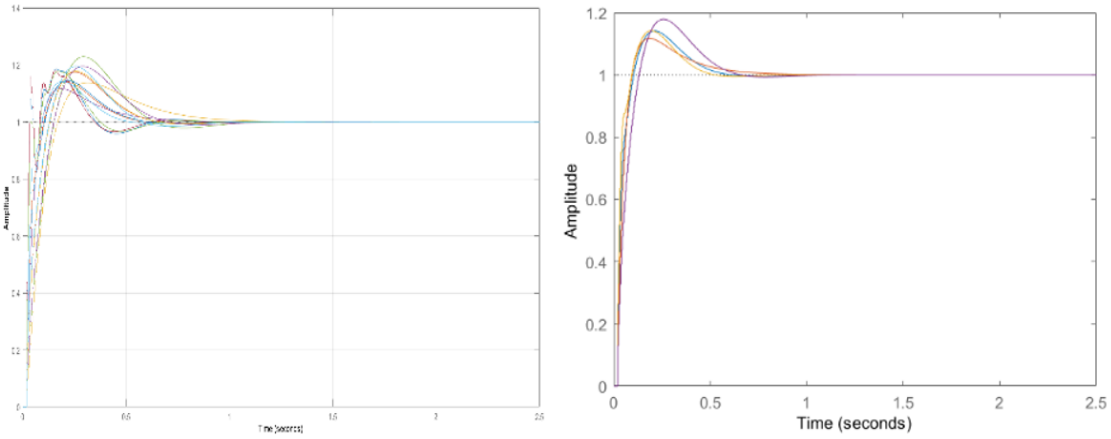


Figure 4.10: Step responses of the weights combinatorial

$$\begin{bmatrix} Wy & W_{\Delta u} & W_{fit} \\ 1 & 1000 & 1 \\ 1 & 10 & 10 \\ 10 & 10000 & 1 \\ 10 & 1000 & 1000 \end{bmatrix} \tag{4.7}$$

Finally, the performance under highly demanding input of a controller synthesized using the weights on (4.8) is shown in Fig 4.11. From the sensitivity analysis, it can

be highlighted that the order of Wu should be at least three orders of magnitude higher with respect to the other tuning variables.

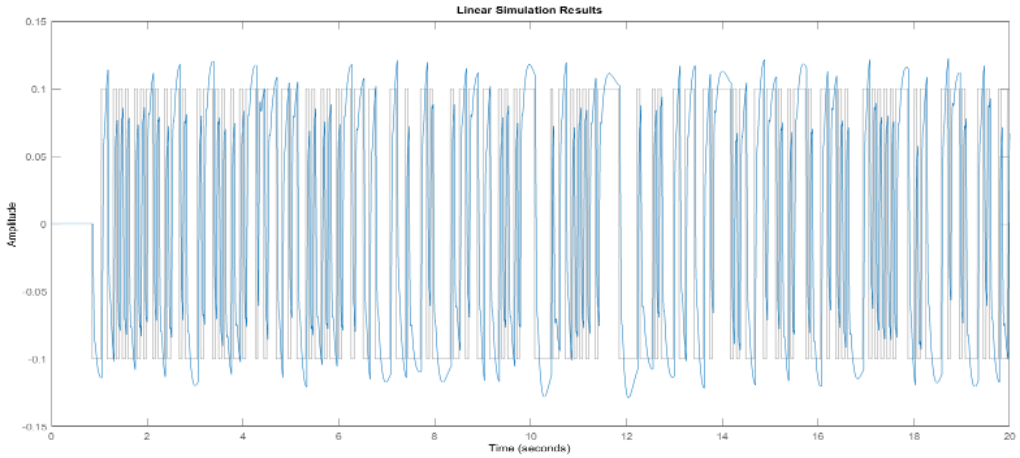


Figure 4.11: PRBS Input Response

$$\begin{bmatrix} Wy & W_{\Delta u} & W_{fit} \\ 1 & 10000 & 10 \end{bmatrix} \tag{4.8}$$

ANT-R Simulation

According to the analysis previously done, the Model-free PSO algorithm is then used without the additional optimization method, since the experiment setup takes part in the convergent scenario previously described in 3.2.1. Furthermore, the approach from the VRFT cascade controller shown in 2.3.3 is used with the PSO algorithm.

As can be seen in Fig 4.12, the method provides an optimal model reference with a settling time of 0.3080s and a stabilizing controller for the inner loop collected in Table 4.7. It is clearly shown that the resulting inner control loop does not follow completely the model reference as other performance factors are being optimized.

$M_i^*(\theta, z)$	$C^*(\varphi, z)$
$\frac{0.002701z+0.00268}{z^2-1.853z+0.8587}$	$Kp = 0.0132$ $K_i = 0.0082$ $K_d = 2.56 \cdot 10^{-6}$

Table 4.7: Model-free PSO inner SISO Parameters

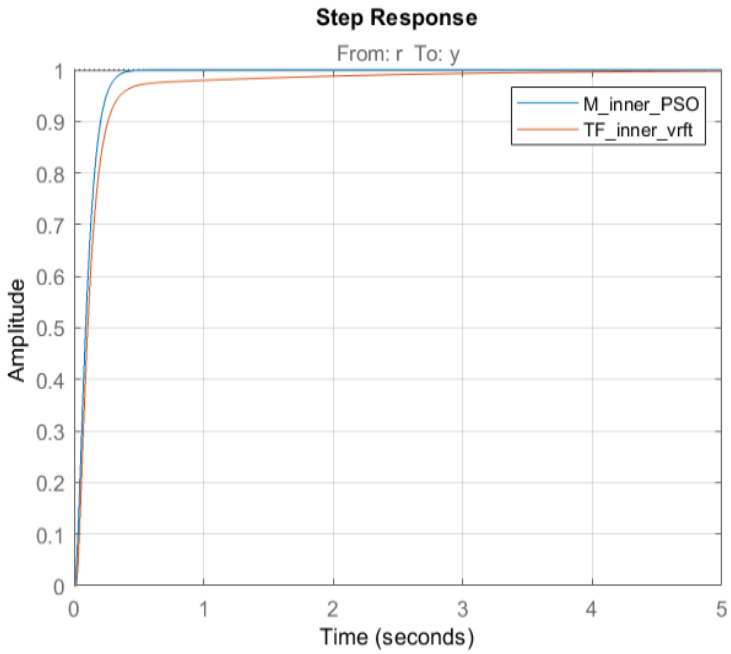


Figure 4.12: Inner loop closed-loop step response

Moreover, after obtaining the optimal inner controller, the input dataset of the outer loop is computed, and then, together with the simulated output response, the same algorithm is applied to give the following outcome:

$M_o^*(\theta, z)$	$C^*(\varphi, z)$
$\frac{1.008z+0.007652}{z^2+0.01516z+5.74\cdot10^{-5}}$	$Kp = 0.005679$

Table 4.8: Model-free PSO outer SISO Parameters

In this case, Fig 4.13 shows that the tracking error is significantly higher than the inner loop, while the outer model reference described in Table 4.8 has a settling time of 1.7040. However, the result is generated from simulated outer loop plant which means that further enhancements on the optimization algorithm had to be done.

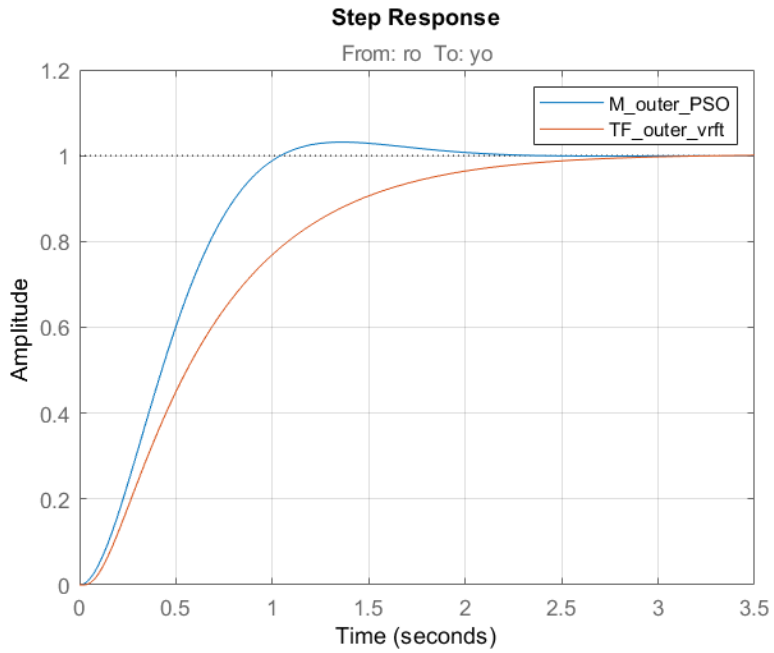


Figure 4.13: Outer loop closed-loop step response

Chapter 5

Drone platforms

To create a basic background for the development of a new control system, it is important to discuss the platforms used for validating the different iterations of the code developed. The platforms, their construction, control strategies and how they have evolved and modified to meet the objectives of this thesis will be presented.

5.1 ANT-X

Figure 5.1: ANT-X quadrotor

5.1.1 Hardware

The hardware of the platform can be summarized as:

The Flight control unit (FCU) is the Pixhawk Mini FCU. The configuration for this FCU Features a modular circuit board that contains the controllers and autopilot control.

The autopilot itself contains the inertial measurement unit (IMU); this device integrates Micro-Electro-Mechanical Systems (MEMS) that act as accelerometers and gyroscopes. The data gathered by the IMU is passed through a Kalman Filter (KF) to account for the integration over time error of the attitude derived from angular rates, the rate is also corrected by the information received from the external sensors (GPS and magnetometer). This allows the autopilot to correctly run the attitude and position control laws.

A remote controller or an Add-on Telemetry module is used to send the controller setpoint, while the output is transformed into a Pulse-Width Modulation (PWM) signal sent to the Electronic Speed controllers (ESC).

The Electronic Speed controllers work with a Brushless DC motor (BLDC), as it cannot be controlled with a direct DC constant source, the BLDC receives a PWM signal that allows more precise control of the speed of the rotors. As the BLDCs are Synchronous, the ESC works as a three-phase inverter receiving a constant DC Voltage from the Power Distribution module and the PWM control signal from the FCU.

5.2 Control strategy

As has been discussed before, the control strategy for the Pitch, roll, and yaw axes uses cascaded PID loops, this control architecture work on the platform and runs at 250Hz. The base architecture of the drone is characterized by a MIMO fed program, where the input vector contains the information of the roll and pitch axis as can be seen on the block diagram of Fig 1.4.

On this diagram, each block is defined by a transfer function characterized by a 2x2 controller matrix. The controller Matrix contains the controller parameters for each axis where the diagonal terms control the action over the same axis being measured. For instance, roll rate error leading to a roll moment. While the $ij(i = j)$ components of the matrix control the action on the other channel, that is, that for a given non-diagonal component that has input on i for an error on component j and therefore generating a decoupling action.

5.3 Testing facility

The Fly-ART laboratory was used as the testing ground for the drone. The laboratory has an indoor flying test cage of 12m x 6m x 3m that allows the experimentations to be done without any potential danger to the drones or testing personnel.

However, by performing an indoor test, the GPS accuracy is diminished, and therefore a different external tracking source is needed to correct the state estimate of attitude, position, and velocity. A 3D Motion capture system (MOCAP) is used to feed the missing GPS data to the FCU. The OptiTrack MOCAP with aid of reflective markers was used to send the information to a NanoPiboard connected to the FCU.

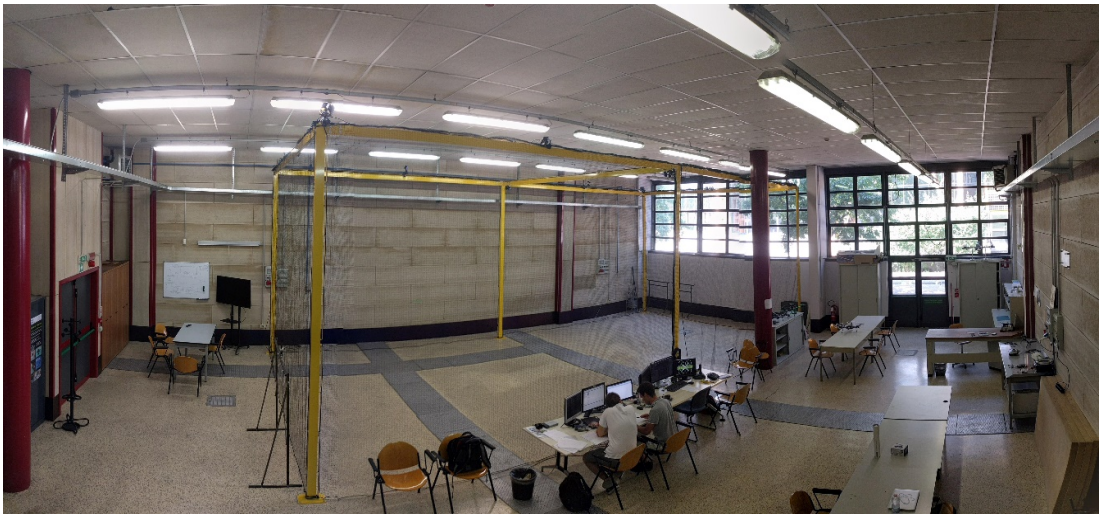


Figure 5.2: Fly-ART laboratory

5.4 Software and firmware

The FCU uses a C++ code derived from a Simulink model that feds the attitude control law function. The C++ is then compiled into custom firmware to be loaded into the FCU. The compiler doesn´t need interaction as it only transfers the code from one language to the other (Simulink onto C++) and onto the FCU hardware (ARM Cortex).

The Optitrack system with its reflective marker aid required additional software, *Motive*, the software computes the current velocity, position, and attitude. *Motive* uses the Robot Operating System (ROS) Libraries to interpolate and process the data for different robotic platforms.

The receiving NanoPi Board connected to the FCU is a MAVROS node contained on the ROS library. MAVROS allows the interaction between the ROS and MAVLink, therefore, creating a communication path between ROS and the FCU. Moreover, this communication path can be used to send different setpoints or inputs from a control station. A MATLAB script run on the ground control station can be transferred to the ROS and therefore to FCU.

Chapter 6

Experimental testing and results

The following chapter will present the experimental results related to the implementation of the Cascade model-free VRFT algorithm into the ANT-X drone platform using the in-flight experimental data collected in the laboratory facilities.

6.1 ANT-X

The ANT-X platform testing is divided into two different viewpoints. First, a constraint local approach is taken to assess the convergence features of the algorithm, once the first test is validated, the second approach will be evaluated using the algorithm to synthesize a stabilizing controller from a global point of view.

6.1.1 Experimental setup

To use the Model-free cascade VRFT algorithm with the ANT-X platform it is necessary to modify Equation (3.18) to account for the PI-D controller configuration previously shown in Fig 1.3, and also, recalling Fig 2.3, thus, let the inner controller be divided into:

$$C_2(\theta, z) = C_{2PI}(\varphi, z) \wedge C_{2D}(\varphi, z) \tag{6.1}$$

So, the Equation (3.18) becomes:

$$J_V(\theta) = \|u_{ini} - (C_2(\varphi, z)\,\overline{e}_1 + C_{2D}(\varphi, z) y_{1ini})\| \tag{6.2}$$

From which:

$$e_1 = \overline{r}_1 - y_{1ini} = C_1(\varphi, z)\, e - y_{1ini} = C_1(\varphi, z)\,(\overline{r} - y_{ini}) - y_{1ini} \tag{6.3}$$

$$y_{ini} = M(\theta(\varphi))\overline{r} \tag{6.4}$$

$$e_1 = C_1(\varphi, z)\left(\frac{1}{M(\theta(\varphi))} - 1\right) y_{ini} - y_{1ini} \tag{6.5}$$

With the resulting cost function of the form:

$$J_V(\theta) = \left\| u_{ini} - (C_{2PI}(\theta, z)\, C_1(\theta, z)\left(\tfrac{1}{M(\theta(\varphi), z)} - 1\right) y_{ini} - (C?_{2PI}(\theta, z) + C_{2D}(\theta, z)) y_{1ini}) \right\| \tag{6.6}$$

Given the proper considerations, the dataset is gathered from the drone platform and processed as shown in Fig 6.1.

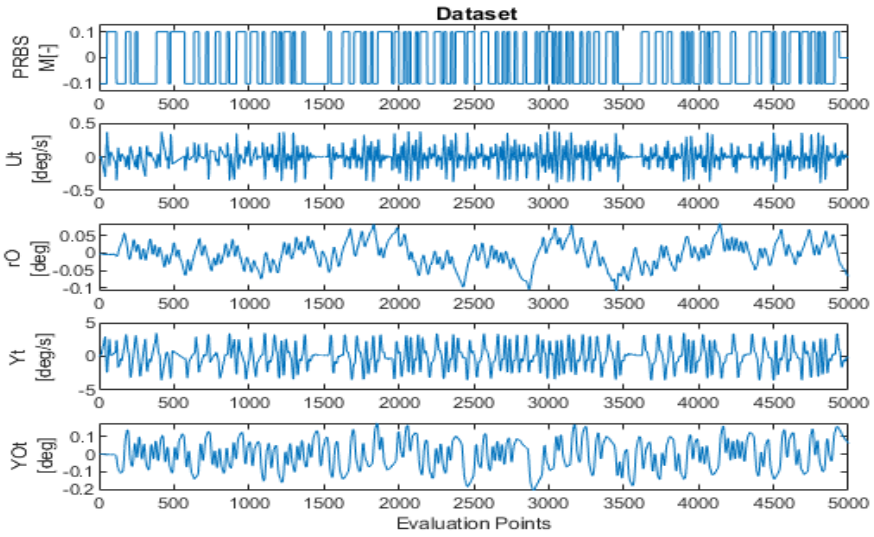


Figure 6.1: Experimental dataset

Where $PRBS$ is the excitation signal used for the PSO algorithm, U_t is the input signal of the inner controller, r_o is the reference signal, Y_t is the output of the 'inner plant' P_1, and finally, Y_{ot} is the output of the closed-loop.

Finally, the dataset was collected using the following control parameters:

θ_{ini}	K_{pi}	K_{ii}	K_{di}	K_{po}
	0.05	0.05	0.001	6.5

Table 6.1: Dataset initial controller parameters

6.1.2 Results

Local optimization

Testing the algorithm with a constraint approach near a known "minimizer" described as:

$$M_o^*(z) = \frac{0.003066z + 0.002938}{z^2 - 1.874 + 0.8799} \tag{6.7}$$

Fig 6.2 and Fig 6.3 illustrate that the algorithm is able to converge into a stable controller. However, the PSO algorithm is not providing any deep exploration while running and the convergence of the swarm it is accomplished on a few trials.

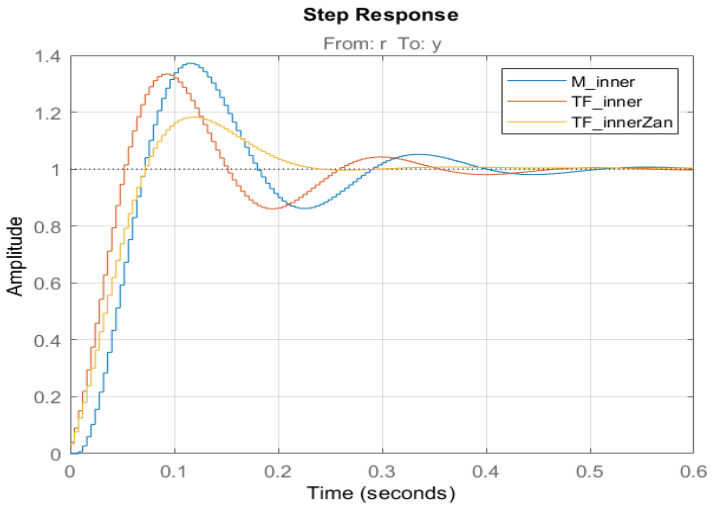


Figure 6.2: Inner Step response of the local test

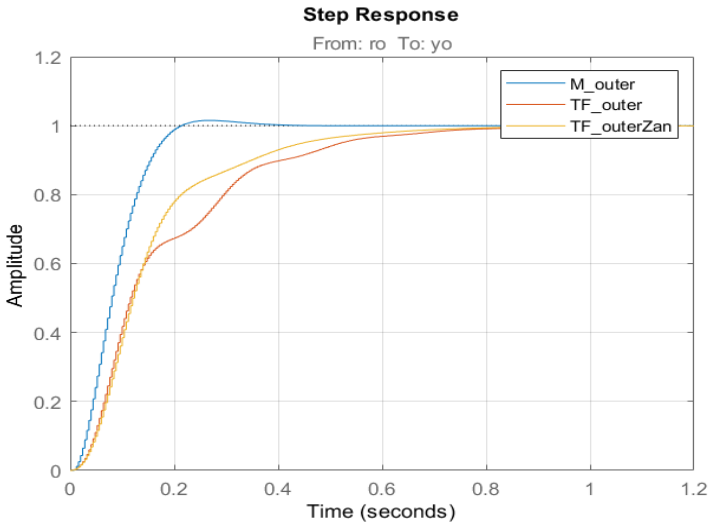


Figure 6.3: Outer Step response of the local test

θ_{ini}	K_{pi}	K_{ii}	K_{di}	K_{po}
VRFT	0.0300	0.2157	0.0004	6.1029
Cascade-PSO	0.0298	0.2671	0.00011	5.452

Table 6.2: Controller parameters of the local approach

For the outer loop test, each weight of the optimization function must be tuned to get a better result; however, the goal of the experiment is to evaluate the approximation of the algorithm into the known controller parameters shown in Table 6.2.

Global approach

The CMA-ES algorithm is meant to deal with the optimization of the controller parameters from a nonlinear formulation of the inner VRFT cost function, as shown in Equation (3.23). From Fig 6.4 it can be seen that the response is better than the proposed known minimizer shown in Equation (6.7).

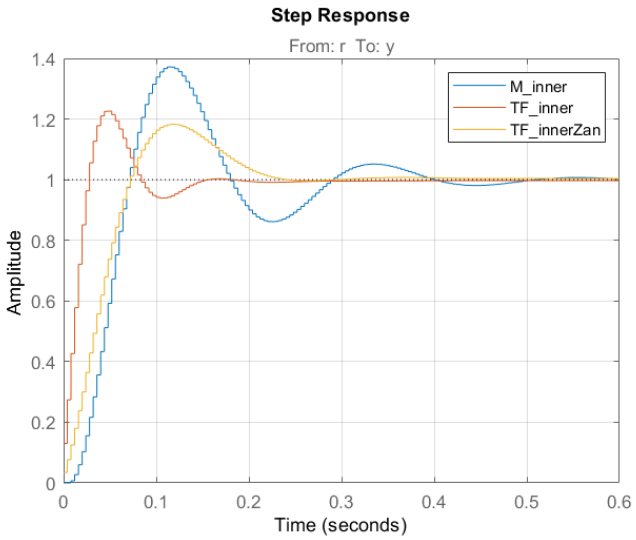


Figure 6.4: Inner Step response of the global test

While the optimization of the parameters of the controllers is given by the CMA-ES algorithm, the outer loop model reference is provided by the PSO formulation. Wherein accordance with the optimizer of the inner loop gives the following result:

θ_{ini}	K_{pi}	K_{ii}	K_{di}	K_{po}
VRFT	0.0300	0.2157	0.0004	6.1029
Cascade-PSO	0.1145	0.4396	0.0000047	6.4652

Table 6.3: Controller parameters of the global approach

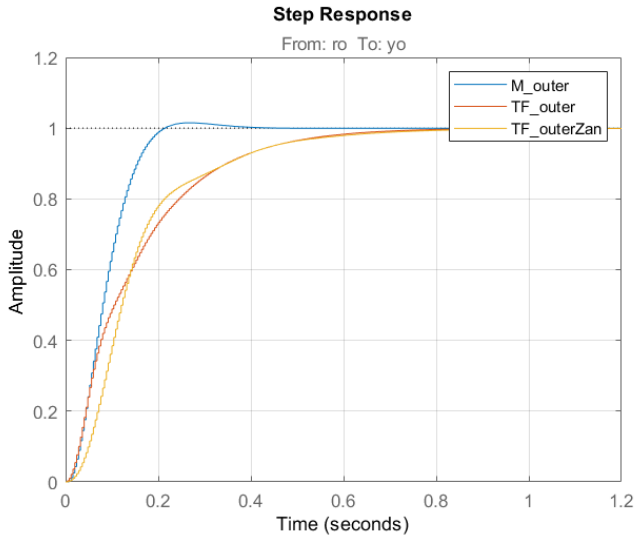


Figure 6.5: Outer Step response of the global test

Even if the results previously shown are demonstrating an acceptable performance on the simulation, the implementation on the real plant cannot be fulfilled due to the lack of stability guarantees and the fact that the resulting closed-loop response is not in accordance with the model reference shown in (6.8), This is because it is not physically implementable, and consequently the theoretical results on the VRFT are not satisfied.

$$M_O^* \left(\theta \left(\varphi\right), z\right) = \frac{1.002z + 0.002578}{z^2 + 0.0364z + 8.12{\cdot}10^{-4}} \tag{6.8}$$

Conclusions

In this work, a model-free data-driven approach for the attitude control of a quadrotor UAV has been developed. The PSO algorithm has been implemented into the VRFT method to solve a bi-level optimization problem where in the light of simulation results, the algorithm presents a further enhancement of the nominal virtual reference feedback tunning both in SISO and MIMO configurations, whilst the presence of the algorithm limitations becomes notable when the scope of the problem is to synthetize stabilizing controllers using the nominal VRFT approach to work with cascade controller structures due to the lack of information related to the settling time since the outer loop must be at least one decade slower than the inner loop to see a unitary closed-loop transfer function for the inner-one.

This gives enough motivation to propose an extension of the algorithm to deal with a cascade control configuration introducing a nonlinear cost function of the VRFT minimized using CMA-ES algorithm. The experimental results using this technique shows that the algorithm is capable to obtain stabilizing regulators with fast dynamics on the output response.

Nevertheless, the resulting model reference given is not physically implementable and the closed-loop response is in most of the cases far away from the reference meaning that the main theoretical goal of the VRFT is not being fulfilled. Moreover, the lack of stability guarantees makes the algorithm not desirable for underactuated highly nonlinear systems such as the UAV quadrotors. In contrast, when a local approach of the model-free algorithm is used, it's possible to meticulously tune the already known control parameters to enhance the performance of the aircraft.

Future developments

Since the proposed methods did not work as expected within the UAV framework, further exploration of the characteristics of the data must be performed to provide a metric that is responsible for narrowing down the optimization spectrum of the outer loop reference model. Likewise, data-based control methodologies that propose only a reference model for the cascade control structure as shown in [35] are attractive to be combined with the proposed optimization models. Diversely, the synthesis of controllers from the unfalsified control panorama contains theoretical

foundations that guarantee the stability of the system [28], so its use within the framework of the optimization of models can be advantageous in terms of performance and simplification of the computational effort when tackling the problem of cascade controller structures.

Bibliography

[1] Xin Zhang, Dexuan Zou, and Xin Shen. A novel simple particle swarm optimization algorithm for global optimization. *Mathematics*, 6(12):287, Nov 2018.

[2] Secktuoh Mora and J. Carlos. Conformación del algoritmo cma-es para la estimación de parámetros del motor de inducción asíncrono trifásico. 2014.

[3] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Advanced Textbooks in Control and Signal Processing. Springer London, 2010.

[4] R. Featherstone. *Rigid Body Dynamics Algorithms*. Springer US, 2014.

[5] Francesco Sabatino. *Quadrotor control: modeling, nonlinearcontrol design, and simulation*. PhD thesis, 2015.

[6] Qasim Ali and Sergio Montenegro. Explicit model following distributed control scheme for formation flying of mini uavs. *IEEE Access*, 4:397–406, 2016.

[7] Samir Bouabdallah and Roland Siegwart. Full control of a quadrotor. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 153–158, 2007.

[8] Nigar Ahmed and Mou Chen. Sliding mode control for quadrotor with disturbance observer. *Advances in Mechanical Engineering*, 10(7):1687814018782330, 2018.

[9] K.J. Åström and R.M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2010.

[10] K. Ogata. *Modern Control Engineering*. Instrumentation and controls series. Prentice Hall, 2010.

[11] Jan Jantzen and Carl Jakobsen. Turning pid controller tuning into a simple consideration of settling time. In *2016 European Control Conference (ECC)*, pages 370–375, 2016.

[12] Nhan T. Nguyen (auth.). *Model-Reference Adaptive Control: A Primer*. Advanced Textbooks in Control and Signal Processing. Springer International Publishing, 1 edition, 2018.

[13] Anuradha M. Narendra, Kumpati S.; Annaswamy. *Stable Adaptive Systems*. Dover Publication, 2012.

[14] M.C. Campi, A. Lecchini, and S.M. Savaresi. Virtual reference feedback tuning: a direct method for the design of feedback controllers. *Automatica*, 38(8):1337–1346, 2002.

[15] G.O. Guardabassi and S.M. Savaresi. Virtual reference direct design method: an off-line approach to data-based control system design. *IEEE Transactions on Automatic Control*, 45(5):954–959, 2000.

[16] Lucíola Campestrini, Michel Gevers, and Alexandre Sanfelice Bazanella. Virtual reference feedback tuning for non minimum phase plants. In *2009 European Control Conference (ECC)*, pages 1955–1960, 2009.

[17] Simone Formentin, Sergio Savaresi, and L. Re. Non-iterative direct data-driven controller tuning for multivariable systems: Theory and application. *Control Theory & Applications, IET*, 6:1250–1257, 06 2012.

[18] M.C. Campi and S.M. Savaresi. Direct nonlinear control design: the virtual reference feedback tuning (vrft) approach. *IEEE Transactions on Automatic Control*, 51(1):14–27, 2006.

[19] Lennart Ljung. *System Identification: Theory for User*. Prentice Hall, 2 edition, 1999.

[20] Algo Carè, Fabrizio Torricelli, Marco C. Campi, and Sergio M. Savaresi. A toolbox for virtual reference feedback tuning (vrft). In *2019 18th European Control Conference (ECC)*, pages 4252–4257, 2019.

[21] Stefano Capocchiano, Pietro Panizza, Davide Invernizzi, and Marco Lovera. Closed-loop data-driven attitude control design for a multirotor uav. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 153–158, 2018.

[22] Fast tuning of cascade control systems*. *IFAC Proceedings Volumes*, 44(1):10243–10248, 2011. 18th IFAC World Congress.

[23] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, 1995.

[24] Riccardo Poli, James Kennedy, and Tim Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, Jun 2007.

[25] Mojtaba Ahmadieh Khanesar, Mohammad Teshnehlab, and Mahdi Aliyari Shoorehdeli. A novel binary particle swarm optimization. In *2007 Mediterranean Conference on Control Automation*, pages 1–6, 2007.

[26] Wojciech Jaśkowski and Marcin Szubert. Coevolutionary cma-es for knowledge-free learning of game position evaluation. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(4):389–401, 2016.

[27] Nikolaus Hansen. The cma evolution strategy: a tutorial. 01 2010.

[28] Daniela Selvi, Dario Piga, Giorgio Battistelli, and Alberto Bemporad. Optimal direct data-driven control with stability guarantees. *European Journal of Control*, 59:175–187, 2021.

[29] Giorgio Battistelli, Daniele Mari, Daniela Selvi, and Pietro Tesi. Direct control design via controller unfalsification. *International Journal of Robust and Nonlinear Control*, 28, 02 2017.

[30] Huy Quang Nguyen, Osamu Kaneko, , and Yoshihiko Kitazaki. Virtual reference feedback tuning for cascade control systems. *Journal of Robotics and Mechatronics*, 28(5):739–744, 2016.

[31] Lucíola Campestrini, Diego Eckhard, Lydia Chía, and Emerson Boeira. Unbiased mimo vrft with application to process control. *Journal of Process Control*, 39:35–49, 03 2016.

[32] M. Yeddanapudi and A.F. Potvin. *SIMULINK: Nonlinear Control Design Blockset; User's Guide, Version 1.* Modeling, simulation, implementation. The MathWorks, 1997.

[33] L. Miskovic, A. Karimi, D. Bonvin, and M. Gevers. Correlation-based tuning of linear multivariable decoupling controllers. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 7144–7149, 2005.

[34] Håkan Hjalmarsson. Efficient tuning of linear multivariable controllers using iterative feedback tuning. *International Journal of Adaptive Control and Signal Processing*, 13(7):553–572, 1999.

[35] F. Previde, D. Belloli, A. Cologne, and S.M. Savaresi. Virtual reference feedback tuning (vrft) design of cascade control systems with application to an electro-hydrostatic actuator. *IFAC Proceedings Volumes*, 43(18):626–632, 2010. 5th IFAC Symposium on Mechatronic Systems.