Università degli studi di Forlì

SCHOOL OF ENGINEERING
Master's Degree in Aerospace Engineering

Master thesis in Advanced Guidance And Control Of Aircraft And Spacecraft

Design of a novel control system using neural network for a VTOL drone

Applicant:
Francesco Romeo

Advisor:
Prof. Paolo Castaldi

Correlators:
Eng. Lorenzo Franchi
Eng. Giuseppe Mattei

Counterpart:
Prof. Matteo Zanzi

Anno Accademico 2022/2023

Abstract

The control of dynamic systems is a highly complex discipline largely due to the non-linearity of the processes involved. Over time, numerous methodologies have been proposed to achieve efficient control, often relying on approximating models or linearizations, which, while effective, are quite distant from direct real-world control. This project aims to implement a new methodology that is increasingly gaining traction today, namely the use of neural networks for problem-solving. While dynamics that are difficult to model with standard mathematical interpretations will still be approximated in their behavior, the approach shifts towards approximators whose internal logic more closely resembles their highly nonlinear nature incredibly increasing the possibilities. This is due to their ability to automatically organize internal logic through learning, with inner relationships between neurons. Despite demonstrating their efficiency, these networks are still heavily studied to understand why they are organized in these ways.

In particular, radial basis function networks will be used, which are slightly different from all the others and are among the best approximators of linear and non-linear functions. This new control methodology will be based on a composite approach, leveraging information on estimations about the states and trajectory errors plus the implementation of a disturbance observer to tailor more the control on the different sources of disturbance. It will be applied to a simulation model of a UAV currently in the design and development phase by Sky Eye Systems, a company specialized in the production of military drones, which already boasts certifications that are difficult to obtain in the aerospace industry.

Contents

Glossary

- UAV = Unmanned Aerial Vehicle.
- $VTOL$ = Vertical Take-Off and Landing.
- FW = Fixed Wing.
- COG = Center Of Gravity.
- *Body centered frame of reference* = Frame of reference centered in the center of gravity of the body, the x-direction is directed to the nose of the aircraft, the y-direction is directed to the right wing, the z-component is perpendicular to the plane that contains the x and y direction and is directed downward.
- $NED\ frame\ of\ reference$ = Fixed frame that contains three orthogonal axes in which the x-axis points to true North, the z-axis points towards the interior of the Earth and the y-axis completes the right-handed system pointing East, and the center is positioned on the earth's surface at the start of the flight.
- $\bar{F}_b = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix}_b$ = Forces in Body frame coordinates.
- $\dot{\bar{V}}_b = \begin{bmatrix} \dot{V}_x \\ \dot{V}_y \\ \dot{V}_z \end{bmatrix}_b$ = Accelerations in the body frame coordinates.
- $\dot{\bar{V}}_e = \begin{bmatrix} \dot{V}_x \\ \dot{V}_y \\ \dot{V}_z \end{bmatrix}_e$ = Accelerations in the inertial frame coordinates.
- $\dot{\bar{V}}_b = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}_b$ = Velocity in the body frame coordinates.
- $\bar{V}_e = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}_e$ = Velocity in the inertial frame coordinates.
- $\bar{X} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ = Position in the inertial frame coordinates.

- m = Mass of the UAV.
- $I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$ = Inertia matrix of the UAV.
- $\bar{M}_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix}_b$ = Moments in the body frame coordinates.
- $\dot{\bar{\omega}} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}$ = Rate of change of body fixed angular velocity.
- $\bar{\omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$ = Body fixed angular velocity.
- $\dot{\bar{E}} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$ = Rate of change of Euler angles.
- $\bar{E} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}$ = Euler angles.
- $J1 = \begin{bmatrix} cos(\psi)cos(\theta) & sin(\phi)sin(\theta)cos(\psi) - cos(\phi)sin(\psi) & cos(\phi)sin(\theta)cos(\psi) + sin(\phi)sin(\psi) \\ cos(\theta)sin(\psi) & sin(\phi)sin(\theta)sin(\psi) + cos(\phi)cos(\psi) & cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi) \\ -sin(\theta) & sin(\phi)cos(\theta) & cos(\phi)cos(\theta) \end{bmatrix}$
  $J1 = R_b^e$ = Rotating matrix from Body fixed frame to NED frame.
- δ_a = Commanded angle for the aileron moving surface.
- δ_e = Commanded angle for the elevator moving surface.
- δ_r = Commanded angle for the rudder moving surface.
- δ_{th} = This term could have two different meanings depending on the context: it could be an infinitesimal increment or it could be the command throttle of the relative actuator.
- W_s = Wingspan of the UAV.
- MAC = Dynamic chord of the UAV.
- $\alpha = AoA$ = Angle of attack of the wing.
- ESC = Electronic Speed Controller.
- $R_\alpha = \begin{bmatrix} cos(\alpha) & 0 & sin(\alpha) \\ 0 & 1 & 0 \\ -sin(\alpha) & 0 & cos(\alpha) \end{bmatrix}$ = Rotation matrix between the reference frame of the wing and the body frame.
- IAE = Integral Time-weighted Absolute Error index.

Chapter 1

Sky Eye Systems S.r.l.

The thesis project is carried out in collaboration with Sky Eye Systems S.r.l. This company develops small tactical UAS weighing between 25 kg and 150 kg for civil and military applications, according to STANAG 4703, SAE.AER. P-2-6-7, DO 178C, MIL-F-8785C norms.

1.1 X-VTOL

The novel controller would be applied to the new X-VTOL 3.6. A small unmanned aerial vehicle (UAV) with a combined weight of approximately 50 Kg, including the drone and payload, would execute vertical take-off and landing using the rotor component and transition into a fixed-wing mode during the checkpoint phase. The intended purpose, as specified by the company's guidelines, is primarily for surveillance missions, equipped with radar and cameras, or for the delivery of critical parcels. This UAV capitalizes on its ability to operate in confined spaces for take-off and landing, as well as the endurance and efficiency of fixed-wing flight.

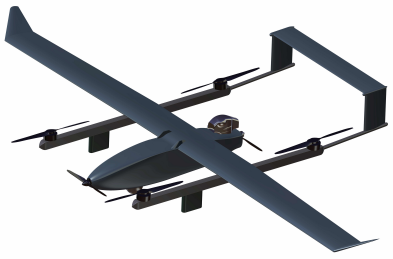

Figure 1.1: X-VTOL rendering

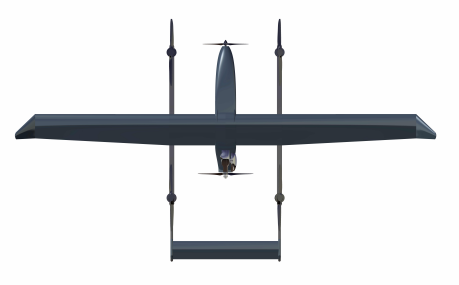

Figure 1.2: X-VTOL rendering, upper view

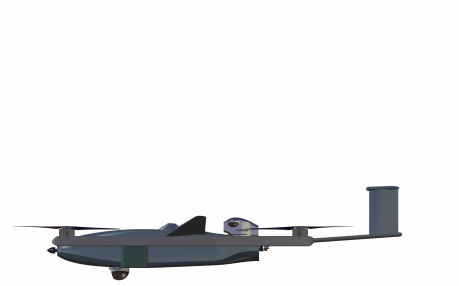

Figure 1.3: X-VTOL rendering, lateral view

Figure 1.4: X-VTOL rendering, frontal view

Chapter 2

Simulation model

The controller has been developed as part of the thesis work and operates on a highly accurate drone simulator. The simulator closely replicates various aspects of the real drone, such as aerodynamics, engine models, and sensor configurations. Specifically, the controller focuses on the rotor mode that works during take-off and landing, while the FW component utilizes a previously developed control system. This decision has been taken as a consequence of certification challenges associated with implementing neural networks or novel approaches for FW. Due to proprietary information restrictions, only necessary or summarized data is provided in this paper. The controller is developed using Simulink and utilizes an *ode*-8 solver with a fixed time-step of 0.0025 s. The chosen time step balances the dynamics that can be observed and studied with the computational resources required for the simulation.

2.1 General scheme

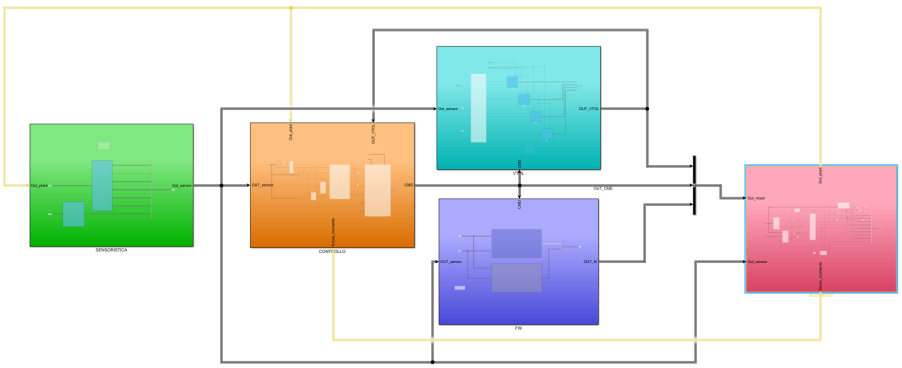

Figure 2.1: General scheme

In Figure 2.1, the general architecture of the simulator is depicted, which comprises multiple sections. This modular organization is commonly adopted in engineering practices to ensure cleanliness and ease of use. Each section is dedicated to a specific task, as evident from the example, the sensor section (green

box) encompasses all the sensor models or the VTOL block (light blue box) encompasses the model of the four rotors.

2.2 Multibody block

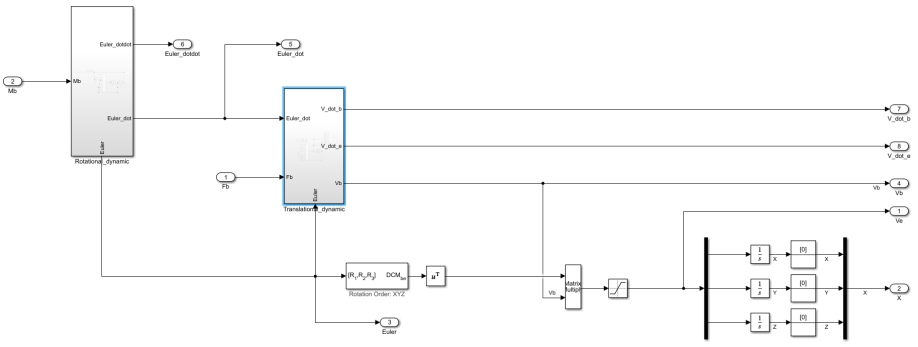

Figure 2.2: 6 DOF block

We start with the primary element representing, to a certain extent, the dynamic characteristics of the drone: the six degrees of freedom block. The dynamics of the aircraft can be represented by six differential equations: three derived from dynamics governing translations in space, and three derived from kinematics governing rotations in space, henceforth we will refer to them respectively as translational dynamics and rotational dynamics. This component receives forces and moments as inputs and computes the acceleration, as well as the acceleration about the Euler angles. It also integrates the velocity in the body frames or the angular velocity, among other functions. Equations are the general, second-order differential system [3]:

- Translational dynamic

$$\bar{F_b} = m(\dot{V_b} + \bar{\omega} \wedge V_b) \tag{2.1}$$

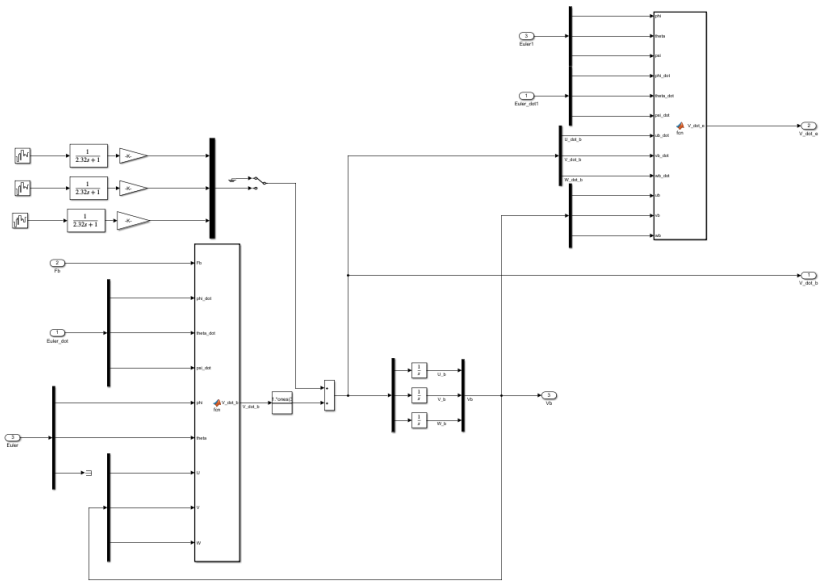

Figure 2.3: Translational dynamic block

- Rotational dynamics

$$\bar{M}_b = I\dot{\bar{\omega}} + \bar{\omega} \wedge (I\bar{\omega}) \tag{2.2}$$

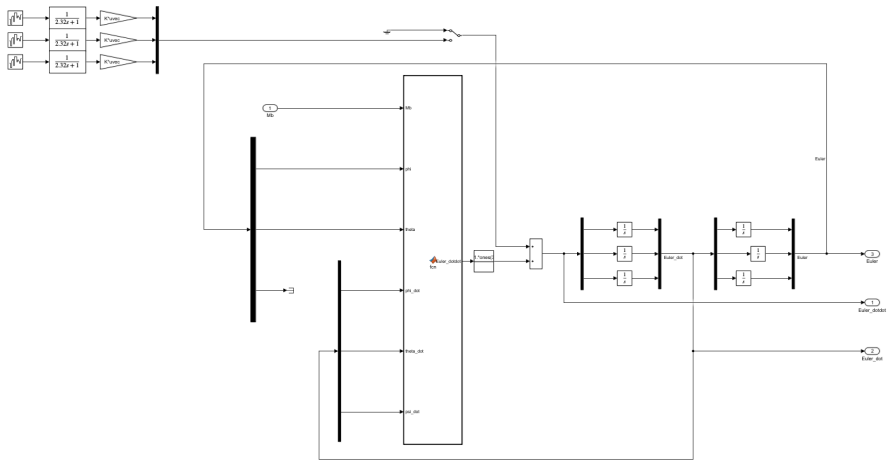

Figure 2.4: Rotational dynamic block

Using them we can obtain all the information about velocity, position, changes between reference systems etc, etc. In this subsystem, there are also some blocks that process information given by the actuators, or generally speaking, compute all the forces and moments acting on the body. The contributions can be divided into three parts:

- Gravitational contribution: It is modeled the gravitational force rotated in the body frame instantaneously through the matrix J1.

$$F_g = J_1^T \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \tag{2.3}$$

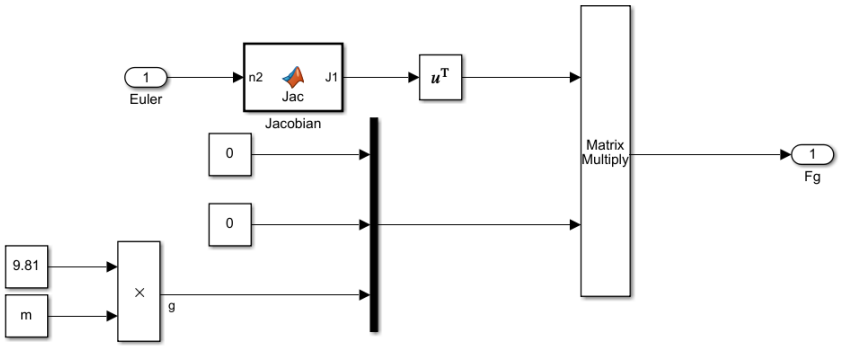

Figure 2.5: Gravity forces

- VTOL contributions: In this block, the forces and moments undergo proper rotation. Specifically, the calculation includes the counter-rotational components for moments and accounts for the moments resulting from distances from the COG.
  - Forces

$$T_{VTOL} = \sum_{i=1}^{4} \begin{bmatrix} 0 \\ 0 \\ T_i \end{bmatrix} \tag{2.4}$$

  - Moments

$$M_{transposed} = \begin{bmatrix} -bcos(\beta) & -bcos(\beta) & bcos(\beta) & bcos(\beta) \\ bsin(\beta) & -bsin(\beta) & bsin(\beta) & -bsin(\beta) \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{2.5}$$

$$M_{reaction} = \begin{bmatrix} 0 \\ 0 \\ M_1 - M_2 - M_3 + M_4 \end{bmatrix} \tag{2.6}$$

$$M_{VTOL} = M_{transposed} + M_{reaction} \tag{2.7}$$

Where T_i and M_i are respectively the thrusts and the moments generated by the i-th propeller, b is the distance between the center of the propeller and the COG, β is the angle between the arm of the rotor and the y-direction of the fixed body frame, the sign convention is according to the contribution due to the mutual position and sense of rotation of every propeller.

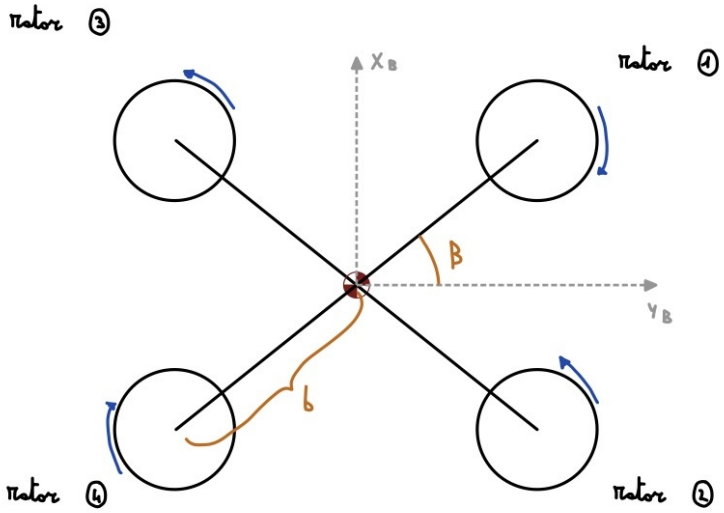

Figure 2.6: Schematic view of the drone with the rotation directions of the rotors

Rotor n°	sgn(M_x)	sgn(M_y)	sgn(M_z)
1	-	+	+
2	-	-	-
3	+	+	-
4	+	-	+

Table 2.1: Moment contribution due to an infinitesimal increment δ_{th} on the i-th rotor

- Moving surfaces and structure contributions: Sky Eye conducted a significant test campaign to analyze the aerodynamic characteristics of the model. The forces (C_x, C_y, C_z) and moments (C_l, C_m, C_n) coefficients are crucial parameters that were thoroughly investigated. These coefficients are computed by aggregating five different contributions, each obtained from a lookup table containing experimental data. Notably, two of these contributions are primarily dependent on the trim:
  - Structure contribution: it is a function of the angle of attack and the Euler rate of change.
  - β contribution: function of the sideslip angle.

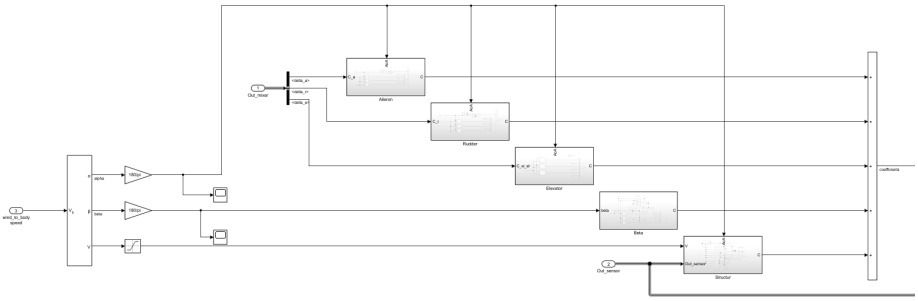

Figure 2.7: Structure of the block that computes aerodynamic coefficients

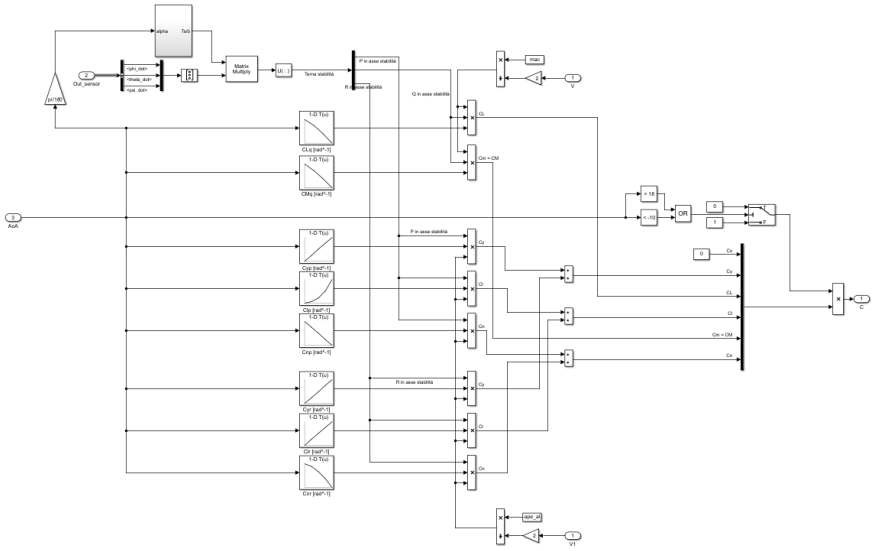

Figure 2.8: Aerodynamic contribution of the structure

The remaining three factors encompass the influence of the dynamic surfaces responsible of the aircraft control during forward flight. These factors are dependent on the respective control commands, namely δ_a, δ_e, and δ_r, as well as on the angle of attack. Special attention must be emphasized as the aerodynamic drag and lift produced by the wing are inherently included within the calculated coefficients of the elevator. It is crucial to note that these factors persist even if the angle of attack or elevator deflection angle (δ_e) are both zero.

Hereunder, the equations to obtain the actual values of the forces and moments rotated in the body frame are reported:

- $\bar{F}_b^{aerodynamic} = \frac{1}{2}\rho|V|^2 W_s R_\alpha^T \begin{bmatrix} -C_x \\ C_y \\ -C_z \end{bmatrix}$

- $\bar{M}_b^{aerodynamic} = \frac{1}{2}\rho|V|^2 R_\alpha^T \begin{bmatrix} W_s\ C_l \\ MAC\ C_m \\ W_s\ C_n \end{bmatrix}$

2.3 Sensors block

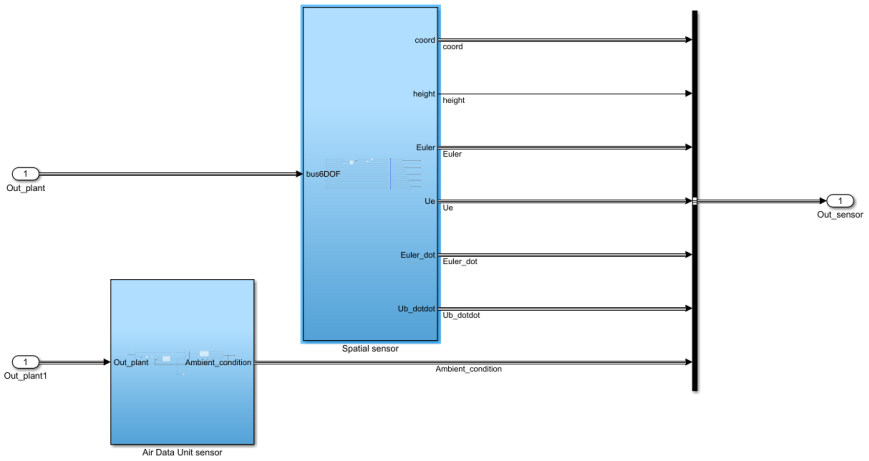

Figure 2.9: Sensors Block

In the depicted diagram shown in figure 2.9, all the sensors currently operating on the UAV are simulated. Given the limited availability of information and the necessity to filter it, caution has been exercised to incorporate only the accessible data in the controller and excluded any surplus information that could be obtained from the 6-DOF system, or to compute them analytically whenever required. For instance, the VTOL controller requires the actual thrust value, which is not obtainable without adding an additional sensor. Thus, alternative methods are employed to overcome this issue.

2.4 Control block

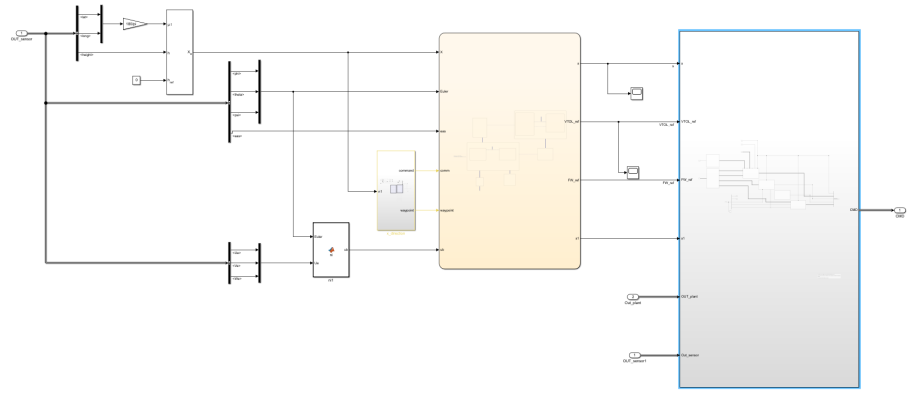

Figure 2.10: Control Block

In figure 2.10 we can see the general scheme of the control block. On the left side, there is the high-level block and on the right side, there is the low-level.

2.4.1 High-Level control

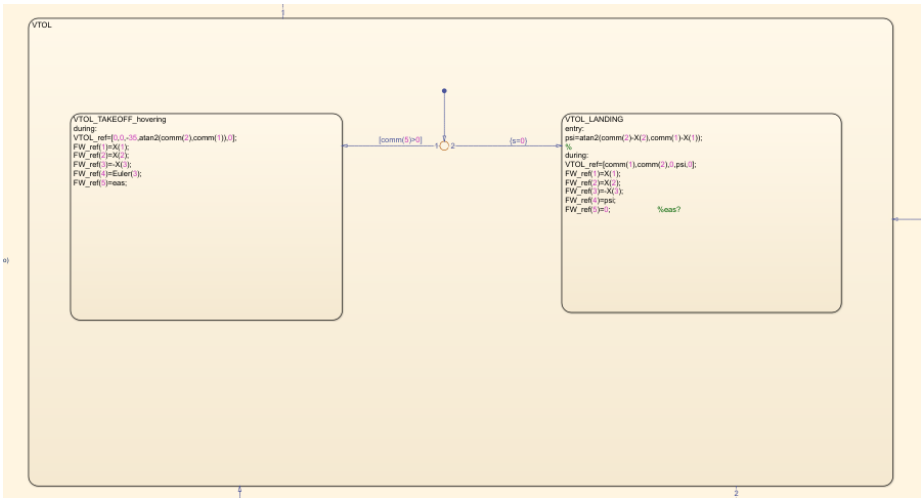

Figure 2.11: High level control particular

Figure 2.11 is an example of a small part of the high-level control block, in particular, it gives references to both VTOL and FW during the VTOL phase. This is the core of the control part of the UAV, which is made with the *StateFlow tool* supplied by Matlab. This is a way of working based on a state-machine logic. The UAV has a series of states (for example, landing phase, waypoint following, etc): depending on the actual position with respect to the waypoint given

by the mission, the control switches between these states which corresponds to different outputs given to the low-level control part.

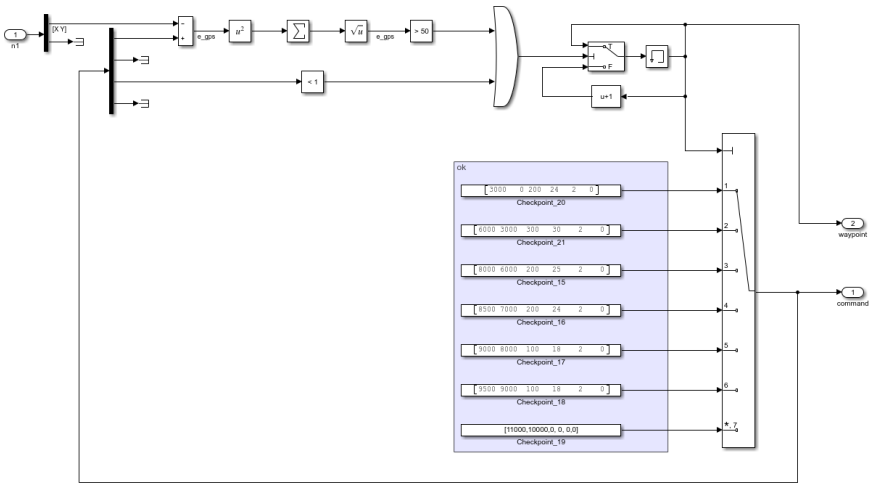

Figure 2.12: Waypoints block

The mission to be executed by the UAV is defined by the block shown in Figure 2.12. Each unique block contains specific waypoints that assign positions or velocities to be reached, providing information to the high-level control module.

2.4.2 Low-level control

The low-level control subsystem is partitioned into two sections: one responsible for controlling the FW portion, which manages the main rotor on the rear, control surfaces, and front electric motor; the other section comprising the novel control mechanism developed for the VTOL part gives reference throttles to the four rotors. The FW control mechanism adopts a widely used standard concept known as cascade PIDs, which processes information obtained from sensors, while the desired references are provided by the high-level control module. This FW control section has already been calibrated using a particle swarm algorithm available in Matlab [1].

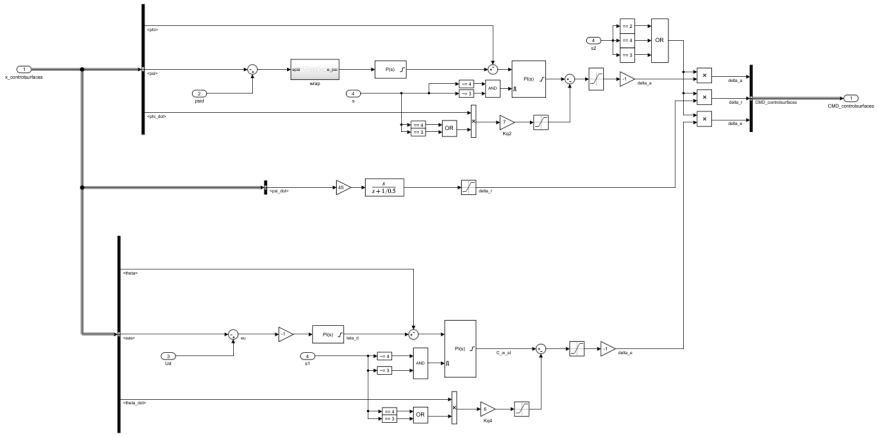

Figure 2.13: Particular of the low-level control-Moving surfaces control

On the other hand, further discussion on the VTOL controller will be provided later, but, in general, it receives input from the high-level control module and provides reference signals to the four actuators.

2.5 VTOL block

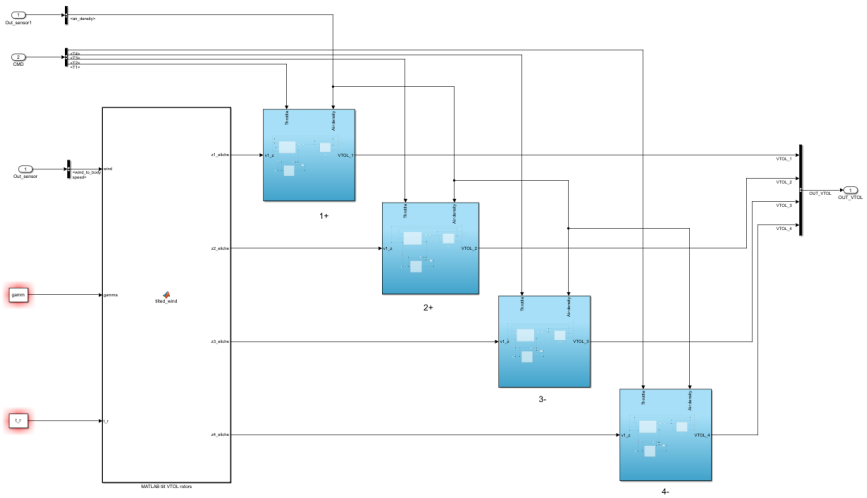

Figure 2.14: VTOL actuators block

Within this block, components parallel to the rotation axis are extracted, since it enables the possibility to simulate pitched or tilted rotors set up, alongside modeling of the actuators. The UAV features four propellers for the VTOL segment, as depicted in Figure 2.6, which also illustrates the rotation direction of the blades. Each actuator comprises a battery, an electric motor, and a propeller, with the outputs being thrusts and moments generated by each propeller.

- The battery model is an integral aspect of the thesis, as it involves constructing a discharge model based on test data obtained from discharging a real battery at various constant currents. This process results in the creation of an interpolant surface from the obtained curve. The model utilizes LIPO batteries configured with 2 banks in parallel of 6 cells in series, with a maximum voltage of 50.4 V. The model employs 2D lookup tables, wherein the instantaneous state of charge and the actual current flow, normalized by the nominal current per cell in parallel, serve as inputs, while the output provides the actual voltage.

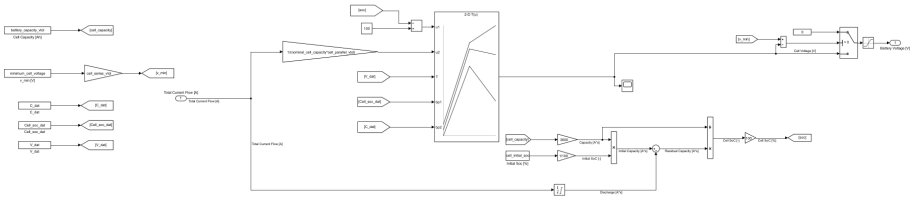

Figure 2.15: Battery model

- The electric motors currently employed in the model are of the Brushless type. These motors are also utilized in Section 2.6 for the frontal propeller, albeit with varying sizes. Generally, they are modeled using the following system of two differential equations:

$$L\dot{I}_m + RI_m + K_e\omega_m = V \tag{2.8}$$

$$K_t I_m = J_m\dot{\omega}_m + K_d\omega_m - T_{ext} \tag{2.9}$$

In the provided equations, I_m represents the current, R denotes the internal resistance of the motor, L signifies the inductance, V indicates the voltage, J_m represents the inertia of the motor in addition to the inertia of the rotor, and K_t, K_d, and K_e denote constants either provided by the manufacturer or computed utilizing real test data. This model accepts as input the actual voltage delivered by the battery and the throttle commanded by the control system as this motor is voltage-controlled. The output corresponds to the angular velocity of the shaft. Since a motor of this type needs an ESC this is modeled through a transfer function:

$$\frac{\delta V}{\delta u} = \frac{1}{0.166s + 1} \tag{2.10}$$

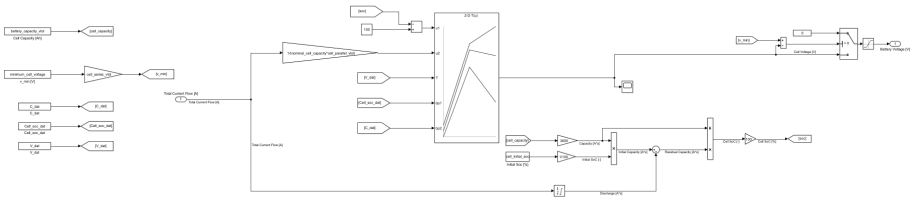

Figure 2.16: Brushless electric motor model

- The thrust and moments produced by the propellers are derived from experimental tests and manufacturer-provided datasheets. The VTOL blades have dimensions of $24x13$ $[inch]$ and typically provide a thrust of 300 N in hovering conditions at the maximum power. Additionally, a scaling factor due to the advance ratio, J, is incorporated into the model;

$$J = \frac{TAS}{\frac{RPM}{60}D} \tag{2.11}$$

RPM $[\frac{rounds}{min}]$ is the angular velocity of the electric motor shaft connected to the propeller and D $[m]$ is the propeller diameter. The thrust, T $[N]$, and the torque, M $[Nm]$, are computed:

$$T = D^4 \rho \omega^2 C_t(J, \omega) \tag{2.12}$$

$$M = \frac{D^5 \omega^3 \rho C_p(\omega)}{\omega[\frac{rad}{s}]} \tag{2.13}$$

Where ρ $[\frac{Kg}{m^3}]$ is the actual air density and the C_t [], thrust coefficient and C_p [], power coefficient that are stored in look-up tables.

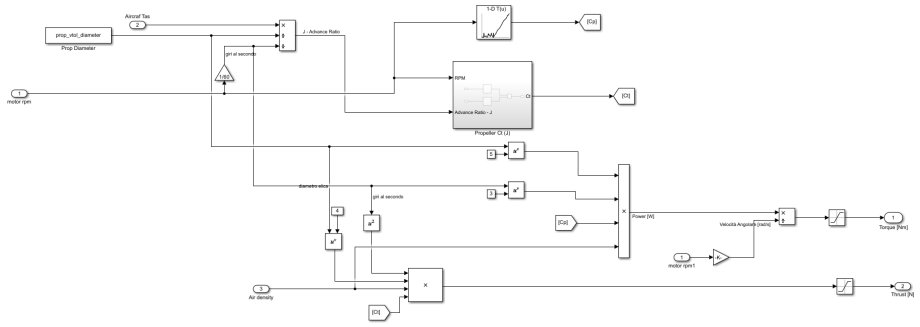

Figure 2.17: Propeller model

2.6 Motors group block

This block encompasses the modeling of motors utilized for the fixed-wing segment. The frontal motor is modeled similarly to the electric one employed in the UAV. Sky Eye sought to assess the impact of employing a counter-rotating motor during the transition from fixed-wing to VTOL, aiming to decelerate the UAV with negative thrust. Conversely, the combustion engine serves as the primary source of thrust in the fixed-wing mode. It comprises four components: the engine itself, a generator, a gearbox, and a propeller. Additionally, the generator powers add-ons such as cameras, infrared (IR) equipment, or other necessary components for the application, receive power from the engine.

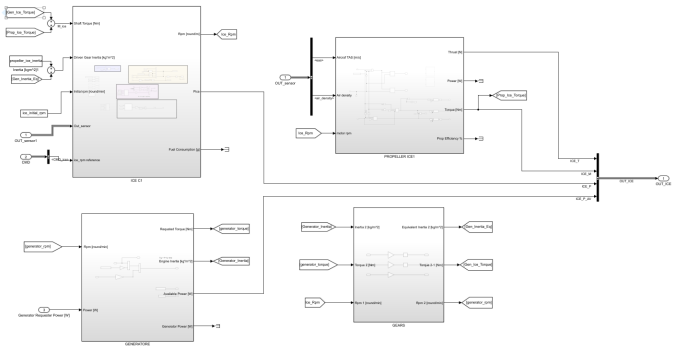

Figure 2.18: Combustion engine model

Chapter 3

VTOL Controller

The challenges posed by nonlinearity in dynamic systems are well recognized, presenting inherent obstacles to the implementation of effective control strategies. Presently, the predominant control systems effectively deployed rely on PID logic, as regulations and certification standards exhibit reluctance towards adopting new strategies in real-world applications. The aim of this thesis is to introduce a novel approach for addressing nonlinearity by utilizing neural networks for estimation. Additionally, the thesis aims to propose a controller designed for the previously described near-real simulator, specifically targeting control of the VTOL component. This approach was chosen because the use of innovative methodologies for this flight segment is more likely to achieve certifications.

3.1 Problem formulation

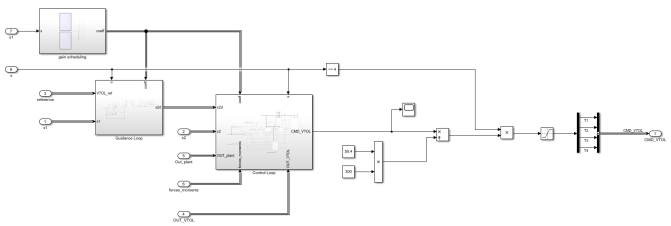

Figure 3.1: VTOL controller which includes the tuning parameters scheduling, the inner loop and the outer loop

The entire control system is based on the assumption that the aircraft could be modeled with a second-order system in the form:

$$\ddot{\chi} = F(\chi, u) + B(\chi) + \Delta(\chi, u) + D(t) \tag{3.1}$$

In this context, we denote χ as the system states, u as the inputs, and Δ as a term addressing nonlinearity arising from unmodeled dynamics or model inaccuracies. Additionally, D represents uncertainties that vary with time, such

as disturbances originating from wind. This categorization into two sources of errors requiring estimation stems from the need to simplify the modeling of neural networks. Designing neural networks with time-dependent characteristics is challenging, as achieving convergence with time-dependent weight matrices or structures is cumbersome. In particular we can identify two subsystems: one that refers to the rotational dynamics:

$$\ddot{\bar{E}} = F_{rot}(\chi, u) + B_{rot}(\chi)u + \Delta(\chi, u) + D(t) \tag{3.2}$$

The second subsystem instead refers to the translational dynamics:

$$\dot{\bar{V}}_b = F_{trans}(\chi, u) + B_{trans}(\chi)u + \Delta(\chi, u) + D(t) \tag{3.3}$$

Collectively, these elements form a system of six nonlinear equations, but this method could be applied to every dynamic system. The structure of these equations is presented in Appendix A for the $F(\chi)$ and $B(\chi)$ parts, specializing them for this case.

The VTOL controller functions as an autopilot, tasked with guiding the aircraft to follow references provided by the high-level control system or trajectory commands issued by the operator. Its output comprises throttle commands or the thrusts needed.

The main structure of the controller is depicted in Picture 3.1. It consists of an Outer Loop responsible for position control, where references in $x_{d1} = [x_d \ y_d]^T$ are transformed into references of $\bar{x}_{d2} = [\phi_d \ \theta_d]$, used in the inner loop, which governs faster variables, with references denoted as $x_{d2} = [\bar{x}_{d2} \ \psi_d \ z_d]$[2][12].

3.2 Outer loop

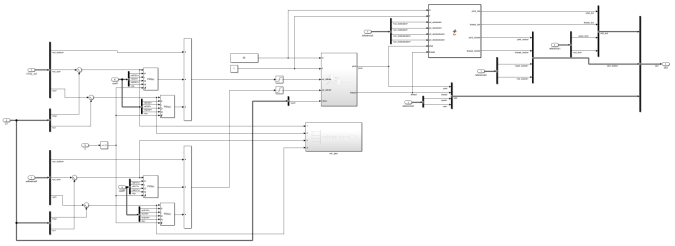

Figure 3.2: Outer loop of the VTOL controller

The outer loop implies a simple inversion law to compute the references [4]. Starting from $F = m\ \ddot{\bar{X}}$, we obtain that:

$$J_1 \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} T_i \end{bmatrix} = m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} \tag{3.4}$$

Taking the first two components of this equation:

$$\begin{bmatrix} cos(\phi)sin(\theta)cos(\psi) + sin(\phi)sin(\psi) \\ cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi) \end{bmatrix} \sum_{i=1}^{4} T_i = m \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} \tag{3.5}$$

With some mathematical passages:

$$\begin{bmatrix} cos(\phi_d)sin(\theta_d) \\ sin(\phi_d) \end{bmatrix} = -\frac{\sum_{i=1}^{4} T_i}{m} \begin{bmatrix} cos(\psi) & sin(\psi) \\ sin(\psi) & -cos(\psi) \end{bmatrix} \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} \tag{3.6}$$

As evident from the schematic representation 3.2, this section employs several PID controllers that account for errors arising from trajectory and velocity discrepancies. Within this loop, computations are performed to determine the desired first and second derivatives with respect to $\bar{x}_{2d}$, as they are essential for the inner loop operations. The mathematical expressions derived from the time differentiation of equations 3.6 are presented:

$$\dot{\phi}_d = \frac{m}{\sum_{i=1}^{4} T_i} \frac{y_d^{iii}}{cos(\phi_d)} \tag{3.7}$$

$$\dot{\theta}_d = (-\frac{m}{\sum_{i=1}^{4} T_i} x_d^{iii} + \dot{\phi}_d sin(\phi_d) sin(\theta_d)) \frac{1}{cos(\phi_d) cos(\theta_d)} \tag{3.8}$$

$$\ddot{\phi}_d = (\frac{m}{\sum_{i=1}^{4} T_i} y_d^{iv} + \dot{\phi}_d^2 sin(\phi_d)) \frac{1}{cos(\phi_d)} \tag{3.9}$$

$$\begin{aligned} \ddot{\theta}_d = (\frac{m}{\sum_{i=1}^{4} T_i} \ddddot{x}_d - \ddot{\phi}_d \, sin(\phi_d) sin(\theta_d) - \dot{\phi}_d^2 cos(\phi_d) sin(\theta_d) - \\ -2\dot{\phi}_d \dot{\theta}_d sin(\phi_d) cos(\theta_d) - \dot{\theta}_d^2 cos(\phi_d) sin(\theta_d)) \frac{1}{-cos(\phi_d) cos(\theta_d)} \end{aligned} \tag{3.10}$$

3.3 Inner Loop

This section constitutes the foundational aspect of the research, employing various mathematical tools and methodologies to develop a robust controller. Within this chapter, we will provide a concise overview of essential mathematical fundamentals and the overall structure utilized in the development process.

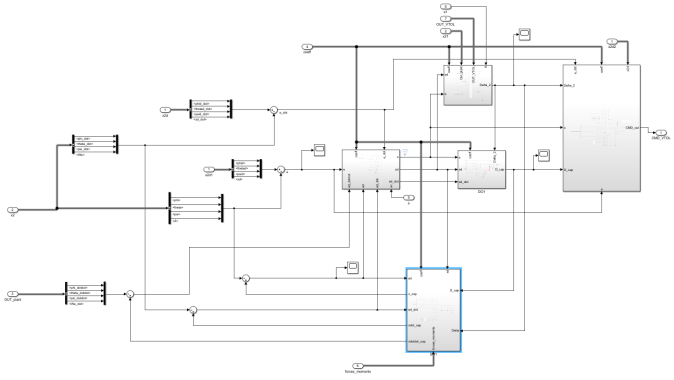

Figure 3.3: Inner loop of the VTOL controller

3.3.1 Radial Basis Functions

The decision to utilize Radial Basis Function Neural Networks (RBFNN), henceforth referred to as RBFNN, arises from the requirement to procure a swift and dependable approximator for nonlinear contribution functions [5]. These neural networks emerged as a solution to the imperative of simulating the processing capabilities of biological systems. RBFNNs acquire knowledge directly from data, making their training process considerably less arduous compared to other renowned neural networks. The typical structure of an RBFNN neuron is as follows:

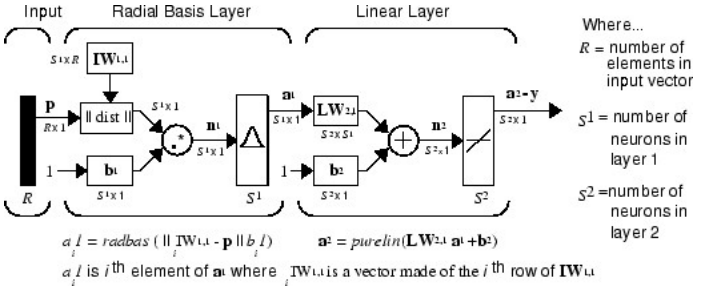

Figure 3.4: Single RBF neuron

They possess a unique structure compared to other well-known neural network architectures, notably the simple perceptron. In picture 3.4, a single neuron is presented for simplicity, although it is straightforward to extend to a complete network, with the general scheme presented later. Upon examination, two main distinctions from conventional neural networks emerge: inputs are not directly multiplied by weights and added to the bias; rather, is computed the distance from the weights and then multiplied by the bias. Consequently, the actual input for the first layer is (In subscript indicates to which layer the formula belongs):

$$n^1 = ||p - w||^1 b^1 \tag{3.11}$$

Many types of transfer functions could be chosen, in this particular case gaussian 3.5 is utilized:

$$a = e^{-n^2} \tag{3.12}$$

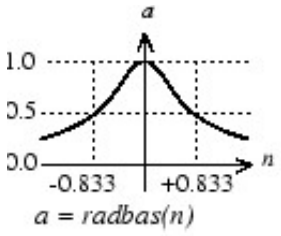

Figure 3.5: Gaussian transfer function

Traducing in a simpler way the weight is the center of the Gaussian instead the bias is the scaling factor for the input. On the other end, the second layer,

called linear as well, is standard:

$$a^2 = W^2 a^1 + b^2 \tag{3.13}$$

Generalizing to a network, each set of inputs is subtracted from the row of weights, which becomes the center of different Gaussians, scaled by the biases. Consequently, RBFNNs comprise two layers, and the training algorithm primarily focuses on determining the optimal number of neurons, the positions of the centers, and the spread for Gaussian transfer functions. Implementing an RBFNN in the control system involves utilizing the Feedback Error Learning technique, simplifying the controller significantly. The NNs contribute to stability, supported by Lyapunov theory[8]. To further enhance stability, a disturbance observer coupled with a state observer is employed, reflecting a composite learning methodology approach.

3.3.2 State observer

The primary role of a state observer [6] is to deduce variables that otherwise are unmeasurable, serving a multitude of applications. Initially, it was discovered that a more precise estimation could be achieved by integrating more accurate information into the observer. This encompasses understanding noise and disturbances, which are characterized by deterministic, differential, polynomial, bounded, and stochastic descriptions, in this case, these informations comes from the disturbance observer and the neural network. In this controller, a Luenberg observer, in the input-output-based observer (IOBO) formulation, is implemented. It could be summarized as follows:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{3.14}$$

$$IOBO : (u, y, A, B, C) \rightarrow (\hat{x}) \tag{3.15}$$

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x}) \tag{3.16}$$

Given that the estimate is circulated back through the estimator, it is frequently referred to as a Closed Loop Observer. The primary advantage of the IOBO lies in its capability to utilize both input and output data, along with plant information, to minimize noise and phase lag without requiring knowledge of initial conditions.

3.3.3 Control algorithm

Recalling the affine form of the dynamic system, presented in Appendix A we can see that we have to consider only a reduced system and not all the six differential equations. In particular in the inner loop the $F(\chi)$ vector and the $B(\chi)$ matrix are composed of the complete structure of rotational dynamics and the third row of translational dynamics. We define the trajectory tracking error as:

$$e = x_2 - x_{d2} \tag{3.17}$$

A filtered tracking error is introduced and its first derivative:

$$s = \dot{e} + \lambda e + \lambda_1 \int e dt \tag{3.18}$$

$$\dot{s} = \ddot{e} + \lambda\dot{e} + \lambda_1 e \tag{3.19}$$

Upon examination of s, we observe its resemblance to the structure of a PID controller and we can consider it a generalization thereof. Here, λ and λ_1 are square matrices that serve as tuning parameters. It is imperative that these matrices must be definite positive but not necessarily diagonal. Introducing off-trace elements in λ and λ_1 induces a form of coupling between the dynamics, inherently amplifying the challenges associated with tuning. Introducing the observed state $\hat{x}_2$ and defining the estimation error as:

$$e_d = \hat{x}_2 - x_2 \tag{3.20}$$

It is possible to introduce a filtered estimation error and its first derivative:

$$s_d = \dot{e}_d + \lambda e_d + \lambda_1 \int e_d dt \tag{3.21}$$

$$\dot{s}_d = \ddot{e}_d + \lambda\dot{e}_d + \lambda_1 e_d \tag{3.22}$$

Since Δ is not known is approximated with the RBFNN and could be formulated as:

$$\Delta = W^{*T}\mu(\chi) + \epsilon \tag{3.23}$$

W^* is the optimal weight matrix, μ is the vector of basis functions and ϵ is the approximation bounded error, as $||\epsilon|| \leq \epsilon_{max}$. We define the variables with the hat as the approximations of the respective variable, for example for Δ the approximation made by the NN is:

$$\hat{\Delta} = \hat{W}^T\mu(\chi, u) \tag{3.24}$$

In this controller is adopted the methodology of normalizing the inputs, so every input variable is divided by the maximum value that could assume, knowing the general mission properties. The state observer is introduced by implementing the following state space model:

$$\hat{\dot{x}} = F(\chi) + B(\chi)u + \hat{\Delta}(\chi, u) + \hat{D}(t) - k_1 s_d - \lambda\dot{e}_d - \lambda_1 e_d \tag{3.25}$$

$\hat{D}$ is the approximation, computed by the disturbance observer lately presented:

$$\hat{D}(t) = D(t) + \epsilon \tag{3.26}$$

With all the previous information we could construct the control command:

$$u = B(\chi)^{-1}[\ddot{x}_d - \lambda\dot{e} - \lambda_1 e - k_1 s - F(\chi) - \hat{\Delta} - \hat{D}] \tag{3.27}$$

Some considerations must be made:

- Typically, aircrafts with a single rotor are considered under-actuated systems since $dim(u) < dim(x)$, where x represents the states controlled in the inner loop. However, in the case of a quadrotor employing this type of control, the system becomes squared. With four controlled variables $x_2 = [\phi\ \theta\ \psi\ z]$ and four inputs $u = [T_1\ T_2\ T_3\ T_4]$, we have $dim(x_2) = dim(u)$. This presents an advantage as $B(\chi)$ results in a square matrix. Consequently, there are fewer complications if dynamic inversion were to be chosen as a possible approach. However, dynamic inversion is not selected at this moment, since we want a condition where the basic information are known.

- $F(\chi)$ and $B(\chi)$, in this scenario, have been linearized under near-hovering conditions. This represents the worst-case scenario, as it provides minimal insight into the system, except for basic information such as mass and inertia matrices, which are readily available once the drone's design is finalized. This decision places a heavier burden on the RFBNN, as it must approximate numerous nonlinearities. However, the effectiveness of rejecting these nonlinearities will be evaluated. The choice of near-hovering conditions over pure hovering was made to preserve all relationships and couplings between variables that would be lost in pure hovering. The linearization conditions are outlined below:

$$\begin{aligned} &\phi = 0.1 \\ &\theta = 0.1 \\ &\dot{\phi} = 0.1 \\ &\dot{\theta} = 0.1 \\ &\dot{\psi} = 0.1 \end{aligned} \tag{3.28}$$

To obtain some information about the updating law of the NN we must study the stability of the system. But before some relationships are needed, so starting from the first derivative of the tracking error 3.19 and substituting the commanded input 3.27 we obtain:

$$\dot{s} = \ddot{x} - \ddot{x}_d + \lambda\dot{e} + \lambda_1 e \tag{3.29}$$

$$\dot{s} = F(\chi) + B(\chi)u + \Delta + D - \ddot{x}_d + \lambda\dot{e} + \lambda_1 e \tag{3.30}$$

$$\begin{aligned} \dot{s} = F(\chi) + B(\chi)[B(\chi)^{-1}(\ddot{x}_d - \lambda\dot{e} - \lambda_1 e - k_1 s - F(\chi) - \hat{\Delta} - \hat{D})] + \\ + \Delta + D - \ddot{x}_d + \lambda\dot{e} + \lambda_1 e \end{aligned} \tag{3.31}$$

$$\dot{s} = -(k_1 s + \bar{\Delta} + \bar{D}) \tag{3.32}$$

Where $\bar{D} = \hat{D} - D$ and $\bar{W} = \hat{W} - W$ has been defined. Instead for the filtered estimation error similarly:

$$\dot{s}_d = \ddot{e}_d + \lambda\dot{e}_d + \lambda_1 e_d \tag{3.33}$$

$$\dot{s}_d = \hat{\ddot{x}} - \ddot{x} + \lambda\dot{e}_d + \lambda_1 e_d \tag{3.34}$$

$$\begin{aligned} \dot{s}_d = F(\chi) + B(\chi)u + \hat{\Delta}(\chi, u) + \hat{D}(t) - k_1 s_d - \lambda\dot{e}_d - \lambda_1 e_d - \\ -(F(\chi) + B(\chi)u + \Delta(\chi, u) + D(t)) + \lambda\dot{e}_d + \lambda_1 e_d \end{aligned} \tag{3.35}$$

$$\dot{s}_d = -(k_1 s_d - \bar{\Delta} - \bar{D}) \tag{3.36}$$

The instrument adopted to study stability is a Lyanupov function (appendix B reports some information about Lyanupov stability theory) formulated as:

$$V = \frac{1}{2}[s^T k_2 s + s_d^T k_3 s_d + \bar{D}^T \bar{D} + tr(\bar{W}^T \Gamma \bar{W})] \tag{3.37}$$

Where k_2, k_3 positive definite matrices, Γ learning rate of the NN (is a parameter that determines how fast NN learns from input). The first derivative of 3.37 is:

$$\begin{aligned} &\dot{V} = k_2 s^T(-k_1 s - \bar{W}^T\mu - \bar{D}) + k_3 s_d^T(-k s_d + \bar{W}^T\mu + \bar{D}) + \\ &+ tr(\bar{W}^T\Gamma^{-1}\dot{\bar{W}}) + \bar{D}^T(\dot{\hat{D}} - \dot{D}) \end{aligned} \tag{3.38}$$

With Making the assumption $\dot{\hat{W}} = \dot{\hat{W}}$ could be proven the uniform stability condition guaranteed by the two sequent updating laws:

$$\dot{\hat{W}} = \Gamma[\mu(k_2 s^T - k_3 s_d^T) - \sigma_W \hat{W}] \tag{3.39}$$

$$\dot{\hat{D}} = (k_2 s - k_3 s_d) - k_4(\dot{s}_d + k_1 s_d) \tag{3.40}$$

The term $\sigma_w \hat{W}$ represents the so-called σ-modification [7], this therm introduces inertia so the weight is influenced by the value of the previous computed, this technique enables significant robustness and speed of adaptation of the weights of the neural networks, preventing uncontrolled growth without bounds. Below is the code to implement an RBFNN with the previously proposed updating law:

```
function [Delta_2,W_dot_cap] = fcn(chi,s,sd,Gamma_2,k2,k3,W_cap)

U_max=[chi_max];
U_norm=([chi]./U_max)';

center=-1+2*rand(50,8);
[n,¬]=size(center);mu=zeros(n,1);
center=center';
width=1.2+4*rand(1,50);
for i=1:n
    mu(i)=exp(-norm(U_norm-center(1:end,i))^2/width(i));
end
W_dot_cap=Gamma_2*(mu*(k2*s-k3*sd)');
Delta_2=W_cap'*mu;
```

Using equation 3.36, we can eliminate the need to compute $\dot{s}_d$, in the disturbance observer formula 3.40 as follows:

$$\dot{s}_d + k_1 s_d = \bar{W}^T \mu + \bar{D} \tag{3.41}$$

$$\dot{\hat{D}} = (k_2 s - k_3 s_d) - k_4(\bar{W}^T \mu + \bar{D}) \tag{3.42}$$

A minor observation regarding the reference model is warranted, as the command provided by this model to the 6-DOF block is a thrust, assuming linearity in the dependency of moments on this input. However, this assumption does not always hold true, and only an actuator model can provide accurate values. The actuator model requires a command in throttle, which is derived by normalizing the thrust command by the maximum achievable value (e.g., 300N) and the maximum voltage supplied by the battery to the electric motor.

3.3.4 Particle swarm algorithm

The control system requires numerous tuning parameters to function effectively. From one perspective, this provides the operator with extensive control over the process being regulated. However, finding a tuning combination that ensures the proper functioning of the control system can be quite challenging. It has been observed during numerous tests that the tuning parameters must be adjusted according to the given references. Different parameter values are required if the references are fixed angles or a continuous trajectory. A brief overview is provided on the parameters within the system and their respective influences:

- λ = Influences the contribution of the proportional term of the filtered variable.
- λ_1 = Influences the contribution of the integral term of the filtered variable.
- k_1 = In general, governs the contribution of the filtered variable s inside the control law and is needed for stability.
- k_2 = In general, governs the contribution of the filtered variable s to the updating laws of the disturbance observer and the neural network and is needed for stability.
- k_3 = In general, governs the contribution of the filtered variable s_d to the updating laws of the disturbance observer and the neural network and is needed for stability.
- k_4 = In general, governs the contribution of the term $\hat{\Delta}+\hat{D}$ in the updating law of the disturbance observer and is needed for stability.
- Tuning parameters about the PIDs inside the outer loop, they contribute to the various derivative, proportional, and integral contributions
- Γ = Learning rate of the RNBFNN.
- $\sigma_w = \sigma$ modification term.

The last two tuning parameters, along with the number of neurons, are crucial for ensuring the proper behavior of the RBFNN. It is acknowledged that the primary influence on the behavior of e_d stems from the k-th terms. Thus, effective tuning should aim to minimize these terms, ensuring the disturbance observer functions appropriately, instead the tracking error is more influenced by the lambdas therms.

Tuning all these parameters simultaneously can be a laborious task due to the numerous variables involved. Therefore, an optimization technique is chosen to streamline this process. The main aspects of the Particle Swarm Optimization (PSO) algorithm are outlined below, along with the code utilized. It is noteworthy that an handwritten code is preferred over built-in code for implementation.

Theory

Particle Swarm Optimization (PSO) is a stochastic optimization algorithm that functions within a population-based framework, drawing inspiration from the intelligent collective behaviors observed in certain animal groups such as bird flocks or fish schools. PSO simulates the social behaviors exhibited by various animals, including insects, herds, birds, and fish. These swarms demonstrate cooperative behaviors aimed at locating food sources, with individual members dynamically adjusting their search patterns based on their own learning experiences and those of other swarm members. By studying the behaviors of social animals within the context of artificial life theory, researchers aim to develop strategies for constructing artificial swarms capable of efficient problem-solving.

Several fundamental principles are considered, as referenced in [10]:

1. Proximity: the swarm should be able to carry out simple space and time computations.

2. Quality: the swarm should be able to sense the quality change in the environment and respond to it.

3. Diverse response: the swarm should not limit its way to get the resources in a narrow scope.

4. Stability: the swarm should not change its behavior mode with every environmental change.

5. Adaptability: the swarm should change its behavior mode when this change is worthy.

In Particle Swarm Optimization (PSO), particles possess the capability to dynamically adjust their positions and velocities in response to environmental changes, thereby fulfilling the requirements for proximity and quality. Moreover, the PSO swarm operates without constraining its movement, instead continuously exploring the feasible solution space in pursuit of the optimal solution. Particles within the PSO system maintain a consistent movement pattern within the search space while also adapting their movement mode to accommodate environmental changes. Consequently, particle swarm systems adhere to the aforementioned five principles.

The PSO algorithm constitutes a searching process based on swarm intelligence, wherein each entity is termed a particle. Defined as a potential solution within the D-dimensional search space, each particle can store information regarding both the optimal positions of the entire swarm and its own, alongside its velocity. During each generation, the collective information of particles is amalgamated to adjust the velocity of each dimension, facilitating the computation of the particle's new position. Particles undergo continuous state changes within the multi-dimensional search space until achieve an equilibrium, an optimal state, or surpass computational limits. The introduction of objective functions establishes a unique linkage among the various dimensions of the problem space. Numerous empirical studies have demonstrated the effectiveness of this algorithm as an optimization tool.

Algorithm

The main flowchart is presented:

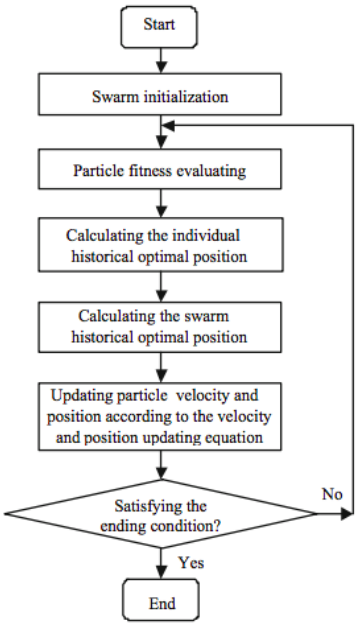

Figure 3.6: PSO flowchart

PSO can be described as follows: assume that swarm size is N, each position and velocity particle's vector in D-dimensional space are respectively:

$$X_i = (x_i^1, x_i^2, -, x_i^d, -, x_i^D) \tag{3.43}$$

,

$$V_i = (v_i^1, v_i^2, -, v_i^d, -, v_i^D) \tag{3.44}$$

Individual's optimal position vector is;

$$P_i = (p_i^1, p_i^2, -, p_i^d, -, p_i^D) \tag{3.45}$$

Swarm's optimal position, the optimal position that any individual in this swarm has experienced) is represented as

$$P_g = (p_g^1, p_g^2, -, p_g^d, -, p_g^D) \tag{3.46}$$

Considering the minimizing problem as an example, updating formula of the individual's optimal position is:

$$p_{i,t+1}^d = \begin{bmatrix} x_{i,t+1}^d & if \quad f(X_{i,t+1}) < f(P_i, t) \\ p_{i,t}^d & otherwise \end{bmatrix} \tag{3.47}$$

The swarm's optimal position is that of all the individuals optimal positions. Update formula of velocity and position is denoted as respectively, using the inertia term,ω formulation:

$$v_{i,t+1}^d = \omega v_{i,t}^d + c1rand(p_{i,t}^d - x_{i,t}^d) + c2rand(p_{g,t}^d - x_{i,t}^d) \tag{3.48}$$

In the PSO algorithm, two distinct versions exist: the global version and the local version. In the global version, particles track two extremes: their own

optimal position (p_{best}) and the optimal position of the entire swarm (g_{best}). Conversely, in the local version, particles, in addition to tracking their own optimal position (p_{best}), monitor the optimal positions of all particles within their topological neighborhood, termed n_{best}.

Examining the velocity update formula from a sociological perspective reveals its components. The first component reflects the influence of the particle's previous velocity, indicating the particle's confidence in its current movement state and its inertial movement based on its existing velocity. This component is governed by the inertia weight parameter, denoted as ω.

The second component is contingent upon the distance between the particle's current position and its own optimal position, referred to as the "cognitive" item. This component signifies the particle's independent decision-making process, or cognitive behavior, stemming from its individual experiences. The parameter associated with this component is termed the cognitive learning factor, denoted as c_1.

The third component is dependent on the distance between the particle's current position and the global (or local) optimal position within the swarm, referred to as the "social" factor. This component represents the information sharing and cooperation among particles, wherein a particle's movement is influenced by the experiences of others within the swarm. This aspect simulates the movement of a particle influenced by the collective knowledge of the swarm, and it is governed by the social learning factor parameter, denoted as c_2 (also known as the social acceleration factor). The choices about the parameters could be found in [10] where some values are suggested. ITAE index is chosen as the objective function, but many others are possible:

$$ITAE = \int_{t=0}^{t=+\infty} t|e| \tag{3.49}$$

This index is proven to optimize a smooth response of the system without saturating the actuators, measures the increased error in the system that results in a good underdamped system, and adds a weight that penalizes errors that occur later [11]. Appendix C presents the script used in Matlab.

Chapter 4

Results

The principal outcomes of this thesis endeavor are presented in the following. The outcomes can be bifurcated into two primary segments. The initial segment features a simplified simulator wherein the UAV exclusively utilizes the VTOL component to exhibit resilience against wind perturbations, a critical concern highlighted by Sky Eye for scrutiny and mitigation. Subsequently, the second segment entails the comprehensive simulator, showcasing the deployment of a novel controller onto an aircraft tasked with executing missions involving both VTOL and fixed-wing flight operations. The wind model remains excluded from the comprehensive simulator due to limitations of the incumbent fixed-wing controller in handling such disturbances.

4.1 Performance analysis of the controller applied to a simplified model

The primary aspect overlooked in this simulator, apart from its exclusive utilization of the VTOL component, is the absence of the actuator model. Consequently, the commands issued to the 6-degrees-of-freedom (6-DOF) representation, which essentially embodies the aircraft's dynamics under external forces, solely pertain to the commanded thrusts. All other factors, including nonlinearities such as aerodynamic forces, remain accounted for, ensuring a highly accurate approximation of the aircraft's real dynamic behavior. The trajectory commanded closely mirrors what the VTOL would pursue during an actual mission, as take-off is modeled accordingly. Here is presented the mission in terms of coordinates and commanded heading angle:

$$\begin{aligned} x_d &= 0 \\ y_d &= 0 \\ z_d &= 30e^{-t/10} - 30; \\ \psi_d &= 0; \end{aligned} \tag{4.1}$$

Where t is the time. A brief note is warranted regarding the controller's requirements, which extend to the fourth-order derivatives of the commanded

trajectories. While these derivatives are not explicitly provided here, they can be readily derived through straightforward mathematical methods, given their selection as continuous functions of time. Additionally, a remark is necessary regarding the z-component, wherein the commanded trajectory must be negative in this instance, aligning with the reference system. The moving surfaces are maintained undisturbed considering $\delta_a = 0$, $\delta_e = 0$ and $\delta_r = 0$ but since the aerodynamic forces and moments are also functions of other variables their contributions are not zero, and their behavior is presented as an example since their contributions depends on the actual trim:

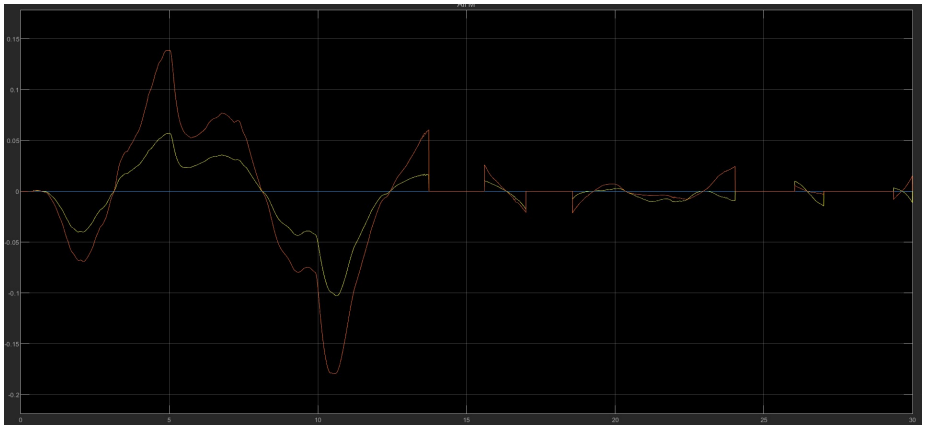

Figure 4.1: Aerodynamic moments[Nm]

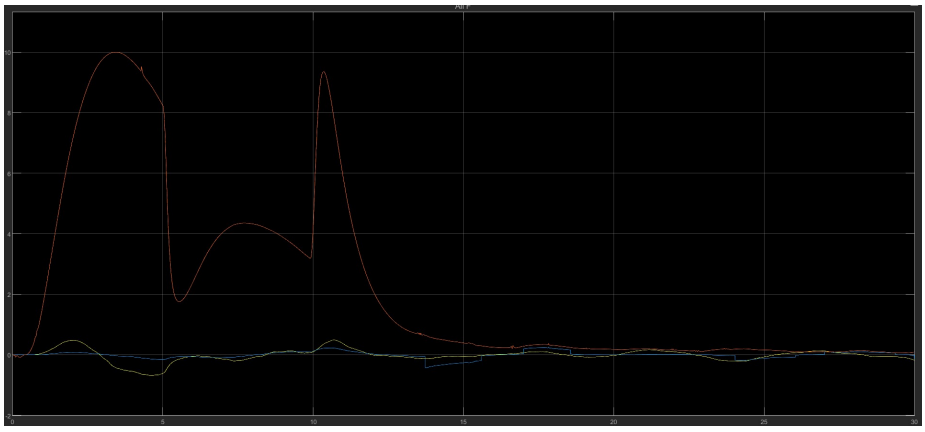

Figure 4.2: Aerodynamic forces[N]

The aircraft operates under challenging conditions due to the presence of wind, which is characterized using a stochastic model. In practice, the modeled wind is generated using a white noise block, which is then adjusted by a transfer function that alters its time constant, normalized by its standard deviation, and scaled by the selected power:

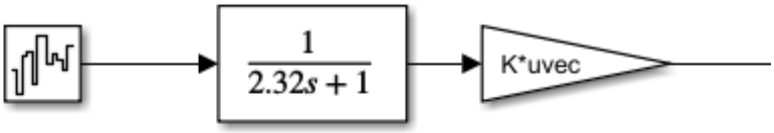

Figure 4.3: Aerodynamic forces[N]

Disturbances are summed up to all the accelerations, in addition, is tested also a vertical wind shear condition (an abrupt vertical acceleration, directed downward, modeled with a trapezoidal curve) in the z-direction, easily recognizable as the red line in image 4.5. Hereunder is presented instead the trends over time of the winds applied in each component.

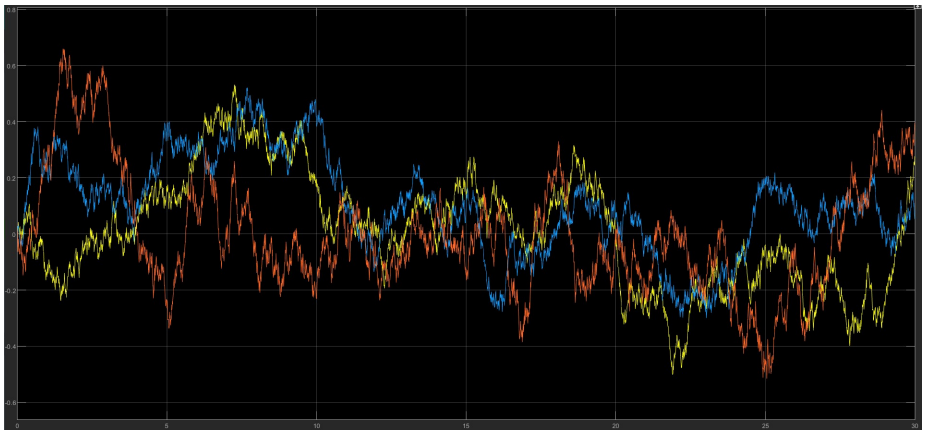

Figure 4.4: Wind disturbances about the Euler angles accelerations

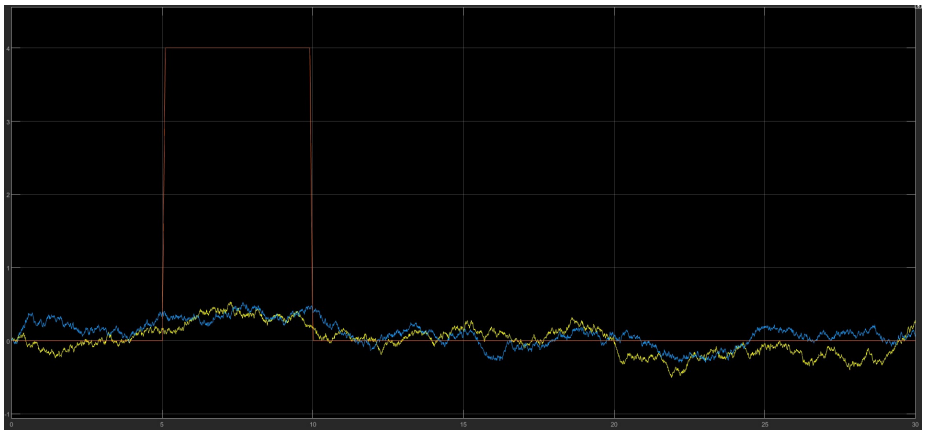

Figure 4.5: Wind disturbances about the position accelerations

The following pictures represent the trajectories performed by the aircraft under disturbances, blue lines, and commanded trajectories, yellow lines (careful attention must be paid to the scale of the graphs):

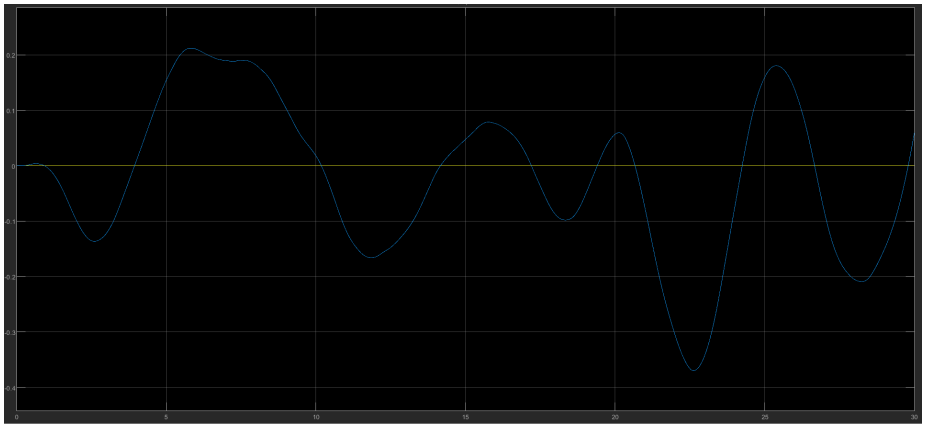

Figure 4.6: x-component trajectories

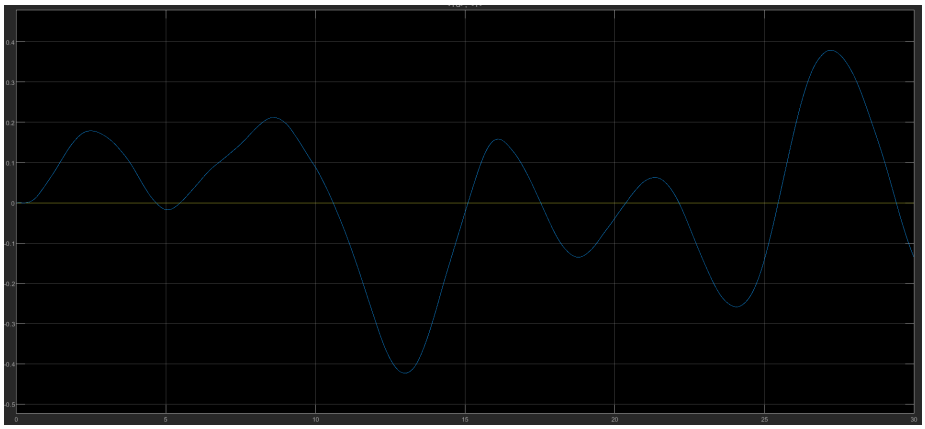

Figure 4.7: y-component trajectories

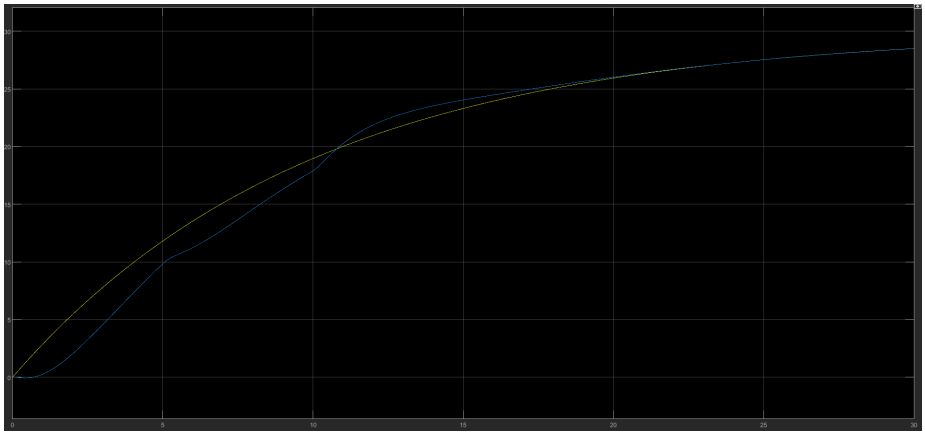

Figure 4.8: z-component trajectories

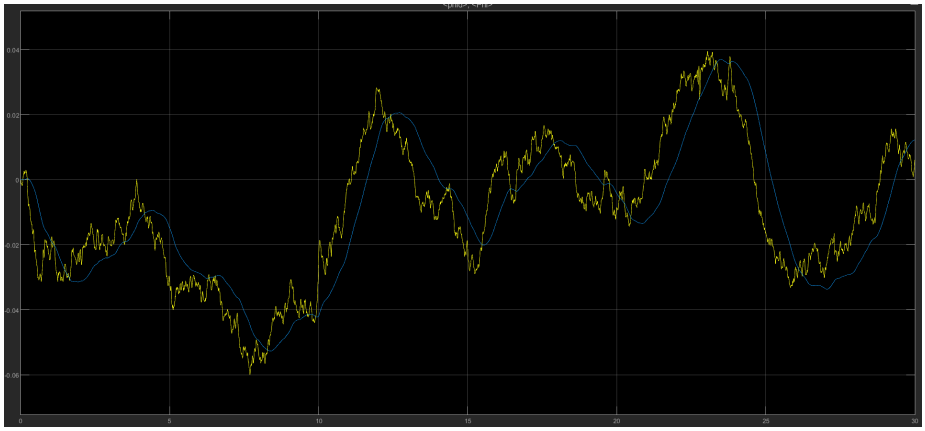

Figure 4.9: ϕ-component trajectories

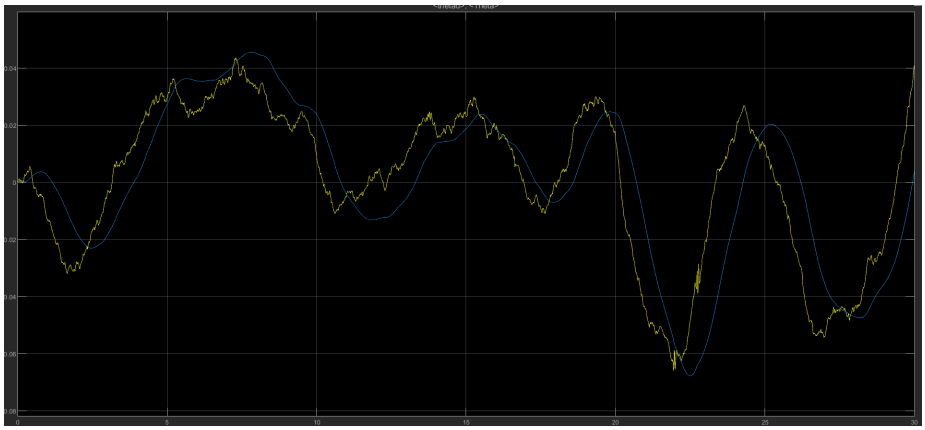

Figure 4.10: θ-component trajectories

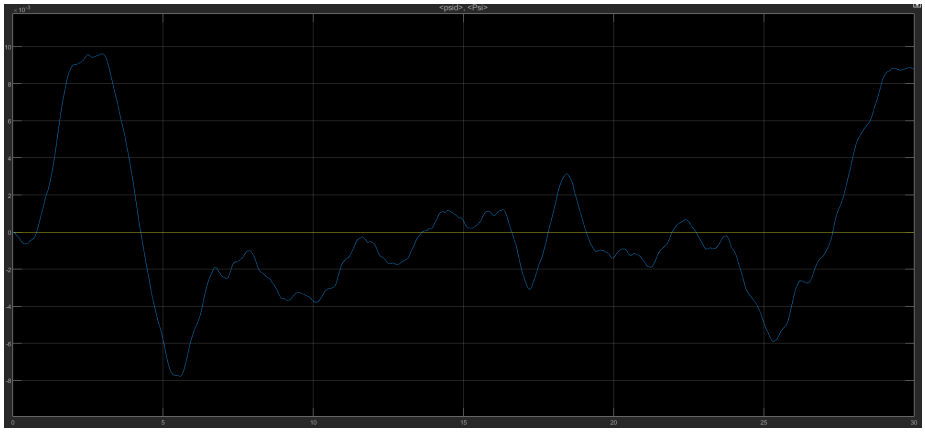

Figure 4.11: ψ-component trajectories

A commendable performance of the aircraft is observed, as it adeptly tracks the commanded trajectories with remarkable accuracy, despite operating under challenging conditions, also under a really particular disturbance as the wind shear at $t = 5s$ that could be noticed how the system not only not collapse but react really well. Meticulous parameter tuning yields enhanced aircraft behavior. It is worth noting that this simulator does not employ parameter optimization procedures.

4.2 Performance analysis of the controller applied to the simulation model

This represents the comprehensive controller capable of operating in both VTOL and FW conditions, incorporating a model for the actuators. Parameter scheduling was indispensable to achieve a smooth trajectory since take-off and landing phases are such distinct maneuvers. Both parameter sets were obtained through

a Particle Swarm Optimization (PSO) procedure. The following are the seven waypoints that the aircraft must reach, as the high-level controller autonomously manages the landing and take-off maneuvers references:

n°	x_d	y_d	z_d
1	3000	0	200
2	6000	3000	300
3	8000	6000	200
4	8500	7000	200
5	9000	8000	100
6	9500	9000	100
7	11000	10000	0

The entire mission is reported:

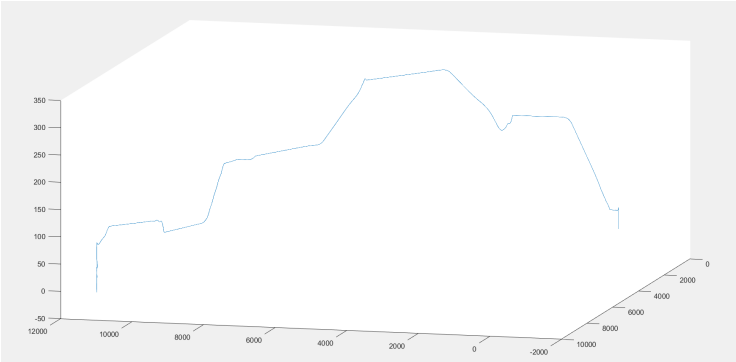

Figure 4.12: Complete mission in 3D environment

Take off

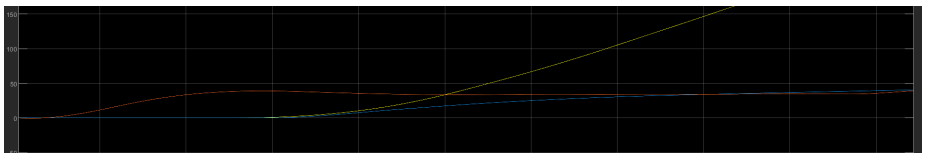

Figure 4.13: Take off and transition performed

The take-off maneuver involves a vertical ascent to an altitude of 35 m, followed by the transition phase characterized by a gradual acceleration towards the first checkpoint direction. As depicted in image 4.13, this operation requires 4 seconds to complete, i.e. to reach the altitude for the transition. Notably, the aircraft demonstrates exemplary tracking of the vertical and stationary trajectory with respect to the x and y axes. Compared to the previous controller, which required 11 seconds for the same task, the current controller exhibits significantly improved efficiency. During the transition phase, the most critical one, it is observed that about the z-component 4.8, the aircraft maintains stability about

the constant altitude of 35 m. However, in the y-direction (blue line), a slight overshoot during the transition, compared to the characteristic mission lengths, is noted. This deviation stems from the references commanded by the state flow, which are relatively inflexible to the operational requirements. Overall, while this block demonstrates satisfactory performance, it also presents certain limitations. Consequently, further investigation into alternative methodologies is warranted to enhance its capabilities. As previously mentioned, the model was assessed without wind, as it was determined that wind disturbances could only be effectively mitigated during the take-off phase. It is unrealistic to assume that wind solely affects a single phase of the flight. Nevertheless, a take-off scenario with wind disturbances is proposed to showcase the controller's capabilities within the comprehensive simulation model. This demonstration aims to further highlight the controller's effectiveness in handling various operational conditions.:

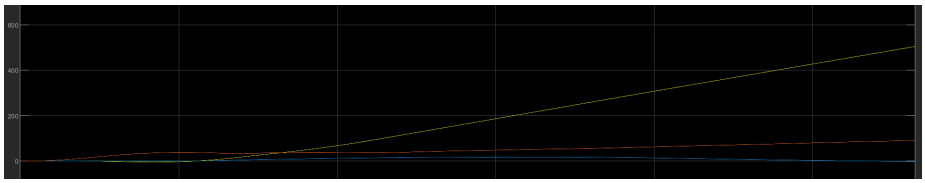

Figure 4.14: Take off and transition performed in the presence of wind

Landing

The most critical phase is the landing process, which demands considerable time investment to determine optimal references and parameters. Moreover, the Particle Swarm Optimization, both handwritten and built-in in Matlab, encountered significant challenges in identifying a suitable parameter set. However, through persistent efforts, an acceptable parameter set was eventually obtained. The transition is performed at a constant altitude and consists of a gradual deceleration (the entire control system governs it adjusting the trim and the throttles combining both FW and VTOL). Regarding altitude, it is observed that at $t = 560$ 4.16, the aircraft experiences an elevation increase. Upon examining the executed phases, it becomes apparent that this elevation surge coincides with the initiation of VTOL operations and the commencement of the transition phase.

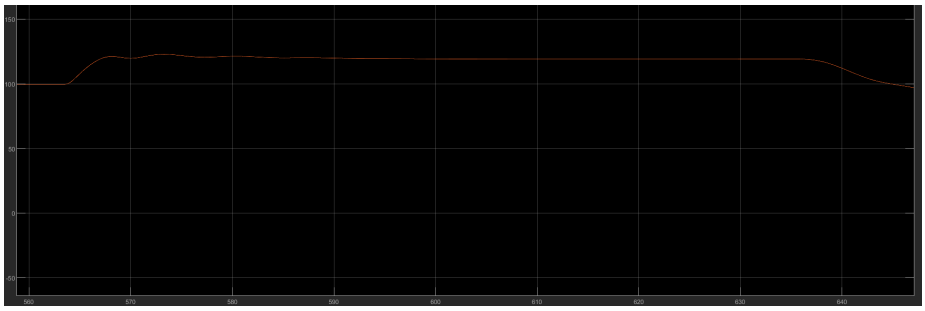

Figure 4.15: Increase in altitude due to transition from FW to VTOL

Subsequently, during a phase characterized by velocity reduction, the landing process initiates. Although not entirely seamless, the landing procedure proves efficient. This efficiency is attributed to the maintenance of the $x_d = 11000$, picture 4.17, and $y_y = 10000$, picture4.18, components, which remain relatively constant in proximity to the commanded position combined with a smooth descent.

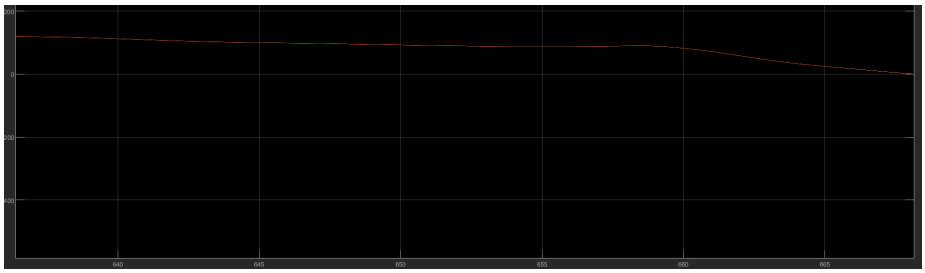

Figure 4.16: z trajectory during landing

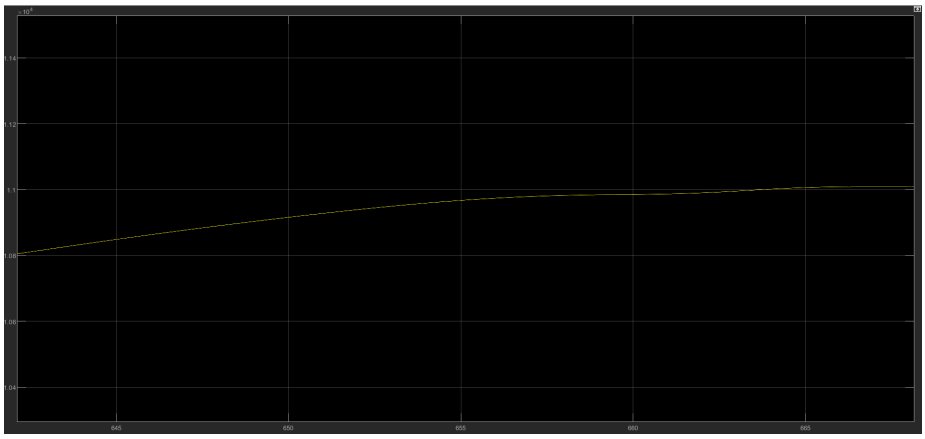

Figure 4.17: x trajectory during landing

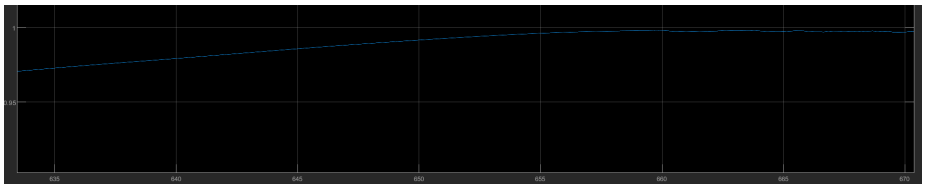

Figure 4.18: y trajectory during landing

It has also been requested to examine the impact of a counter-rotating motor, specifically the electric one located at the front, which effectively generates a negative thrust, albeit with a slight compromise in aerodynamic efficiency. Its operational logic is relatively straightforward: during the transition phase from fixed-wing flight to landing, it operates with a stationary throttle, in particular

during the testing campaign, it has been recognized an upper limit of $\delta_{th} = 0.5$. We could see from the following pictures that the aircraft needs less time to do the transition and landing phases,aproximately 15 s lesss, with mostly the same performances about the vertical descent, the error increase but the trajectory is maintained, further developments are needed:

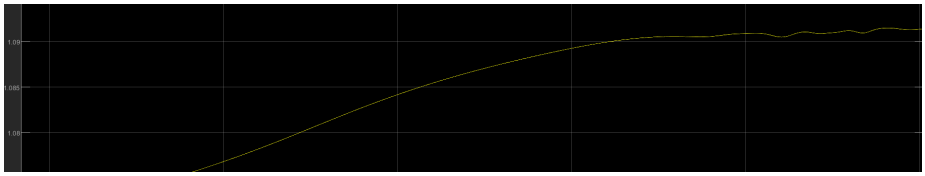

Figure 4.19: x trajectory during landing with breaking rotor

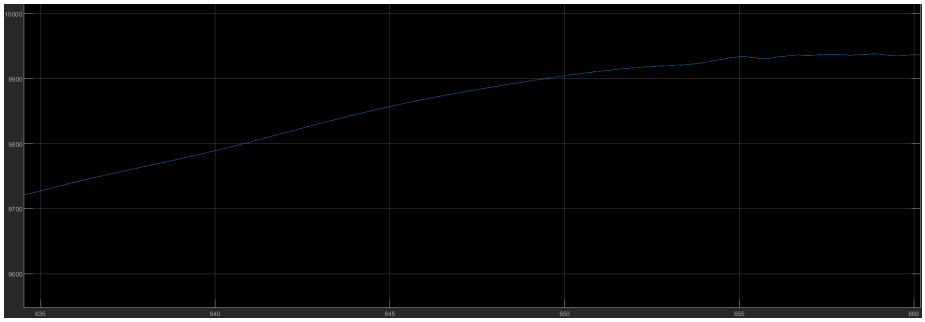

Figure 4.20: y trajectory during landing with breaking rotor

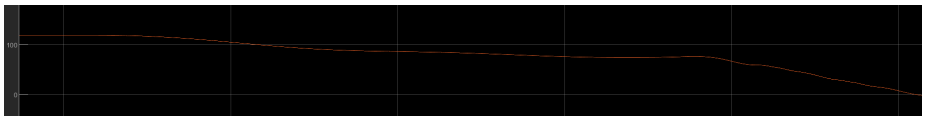

Figure 4.21: z trajectory during landing with breaking rotor

Chapter 5

Conclusions and future developments

As demonstrated in the previous chapter, the primary objective of developing and implementing an advanced controller for the VTOL segment within the complete simulation model has been achieved. This controller has exhibited reliability in trajectory tracking and disturbance rejection, showcasing optimal performance. However, there is room for enhancement through the adoption of a more sophisticated methodology for generating references. The counter-rotating rotor works since we can recognize a faster reduction in velocity, this implies less discharge of the VTOL batteries, is an immature technology, but with a good perspective.

Numerous aspects require further improvement: a primary focus could involve implementing a technique to adaptively update the tuning parameters of the outer loop. This could be achieved through methodologies such as fuzzy logic or by introducing a new outer loop based on approaches akin to those utilized in the inner loop. Regarding the inner loop, exploration of various learning techniques and assessing their performance differences, such as reinforced learning algorithms, presents a viable avenue for development. Additionally, attention could be directed towards enhancing the High-level controller segment, as the current employment of Stateflow appears to lack flexibility and poses challenges in proper tuning. An alternative approach could involve integrating a neural network that autonomously determines the different phases and references.

Chapter 6

Appendix

6.1 Appendix A

REFERENCE MODEL
For the reference model, the 6 degree of freedom equations of motion are used. We will approach the two main separated subsystems, and how to identify the main components of each one.

6.1.1 Rotational dynamics

$$\bar{M}_b = I\dot{\bar{\omega}} + \bar{\omega} \wedge (I\bar{\omega}) \tag{6.1}$$

They are the same of the ones used in 2.2, and the same would be for the translational dynamics, but the main differences would be that we will specialize changing the frame of reference and obtaining different forms in particular for the B component. Instead of the body's fixed angular velocity is necessary to express it in the Euler rate of change form. The transformation between the two is the sequent:

$$\bar{\omega} = J\bar{E} \tag{6.2}$$

Where the matrix J is equal to:

$$J = \begin{bmatrix} 1 & 0 & -sin(\theta) \\ 0 & cos(\phi) & sin(\phi)cos(\theta) \\ 0 & -sin(\phi) & cos(\phi)cos(\theta) \end{bmatrix} \tag{6.3}$$

But we need also the transformation about the accelerations about the euler angles that is obtained deriving 6.2, in matrix form:

$$\dot{\bar{\omega}} = J\ \ddot{\bar{E}} + L \tag{6.4}$$

Where L is a column vector:

$$L = \begin{bmatrix} -cos(\theta)\dot{\theta}\dot{\psi} \\ -sin(\phi)\dot{\phi}\dot{\theta} - sin(\theta)sin(\phi)\dot{\theta}\dot{\psi} + cos(\phi)cos(\theta)\dot{\phi}\dot{\psi} \\ -cos(\phi)\dot{\phi}\dot{\theta} - sin(\phi)cos(\theta)\dot{\phi}\dot{\psi} - cos(\phi)sin(\theta)\dot{\theta}\dot{\psi} \end{bmatrix} \tag{6.5}$$

Through some mathematics passages and substituting the previously obtained relationship in equation 6.2, the final requested form is presented with the main passages:

$$\begin{aligned} &I\dot{\bar{\omega}} = \bar{M}_b - \bar{\omega} \wedge I\bar{\omega} \\ &\dot{\bar{\omega}} = I^{-1}(\bar{M}_b - \bar{\omega} \wedge I\bar{\omega}) \\ &J\ \ddot{\bar{E}} + L = I^{-1}(\bar{M}_b - J\bar{E} \wedge IJ\bar{E}) \\ &J\ \ddot{\bar{E}} = I^{-1}(\bar{M}_b - J\bar{E} \wedge IJ\bar{E}) - L \\ &\ddot{\bar{E}} = J^{-1}(I^{-1}(\bar{M}_b - J\bar{E} \wedge IJ\bar{E}) - L) \end{aligned} \tag{6.6}$$

Where:

- $F(\chi)_{rot} = -J^{-1}(I^{-1}(J\bar{E} \wedge IJ\bar{E}) + L)$
- $B(\chi, u)_{rot} = J^{-1}I^{-1}\bar{M}_b$

In particular $B(\chi, u)_{rot}$, the same for the rotational part would be done, must be obtained in matrix form, instead of $F(\chi)$ that could remain in a column vector form, however, the commanded variables are not the moments and we need to search some connections with the thrusts. Looking at the steady values we found out out that there is a linear dependence, with k a constant experimentally obtained so:

$$M_i = kT_i \quad i = 1, 2, 3, 4 \tag{6.7}$$

Recalling matrix 2.7 and subsituting the previous relationship:

$$\bar{M}_b = \begin{bmatrix} -bcos(\beta) & -bcos(\beta) & bcos(\beta) & bcos(\beta) \\ bsin(\beta) & -bsin(\beta) & bsin(\beta) & -bsin(\beta) \\ k & -k & -k & k \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{6.8}$$

$$\bar{M}_b = N\bar{T} \tag{6.9}$$

In the end we obtain:

$$B(\chi, u)_{rot} = J^{-1}I^{-1}N\bar{T} \tag{6.10}$$

With the requested matrix ft the control system:

$$B(\chi)_{rot} = J^{-1}I^{-1}N \tag{6.11}$$

6.1.2 Translational dynamics

A quite similar process could be done for the translational dynamics so we start from the equation 2.1:

$$\bar{F}_b = m(\dot{V}_b + \bar{\omega} \wedge V_b) \tag{6.12}$$

After we making the conversion to Euler angles we will rearrange the equation in the following way:

$$\begin{aligned} &\bar{F}_b = m(\dot{V}_b + \bar{\omega} \wedge V_b) \\ &\dot{V}_b = \tfrac{1}{m}\bar{F}_b - \bar{\omega} \wedge V_b \\ &\dot{V}_b = \tfrac{1}{m}\bar{F}_b - J\bar{E} \wedge V_b \end{aligned} \tag{6.13}$$

We find effortlessly $F(\chi)_{trans}$:

$$F(\chi)_{trans} = -J\bar{E} \wedge V_b \tag{6.14}$$

And with a little of mathematics:

$$B(\chi,u)=\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m} & -\frac{1}{m} & -\frac{1}{m} & -\frac{1}{m} \end{bmatrix}\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}=B(\chi)_{trans}\bar{T} \tag{6.15}$$

6.2 Appendix B

LYANUPOV STABILITY THEOREM

The necessity of studying the stability of a nonlinear system poses a significant challenge for engineering purposes, as unstable systems are unuseful in practice. Various approaches proposed by Lyapunov can be considered: the first, known as the indirect method, involves linearization leading to transfer functions, SISO systems (where SISO stands for single input single output), etc. However, this method only yields local stability with small stability regions, which may not be suitable for certain cases. The direct method, employed in this work, addresses the stabilization of nonlinear systems directly.

The general concept is that if the total energy continuously dissipates, the system will eventually reach a stable state and remain there. This involves two main steps: firstly, selecting a suitable scalar function known as the Lyapunov function, and secondly, evaluating the first derivative of this function along the system trajectory. If this derivative is negative, the energy of the system is also decreasing, indicating stability.

Thus far, we will refer to a nonlinear system in the form:

$$\begin{aligned} &\dot{x}=f(x,t)\ for\ t\geq 0 \\ &x(t_0)=x_0\ for\ t_0\geq 0 \end{aligned} \tag{6.16}$$

where $x\in\Re^n$, f is a continuous nonlinear function that satisfies the standard condition of existence and uniqueness of solution, $t\in\Re^+$. A nonlinear system is called autonomous if is not dependent on time explicitly. So could be written in the form:

$$\dot{x}=f(x) \tag{6.17}$$

Another term for this concept is a time-invariant system. Selecting a specific state x^* defines an equilibrium point of the system if $f(x^*)=0$. Intuitively, an equilibrium point is deemed locally stable if all solutions originating near x^* (indicating initial conditions within a neighborhood of x^*) remain bounded over time near x^*. The equilibrium point x^* is considered locally asymptotically stable if x^* is locally stable and, additionally, all solutions starting near x^* tend towards x^* as $t\to\infty$. Let's introduce some theorems to elucidate the Lyapunov direct method:

Definition 1 (Lipchitz condition for metric spaces)**.** *Given two metric spaces (X,d_x) and (Y,d_y) where d_x denotes the metric on the set X and d_y the metric on the set Y, a function $f:X\to Y$ is called Lipchitz continuous if there exists a real constant $K\geq 0$, such that, for all x_2 and x_1 in X:*

$$d_y(f(x_1),f(x_2))\leq Kd_x(x_1,x_2) \tag{6.18}$$

where K is referred to as a Lipchitz constant for the function f. We can refer to f as to be $K - Lipchitz$.

Definition 2 (Lipchitz condition for normed spaces)**.** *A function $f : \Omega \subseteq \Re^n \rightarrow \Re^m$ is Lipchitz on Ω if and only if there exists a costant $K \leq 0$ such that, $\forall x_1, x_2 \in \Omega$, $x_1 \neq x_2$:*

$$\frac{d_y(f(x_1), f(x_2))}{d_x(x_1, x_2)} \leq K \tag{6.19}$$

Theorem 1 (Lyanupov direct method stability theorem)**.** *Let $x = 0$ be an equilibrium point for the time-invariant system $\dot{x} = f(x)$, where $f : U \rightarrow \Re^n$ is locally Lipchitz, $U \subset \Re^n$ a domain that contains the origin. Let $V : U \rightarrow \Re$ be a continuously differentiable, positive definite function in U:*

1. *if $\dot{V}(x)$ is negative semidefinite, then $x = 0$ is a stable equilibrium point.*
2. *if $\dot{V}(x)$ is negative definite, then $x = 0$ is an asymptotically stable equilibrium point.*

In both cases above, V is called a Lyanupov function. Moreover, if the condition holds for all $x \in \Re^n$ and $||x|| \rightarrow \infty$ implies that $V(x) \rightarrow \infty$, then $x = 0$ is globally stable in case 1 and globally asymptotic stable in case 2. Proofs are omitted. The Lyapunov approach is a useful method since it allows to assess the stability of equilibrium points of a system without solving the differential equations that describe the system. However, there is no generally applicable method for finding Lyapunov functions. Trial and error and mathematical/-physical insight are often used. Uniform stability is a concept that guarantees that the equilibrium point is not losing stability.

Theorem 2 (Uniform stability)**.** *Let $x = 0$ be an equilibrium point for a dynamic system nonautonomous and $U \subset \Re$ a domain containing it. Let $V : U \times [0, \infty] \rightarrow \Re$ be a continuously differentiable function that satisfies:*

$$W_1(x) \leq V(x, t) \leq W_2(x) \tag{6.20}$$

$$\dot{V}(x, t) = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(x, t) \leq -W_3(x) \tag{6.21}$$

for all $t \geq t_0$, $x \epsilon U$, where $W_1(x)$,$W_2(x)$ and $W_3(x)$ are continuous positive definite functions on U. Then, $x = 0$ is uniformly asymptotically stable and V is called a Lyapunov function. Furthermore, if $W_3(x) = 0$ then $x = 0$ is uniformly stable.

Corollary 1 (Sufficient condition for global uniform asymptotic stability for nonlinear autonomous system)**.** *Suppose that the assumptions of Theorem 2 hold for all $x \in \Re^n$ and $W_1(x) \rightarrow \infty$ for $||x|| \rightarrow \infty$, then $x = 0$ is globally uniformly asymptotically stable.*

Corollary 2 (Sufficient condition for global exponential stability for non autonomous system)**.** *Suppose that the assumptions of theorem 2 are replaced by:*

$$c_1||x||^q \leq V(x, t) \leq c_2||x||^q \tag{6.22}$$

$$\dot{V} \leq -c_3||x||^q \tag{6.23}$$

For some positive constants c_1, c_2, c_3 and q. Then $x = 0$ is exponentially stable. Furthermore, if the assumptions are satisfied for all $x \epsilon \Re^n$, then $x = 0$ is globally exponentially stable.

From Theorem 2, the function V is required to be dominated by a time-invariant function $W_2(x)$ This means that V is decrescent. Therefore, sufficient conditions for uniform stability and uniform asymptotic stability are achieved if there is a continuously differentiable, positive definite, and decrescent function V that satisfies the condition (10). We can also conclude that the origin is uniformly stable if $\dot{V}$ is negative semi-definite. If $\dot{V}$ is negative definite then the equilibrium point is uniformly asymptotically stable

6.3 Appendix C

PARTICLE SWARM OPTIMIZATION ALGORITHM

```
    function out=PSOf(problem,params)

%% Problem Definition

CostFunction = problem.CostFunction;              % Cost Function

nVar = problem.nVar;                              % Problem Dimension
VarSize=[1 nVar];

VarMin = problem.VarMin;                          % Min Value
VarMax = problem.VarMax;                          % Max Value

%% PSO parameter

MaxIt= params.MaxIt;                              % Maximum number ...
    of iteration

nPop = params.nPop;                               % Population Size

w = params.w;                                     % Inertia Moment
wdamp = params.wdamp;                             % Damping ratio of ...
    Inertia Coefficient
c1 = params.c1;                                   % Personal ...
    Acceleration Coefficient
c2 = params.c2;                                   % Social ...
    Acceleration Coefficient

%% Inizializzazione PSO

% Particle template
empty_particle.Position=[];
empty_particle.Velocity=[];
empty_particle.Cost=[];
empty_particle.Best.Position=[];
empty_particle.Best.Cost=[];

% Create Population Array
particle=repmat(empty_particle,nPop,1);

% Initialize Global Best
GlobalBest.Cost = inf;

% Initialize Population member
for i=1:nPop
```

```
    % Generate Random Solution
    particle(i).Position=unifrnd(VarMin,VarMax,VarSize);

    % Initialize Velocity
    particle(i).Velocity=zeros(VarSize);

    % Evaluation
    particle(i).Cost= CostFunction(particle(i).Position);

    % Update Personal Best
    particle(i).Best.Position=particle(i).Position;
    particle(i).Best.Cost=particle(i).Cost;

    % Update Global Best
    if particle(i).Best.Cost < GlobalBest.Cost
       GlobalBest=particle(i).Best;
    end

end

% Array to Hold Best Cost Value on Each Iteration
BestCosts = zeros (MaxIt,1);

%% Main Loop of PSO
it=0;
while it<(MaxIt) && GlobalBest.Cost>10^-4
    it=it+1;
    for i=1:nPop

        % Update Velocity
        particle(i).Velocity =  w * particle(i).Velocity...
                              + c1 * rand(VarSize) .* ...
                                  (particle(i).Best.Position - ...
                                  particle(i).Position)...
                              + c2 * rand(VarSize) .* ...
                                  (GlobalBest.Position - ...
                                  particle(i).Position);
        % Update Position
        particle(i).Position = particle(i).Position + ...
            particle(i).Velocity;

        % Evaluation
        particle(i).Cost = CostFunction(particle(i).Position);

        % Update Personal best
        if particle(i).Cost < particle(i).Best.Cost

            particle(i).Best.Position = particle(i).Position;
            particle(i).Best.Cost = particle(i).Cost;

            % Update Global Best
            if particle(i).Best.Cost < GlobalBest.Cost
               GlobalBest=particle(i).Best;
            end

        end

    end
```

```
    % Store the best Cost Value
    BestCosts(it) = GlobalBest.Cost;

    %Display Iteration information
    disp([' Iteration ' num2str(it) ': Best Cost= ' ...
        num2str(BestCosts(it))]);

    %Damping Inertia Coefficient
    w= w*wdamp;

end

out.pop = particle;
out.BestSol = GlobalBest;
out.BestCosts =BestCosts;

end
```

Bibliography

Thesis

[1] Simulation and Control of Quadplanes: a Particle Swarm and Model Predictive Control Approach, http://amslaurea.unibo.it/30538/

[2] Identificazione di modelli per la dinamica di un elicottero quadrirotore https://hdl.handle.net/10589/81082

Web sites

[3] 6DOF (Euler Angles) https://it.mathworks.com/help/aeroblks/6dofeulerangles.html

articles

[4] Julius A. Marshall, Wei Sun, Andrea L'Afflitto, A survey of guidance, navigation, and control systems for autonomous multi-rotor small unmanned aerial systems, Annual Reviews in Control, Volume 52, 2021, Pages 390-427, ISSN 1367-5788 https://doi.org/10.1016/j.arcontrol.2021.10.013 https://www.sciencedirect.com/science/article/pii/S1367578821000882

[5] Radial Basis Function Neural Networks: A Review ise.ncsu.edu/fuzzy-neural/wp-content/uploads/sites/9/2022/08/RBFNN.pdf

[6] A. Radke and Zhiqiang Gao, "A survey of state and disturbance observers for practitioners," 2006 American Control Conference, Minneapolis, MN, USA, 2006, pp. 6 pp.-, doi: 10.1109/ACC.2006.1657545.

[7] K. Y. Volyanskyy, W. M. Haddad and A. J. Calise, "A New Neuroadaptive Control Architecture for Nonlinear Uncertain Dynamical Systems: Beyond σ and ϵ -Modifications," in IEEE Transactions on Neural Networks, vol. 20, no. 11, pp. 1707-1723, Nov. 2009, doi: 10.1109/TNN.2009.2030748.

[8] A Review of Fundamentals of Lyapunov Theory The Journal of Applied Science [2011] Vol. 10 No. 2

[9] Wang, D., Tan, D., Liu, L. Particle swarm optimization algorithm: an overview. Soft Comput 22, 387–408 (2018). https://doi.org/10.1007/s00500-016-2474-6 https://doi.org/10.1007/s00500-016-2474-6

[10] van den Bergh F. An analysis of particle swarm optimizers. [Order No. 0804353]. University of Pretoria (South Africa); 2001.

[11] M. H. Marzaki, M. Tajjudin, M. H. F. Rahiman and R. Adnan, "Performance of FOPI with error filter based on controllers performance criterion (ISE, IAE and ITAE)," 2015 10th Asian Control Conference (ASCC), Kota Kinabalu, Malaysia, 2015, pp. 1-6, doi: 10.1109/ASCC.2015.7244851.

Books

[12] J.G. Leishman. Principles of Helicopter Aerodynamics. Cambridge University Press, 2006.

Ringraziamenti

In questi tre anni sono state molte le persone che mi sono state vicine e mi hanno aiutato e quindi se mi dovessi dimenticare qualcuno non me ne voglia male. Al Professor Paolo Castaldi, per il suo corso che ha costituito un punto di svolta fondamentale nel mio percorso accademico, nonché per l'opportunità unica offertami di svolgere un periodo di tirocinio, e per il costante supporto e orientamento forniti durante tutto il percorso. All'azienda Sky Eye Systems, e in particolare all'Ing. Lorenzo Franchi e all'Ing. Giuseppe Mattei, per l'opportunità fornita e per il sostegno pratico continuo, oltre ai preziosi feedback, durante il mio coinvolgimento nei progetti. Ringrazio la mia famiglia, forse la componente che più ha sentito oltre a me le difficoltà di questi anni, che sono filtrate, filtrano e filtreranno sempre e arriveranno sempre anche a loro, ma che nonostante ciò mi hanno sempre accompagnato e non hanno mai smesso di supportarmi con tanta pazienza in tutto e anche per avermi dato la possibilità sempre di seguire le mie aspirazioni e passioni. Ringrazio Elisabetta, ci siamo conosciuti solo negli ultimi due anni, ma devo dire che senza di te davvero non ce l'avrei mai fatta. Anche se con percorsi diversi abbiamo condiviso molto, mi hai ascoltato tanto, anche troppo, però sempre con attenzione e affetto e in te ho davvero trovato un'amica preziosa. Ringrazio Andrea, ci conosciamo ormai da molti anni, ma anche se il tempo passa, anche se distanti, anche se magari non ci siamo sentiti per molto, io so che sei sempre lì per me e che posso contare su di te. Ringrazio Ilaria e Martina, per le lunghe passeggiate, per essere sempre disposte ad ascoltarmi e per l'affetto che mi dimostrano sempre. Ringrazio i ragazzi della contrada, grazie a voi ho scoperto una passione che voglio continuare a coltivare, ma soprattutto delle persone da tenersi molto strette(sì, siete proprio voi: Simo, Fossa, Stefanino, Betta, Sofi, Catta, Sig, Said, Gabri, Edo, Energy, Zotto, Jacopo, Ricki, Thomas, Linda....). Ringrazio i miei educandi, siete i miei pargoli, e con molto affetto mi rendo conto che siete stati parte integrante del percorso degli ultimi anni. Ringrazio Giulia e Francesco, preziosi amici. Ringrazio Uto, Rosario e i compagni di corso, con voi ho condiviso poco tempo fisicamente assieme, ma il continuo confronto e il reciproco aiuto è di sicuro stato fondamentale e un punto di forza e probabilmente non sarei nella situazione attuale senza di loro.