Project 1 – Crazyflie Drone Modelling and Identification

Course: Unmanned Aerial Vehicles (UAVs)

M.Sc. in Aerospace Engineering

Professor: Bruno Guerreiro

April 2024

Group 3

José Duarte Dias Corvo, student number 67118

Rodrigo Miguel Santos Silva Pardela Veríssimo, student number 67133

Vasco Rafael da Ponte Luís Nunes, student number 67304

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Abstract

"This project aims at obtaining a realistic model of the Crazyflie 2.1 drone for simulation and control design purposes. In a first part, a physics first principles approach is used to obtain the model of the drone, while a second part uses an experimental data collection procedure to obtain real responses by the vehicle at the CybAer drone arena. This first project will pave the way for the subsequent two projects, aimed at sensor fusion and motion control for this vehicle" – Bruno Guerreiro, 14th March 2024

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Index

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

List of Figures

1. Introduction

As cited by Captain Brian Tice from United States Air Force[1], Unmanned Aerial Vehicles (or UAVs) are powered vehicles that do not carry human operators. These vehicles started to be developed on the twentieth century for military use, but have evolved into other uses like agriculture, entertainment or product deliveries.

Nowadays it's still a technology under intense development, where the safety and security regarding the usage of these vehicles in our society is a big concern. Operating an unmanned aerial vehicle is common when a human is in charge of the commands, the use of fully autonomous drones in complex and challenging environments is still underdeveloped and under intense research by universities and the relevant industries that could benefit from this.

In this first project (from a series of three projects), the group was designated to consider the modelling and identification of a micro aerial vehicle called Crazyflie 2.1.

Figure 1 – Crazyflie 2.1 by Bitcraze Store

A Micro Aerial Vehicle (or MAV) is a man-portable aerial vehicle which, by EASA's (European Union Aviation Safety Agency) definition, weighs less than 250 grams[2].

The coding resolution of this project is in the format of a GitHub repository after working on the software MATLAB R2021a.

2. Project 1 Subjects and Goals

In this first part of the project, our group will address the full nonlinear model of the Crazyflie drone in 3-D space, considering an ENU (East-North-Up) local tangent plane centred at the drone area of the CybAer laboratory in FCT-UNL (Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa), with the respective coordinates (Latitude 38.660319ºN, Longitude -9.204972ºW) as the inertial frame. It will be assumed that the drone is a rigid body in 3-D space with the relevant external forces acting on the centre of mass of the drone. In order to simplify, it will also be assumed that the body frame is also located at the centre of mass of the drone, considering the constant of mass ("$\boldsymbol{m} = 0.032\ [kg]$", as mentioned by Professor Bruno Guerreiro during the class on the 21st of march 2024 of "Unmanned Autonomous Vehicles") and moment of inertia's matrix ("$\boldsymbol{J} = diag(J_x, J_y, J_z)$").

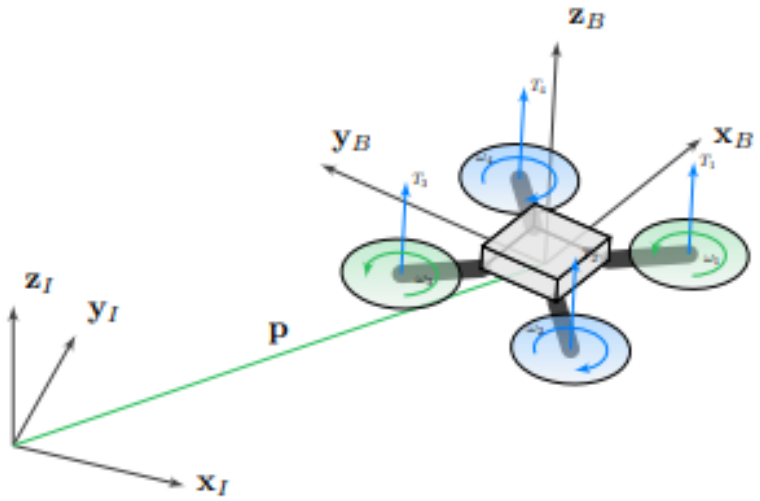

Figure 2 – Crazyflie 2.1 reference frames and configuration

In order to develop the project, the group will consider the following state-space variables of the drone:

- The drone's 3-D position (from the centre of mass "B" and origin of the body frame) relative to the inertial frame "I" ("$\boldsymbol{p} = {}^{I}p_B \in \mathbb{R}^3$");
- The velocity of the drone described on the body frame ("$\dot{\boldsymbol{p}} \in \mathbb{R}^3$");
- The attitude of the body frame regarding the inertial frame ("$\boldsymbol{R} \in SO(3)$"), parametrized by the $Z-Y-X$ Euler angles vector ("$\boldsymbol{\lambda} = [\phi \quad \theta \quad \psi]^T$");
- The relative angular velocity acting on the body frame ("$\boldsymbol{\omega} \in \mathbb{R}^3$").

Regarding the forces acting on the drone, it considered the Earth Gravity Acceleration ("$\boldsymbol{g_{earth}} = 9.82\ [{}^{m}/_{S}]$"), the Terrestrial Atmosphere Surface Density ("$\boldsymbol{\rho_{earth}} = 1.217\ \left[{}^{kg}/_{m^3}\right]$")[3], the sum of the four forces of propulsion generated by the rotors ("$\boldsymbol{F_p} = F_{p_1} + F_{p_2} + F_{p_3} + F_{p_4}$") and the sum of the four moment vectors of the propulsion generated by the rotors ("$\boldsymbol{n_p} = n_{p_1} + n_{p_2} + n_{p_3} + n_{p_4}$"). The maximum thrust needed by the sum of the forces of propulsion is obtained by the formula "$T_{maximum} = (m_{drone} + m_{maximum\ load}) \times g_{earth} = (0.032\ kg + 0.015\ kg) \times 9.82\ {}^{m}/_{S} = 0.46154\ N$".

The group, in order to obtain the most accurate simulation of the drone, verified the external dimensions of the Crazyflie 2.1.

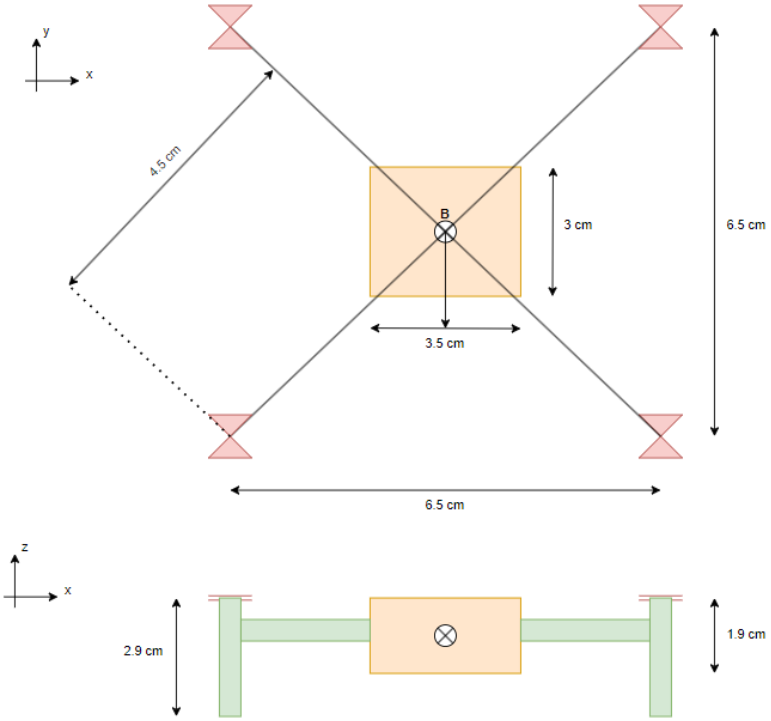

Figure 3 – Sketch displaying the dimensions of the Crazyflie 2.1 drone

Regarding the rotors, it can be assured that they have a vertical propulsion system (90º or $\frac{\pi}{2}\ rad$ regarding the axis angle xz).

The goals imposed for this part of the project are the following:

- **Goal number 1.1:** Obtain the rigid-body nonlinear model of the drone, considering the state variables above.
- **Goal number 1.2:** Obtain the Earth gravity force vector "$\boldsymbol{f_g}$" and the air drag force vector "$\boldsymbol{f_a}$" described at the body frame, considering the respective moments to be negligible.
- **Goal number 1.3:** Obtain the propulsion force vector "$\boldsymbol{f_p}$" and moment vector "$\boldsymbol{n_p}$" expressions at the vehicle centre of mass (origin of the body frame) as a function of the individual thrust forces generated by each of the four rotors "$\boldsymbol{u_T} = [T_1 \quad T_2 \quad T_3 \quad T_4]^T$".
- **Goal number 1.4:** Obtain the 4×4 matrix that maps the four individual forces of the rotors "u_T" into the control input vector "$\boldsymbol{u} = [T \quad n_p]^T$", where "$\boldsymbol{T} = T_1 + T_2 + T_3 + T_4$" is the total thrust of the drone and "$\boldsymbol{n_p} = [n_x \quad n_y \quad n_z]^T$" are the moments generated by the propulsion of the four rotors.
- **Goal number 1.5:** Obtain the final nonlinear model of the drone and simulate the model in MATLAB R2021a, considering initial resting state vector and small constant thrust equal to all rotors.

After developing the full nonlinear model of the vehicle, it will be possible to analyse the main modes of the drone individually. To simplify this action, it will be assumed a diagonal inertia matrix "$\boldsymbol{J} = diag(J_x, J_y, J_z)$" and the following operating modes and equilibrium points:

- **Operating mode 1:** The drone is hovering, having zero linear and angular velocity.
- **Operating mode 2:** The drone is on a horizontal flight, having zero vertical and angular velocity and a constant horizontal velocity.

Considering these conditions and that the perturbations to the equilibrium conditions can be defined for the state as "$\tilde{\boldsymbol{u}} = u - u_e$" and given as the same input, another set of goals were imposed to develop the operation mode:

- **Goal number 1.6:** Obtain the equilibrium conditions ("$\boldsymbol{x_e}$" and "$\boldsymbol{u_e}$") for each operation mode.
- **Goal number 1.7:** Obtain the linearized models ("$\dot{\tilde{\boldsymbol{x}}} = A_{OPi}\tilde{x} + B_{OPi}\tilde{u}$") around the equilibrium points and trajectories, considering $Z - Y - X$ Euler angles kinematics.
- **Goal number 1.8:** Decompose each linearized model into modes and discuss the results.
- **Goal number 1.9:** Analyse each linearized model regarding their controlability, observability and stability.
- **Goal number 1.10:** Deduce the transfer functions and state-space models related with the inner and outer dynamic loops defined by the following conditions:
  - "$\boldsymbol{G_{n_x,\phi}(s)} = {\phi(s)}/{n_x(s)}$";
  - "$\boldsymbol{G_{n_y,\theta}(s)} = {\theta(s)}/{n_y(s)}$";
  - "$\boldsymbol{G_{n_z,\psi}(s)} = {\psi(s)}/{n_z(s)}$";
  - "$\boldsymbol{G_{\phi,p_y}(s)} = {p_y(s)}/{\phi(s)}$";
  - "$\boldsymbol{G_{\theta,p_x}(s)} = {p_x(s)}/{\theta(s)}$";
  - "$\boldsymbol{G_{T,p_z}(s)} = {p_z(s)}/{T(s)}$".
- **Goal number 1.11:** Discuss the closed loop stability of the modes defined by "$\boldsymbol{G_{T,p_z}(s)}$" and "$\boldsymbol{G_{n_x,p_y}(s)}$", for instance, considering the root-locus arguments for a given control structure.

Building on the model analysis concluded, the group must identify parts of the Crazyflie 2.1 drone and compare it with the simulated model. Using a data set acquired with the drone featuring abrupt position changes.

For this section, the group will need to conclude another set of goals:

- **Goal number 2.1:** Analyse the data and divide it in three sets, considering motions in each of the position degrees of freedom (axis "x", "y" and "z").
- **Goal number 2.2:** Show the relation between the inputs and outputs of the transfer functions:

- "$\boldsymbol{G_{\theta,p_x}(s)} = {p_x(s)}/{\theta(s)}$";
- "$\boldsymbol{G_{\phi,p_y}(s)} = {p_y(s)}/{\phi(s)}$";
- "$\boldsymbol{G_{T,p_z}(s)} = {p_z(s)}/{T(s)}$".

- **Goal number 2.3:** Use the tools provided by MATLAB R2021a for model identification for each of these modes.
- **Goal number 2.4:** Compare the outputs of the models using the same inputs from the dataset.

3. Development of the goals imposed

In order to obtain the goals proposed by coding in MATLAB R2021a, the group based its research on the examples provided by Professor Bruno Guerreiro on the 2023/2024 version of FCT/UNL Moodle.

3.1. Goal number 1.1

Obtain the rigid-body nonlinear model of the drone, considering the state variables above.

In order to obtain the rigid-body nonlinear model of the drone, the group designated the state variables described above in the code. Afterwards, the nonlinear model is defined by the cinematic and multirotor dynamic equations.

The cinematic equations developed for the rigid-body nonlinear model are the following:

- For the derivative function of the position, the code developed referred to the equation "$\dot{p} = Rv$";
- The angular velocity regarding the Euler angles is defined by the function "$\dot{\lambda}(t) = Q(\lambda(t)\omega(t))$", where "$Q(\lambda) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{bmatrix}$" and the angular velocity "ω".

The multirotor dynamic equations developed for this model are the following:

- The acceleration is developed with "$\dot{v} = -S(\omega)v + \frac{1}{m} * (f_g + f_a + f_p)$", where "$S = skew$" and "$f_a$" is the air drag force on the body;
- The angular acceleration of the drone is defined by "$\dot{\omega} = -J^{-1}S(\omega)J\omega + J^{-1}n_p$".

3.2. Goal number 1.2

Obtain the Earth gravity force vector "$\boldsymbol{f_g}$" and the air drag force vector "$\boldsymbol{f_a}$" described at the body frame, considering the respective moments to be negligible.

The Earth's gravity force vector "f_g" is given through the development of the equation "$f_g = -gR^T z_I$", where the matrix of "R^T" is the axis change required to observe the gravity force, with this matrix being defined by the rotation of the inertial frame into the body frame.

Meanwhile, the air drag force vector "f_a" is given by the equation "$f_a = \left(-{}^1/_2\, \rho |v| A\right) v$", in order to develop this project the equation will be "$f_a = \left(-\frac{1}{2} \rho * cte * A\right) v$", where "$A = \begin{bmatrix} 3*1.9 & 0 & 0 \\ 0 & 3.5*1.9 & 0 \\ 0 & 0 & 3.5*3 \end{bmatrix} * 10^{-4}\ [m^2]$" is the surface area of the drone, with each value of the diagonal being the surface area of the drone perpendicular to the respective axis.

The parameter "$cte = 3\ [{}^m/_s]$" is equal to the velocity considered reasonable for the movement of the drone. This parameter has been established considering the estimated velocity of the drone in the simulation of real life and the linearization of "Horizontal Flight" with the equilibrium Euler Angles defined in order to obtain an equilibrium velocity close to the constant "cte", in order to obtain a representation of the system's linearized model as close to reality as possible. This value was estimated to be lower than the real of drag coefficient, in order to not overshoot the thrust and avoid crashing the drone. The reason why our group did this was due to difficulty to find the equilibrium points and to simplify complex mathematic operations, therefore also minimizing the error.

3.3. Goal number 1.3

Obtain the propulsion force vector "$\boldsymbol{f_p}$" and moment vector "$\boldsymbol{n_p}$" expressions at the vehicle centre of mass (origin of body frame) as a function of the individual thrust forces generated by each of the four rotors "$\boldsymbol{u_T} = [T_1 \quad T_2 \quad T_3 \quad T_4]^T$".

In order to obtain the propulsion force vector "f_p", it must be acknowledged that this vector is the sum of the propulsion force vectors of each rotor of the drone ("$f_{p_1}, f_{p_2}, f_{p_3}, f_{p_4}$"). Each one of these vectors has a null dimension on the axis x and y, while $z = 1$, representing the direction of the thrust as "$\begin{bmatrix}0\\0\\1\end{bmatrix}$". The propulsion force of each rotor is arbitrary considering the variation of the value of the total thrust. Considering the variation of each rotor thrust is between $0\ N$ and its maximum thrust (which can be obtained from the equation "$T_{max} = \frac{(m_{drone} + m_{maximum\ pay-load})}{number\ of\ rotors} * g_{earth} = \frac{(0.032+0.015)}{4} * 9.82 = 0.1154\ N$"), the propulsion force of each rotor given by the following multiplication: "$\begin{bmatrix}0\\0\\1\end{bmatrix} * T_i$", with "$i$" the identification of each rotor.

The moment vector "n_p" is defined by the position of each rotor relative to the centre of mass of the drone multiplied by the propulsion force of each rotor, relating each rotor with the direction and rotation of the rotors. To this multiplication it's added the moment generated by each rotor around the axis x ("$n_i = \begin{bmatrix}0\\0\\C_Q * {\omega_i}^2\end{bmatrix} = \begin{bmatrix}0\\0\\\frac{C_Q}{C_T} * (\pm T_i)\end{bmatrix} = \begin{bmatrix}0\\0\\\pm cte * T_i\end{bmatrix}$", where "$\omega_i$" is the angular velocity for each rotor, "C_Q" is the coefficient of drag of the fins[4] and "C_T" the coefficient of thrust, with these two coefficients being constants defined from experiment).

In order to define the moment vector for each rotor there was the development of the following equations:

- $n_{p_1} = n_1 + (pos_1 \times f_{p_1}) = \begin{bmatrix}0\\0\\cte.T_1\end{bmatrix} + \begin{bmatrix}3.25\\-3.25\\1.45\end{bmatrix} \times 10^{-2} \times f_{p_1}\ [N.m]$;
- $n_{p_2} = n_2 + (pos_2 \times f_{p_2}) = \begin{bmatrix}0\\0\\-cte.T_2\end{bmatrix} + \begin{bmatrix}-3.25\\-3.25\\1.45\end{bmatrix} \times 10^{-2} \times f_{p_2}\ [N.m]$;
- $n_{p_3} = n_3 + (pos_3 \times f_{p_3}) = \begin{bmatrix}0\\0\\cte.T_3\end{bmatrix} + \begin{bmatrix}-3.25\\3.25\\1.45\end{bmatrix} \times 10^{-2} \times f_{p_3}\ [N.m]$;
- $n_{p_4} = n_4 + (pos_4 \times f_{p_4}) = \begin{bmatrix}0\\0\\-cte.T_4\end{bmatrix} + \begin{bmatrix}3.25\\3.25\\1.45\end{bmatrix} \times 10^{-2} \times f_{p_4}\ [N.m]$.

The sum of the moment vectors of each rotor defines the moment generated by the drone ("$n_p = \sum n_{p_i} = n_{p_1} + n_{p_2} + n_{p_3} + n_{p_4}$").

3.4. Goal number 1.4

Obtain the 4×4 *matrix that maps the four individual forces of the rotors "*u_T*" into the control input vector "*$\boldsymbol{u} = [T \quad n_p]^T$*", where "*$\boldsymbol{T} = T_1 + T_2 + T_3 + T_4$*" is the total thrust of the drone and "*$\boldsymbol{n_p} = [n_x \quad n_y \quad n_z]^T$*" are the moments generated by the propulsion of the four rotors.*

The control input vector "u" is defined by the total thrust of the drone and the moments of the drone that are produced:

- $u = [Thrust \quad n_{p_x} \quad n_{p_y} \quad n_{p_z}]^T$.

For the non-linear model the control input vector defined was not in this form, but instead it was defined considering the variation of values of the thrust for each rotor, as showed in the following matrix:

- $$u = \begin{bmatrix} T_1 + T_2 + T_3 + T_4 \\ n_{p_{x_1}} + n_{p_{x_2}} + n_{p_{x_3}} + n_{p_{x_4}} \\ n_{p_{y_1}} + n_{p_{y_2}} + n_{p_{y_3}} + n_{p_{y_4}} \\ n_{p_{z_1}} + n_{p_{z_2}} + n_{p_{z_3}} + n_{p_{z_4}} \end{bmatrix} \Leftrightarrow$$

$$\Leftrightarrow u = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -3.25 & -3.25 & 3.25 & 3.25 \\ -3.25 & 3.25 & 3.25 & -3.25 \\ -cte & cte & -cte & cte \end{bmatrix} \times 10^{-2} * \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}.$$

3.5. Goal number 1.5

Obtain the final nonlinear model of the drone and simulate the model in MATLAB R2021a, considering initial resting state vector and small constant thrust equal to all rotors.

In order to obtain the final nonlinear model of the drone, the initial conditions of the simulation were assumed as:

- $p(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [m]$;
- $\lambda(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [rad]$;
- $v(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [{}^{m}/_{s}]$;
- $\omega(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \left[{}^{rad}/_{s}\right]$.

As soon as the simulation started, maximum thrust is applied to each rotor, supposedly creating a vertical movement with vertical velocity.

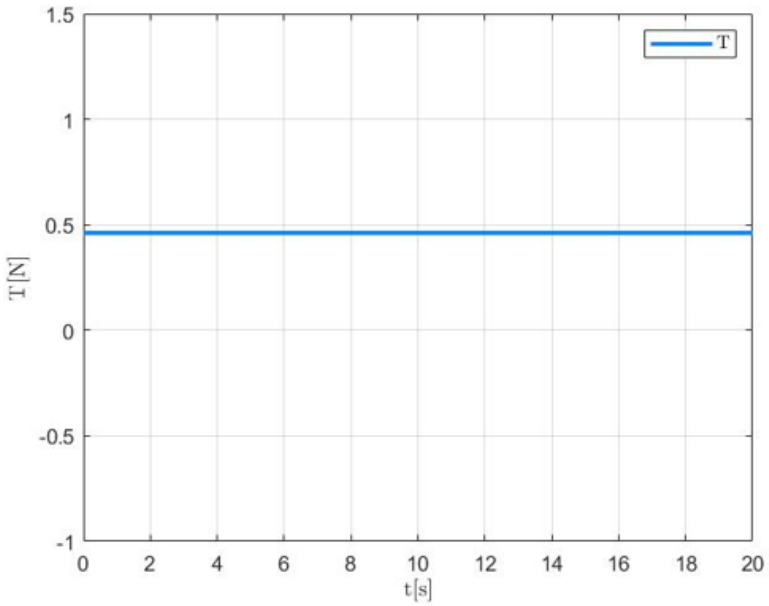

Figure 4 – Total Thrust ("$T[N]$") over Time ("$t[s]$") on the nonlinear model

The maximum thrust is constant during the whole duration of the simulation.

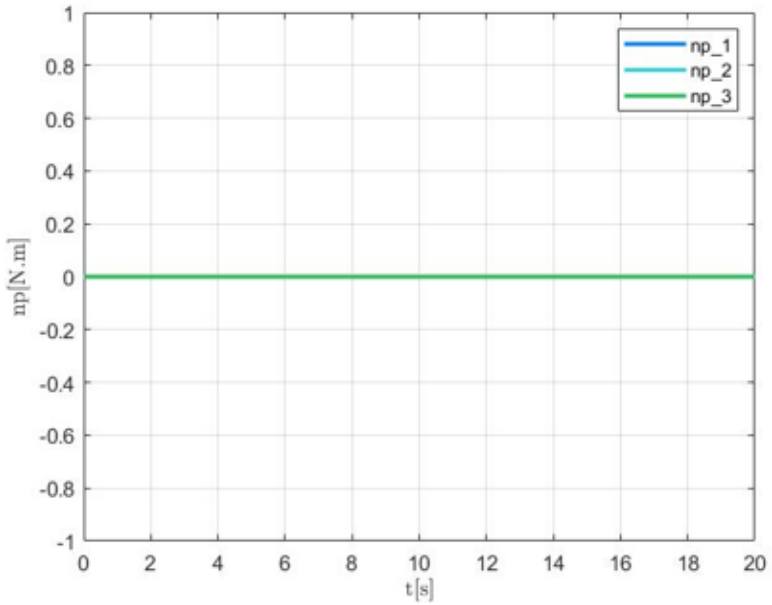

Figure 5 – Moment generated by the drone ("$np[N.m]$") over Time ("$t[s]$") on the nonlinear model

The moment generated by the drone is null during the whole duration of the simulation, due to the fact that there's only movement in axis z, with all rotors developing the same thrust cancelling the moments for each other.

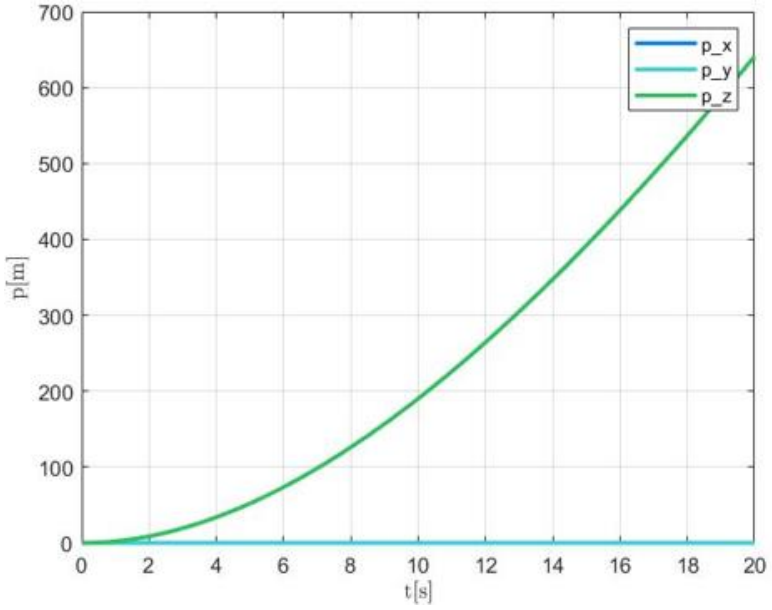

Figure 6 – Position ("$p[m]$") over Time ("$t[s]$") on the nonlinear model

With the rotors developing maximum thrust, the position in z is growing in an exponential way, developing more and more velocity while its acceleration decreasing, converging to close to null acceleration.

The drag is an important factor in the parameter of position, making it hard to develop a realistic simulation in consideration of the drag.

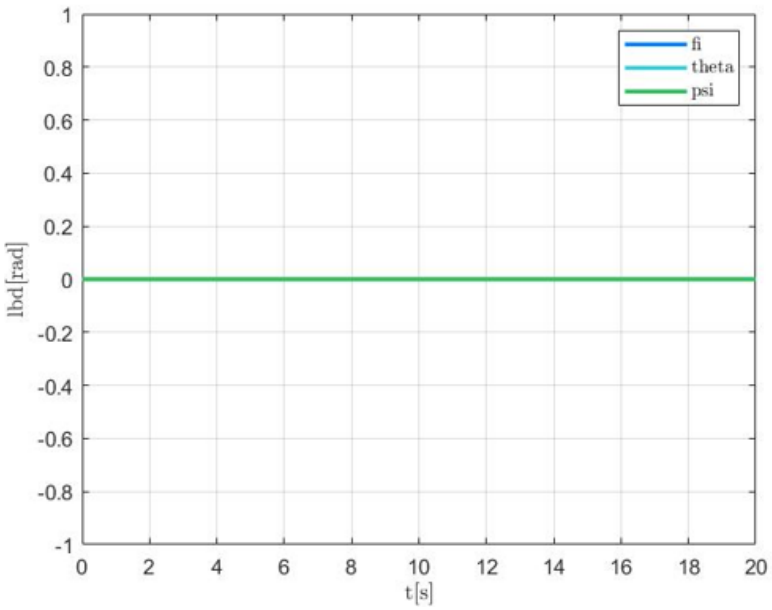

Figure 7 – Euler Angles ("$lbd[rad]$") over Time ("$t[s]$") on the nonlinear model

The Euler angles are null during the whole duration of the simulation since there are no variations on other axis besides the axis z and because there's no rotation of the drone.

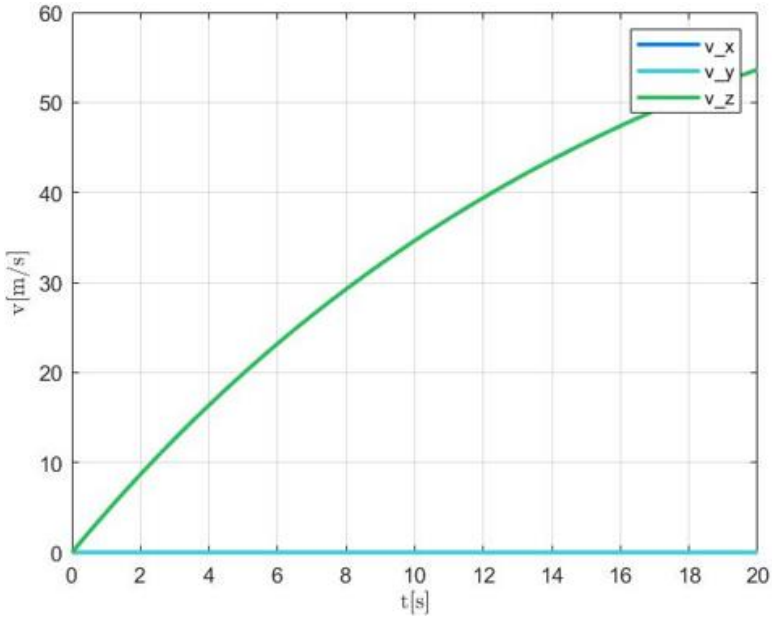

Figure 8 – Velocity ("$v[m/s]$") over Time ("$t[s]$") on the nonlinear model

The velocity grows in a logarithmic way due to the fact that the drag forces grow when the velocity grows, making it harder to increase the speed of the drone, thus preventing the growth of velocity in a linear way.

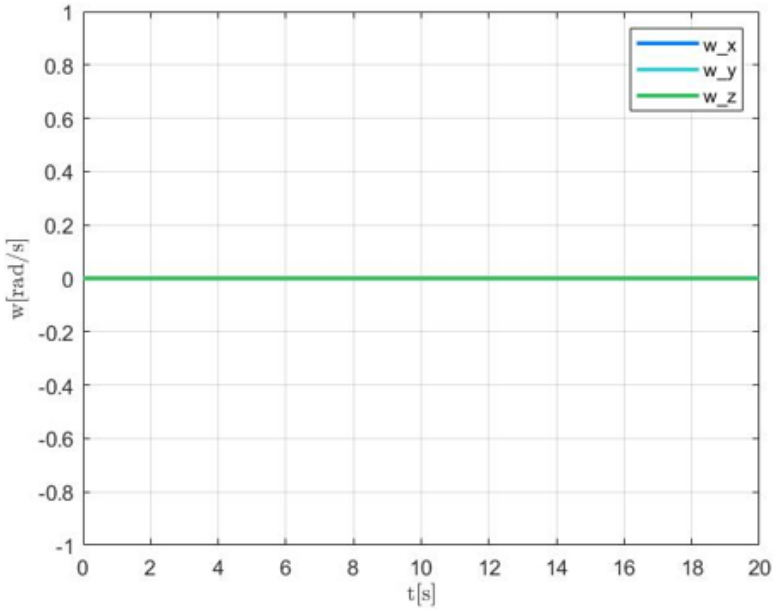

Figure 9 – Angular Velocity ("$w[rad/s]$") over Time ("$t[s]$") on the nonlinear model

There's no variation in the angular velocity of the drone since there is no variation in the moments of the drone.

3.6. Goal number 1.6

Obtain the equilibrium conditions ("$\boldsymbol{x_e}$" and "$\boldsymbol{u_e}$") for each operation mode.

In order to progress in this project, the group need to define the thrust used in order to sustain a stable position. This thrust will be constant and without change the position parameters of the drone.

Developing the system into two different modes ("Hovering" and "Horizontal Flight"), the group needs to calculate different variables that define the equilibrium of the drone.

For the first operating mode ("Hovering"), there's no change in the position of the drone, with null velocity and null angular velocity.

- $v_{eq} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} {}^{m}/_{s}$;
- $\omega_{eq} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} {}^{rad}/_{s}$.

In order to obtain the cinematic equilibrium of this mode, then following equations were developed:

- $\dot{\lambda}_{eq}(t) = Q(\lambda_{eq})\omega_{eq} = 0 \Leftrightarrow \dot{\lambda}_{eq}(t) = \lambda_{eq}(t=0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$;
- $\dot{p}_{eq}(t) = R(\lambda_{eq})v_{eq} = \int v_{eq}dt + cte = v_{eq}t + cte \Leftrightarrow \dot{p}_{eq}(t) = p_{eq}(t=0)$.

It's also important to obtain the equilibrium dynamic points of this mode, which come from the following equations:

- $\dot{v}_{eq}(t) = -S(\omega_{eq})v_{eq} + g * R^T * \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} + \frac{f_a}{m} + \frac{f_{p_{eq}}}{m} \Leftrightarrow$

$\Leftrightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{T_{eq}}{m} \end{bmatrix} \Leftrightarrow \boldsymbol{T_{eq}} = mg$;

- $\dot{w}_{eq}(t) = J^{-1}S(\omega_{eq})J\omega_{eq} + J^{-1}n_{p_{eq}} \Leftrightarrow 0 = J^{-1}n_{p_{eq}} \Leftrightarrow \boldsymbol{n_{p_{eq}}} = 0$.

Knowing that "$n_{p_{eq}} = 0$", it can be assumed that the thrust of each rotor will be similar for all, concluding that, while in "Hovering", the thrust of each rotor is equal to the equivalent thrust divided by every rotor ("$T_{eq} = T_1 + T_2 + T_3 + T_4$" and "$T_1 = T_2 = T_3 = T_4 = T_{eq}$").

In order to develop the second operating mode ("Horizontal Flight"), it needs to be described as a constant horizontal movement along the inertial axis x and y and null velocity along the inertial axis z.

To guarantee “Horizontal Flight”:

- $\omega_{eq} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} {}^{rad}/_{s}$;
- ${}^{I}v_{eq} = \begin{bmatrix} {}^{I}v_{x_{eq}} \\ {}^{I}v_{y_{eq}} \\ 0 \end{bmatrix} {}^{m}/_{s}$.

The horizontal movement described means that the position is not constant, meaning that there’s no need for an equilibrium point for it, but rather an equilibrium trajectory (which doesn’t influence the calculation of matrices “A” and “B”).

- $\dot{p}_{eq}(t) = R(\lambda_{eq})v_{eq} \Leftrightarrow \dot{p}_{eq}(t) = {}^{I}v_{eq} \Leftrightarrow p_{eq}(t) = {}^{I}v_{eq}t + p_{eq}(t = 0)$.

Since the equilibrium trajectory doesn’t influence the calculation of linearized matrices “A” and “B”, “$p_{eq}(t = 0)$” is considered to be equal to any value as long as it is equal to the initial position when using the linearized model referred.

As already mentioned, in this operating mode the drone will have a constant inclination (relative to the surface, therefore also relative to the inertial frame), and given this inclination the drone will obtain a certain speed. Exemplifying this phenomenon, for a constant inertial velocity “${}^{I}v_{z}$”, the higher roll (“ϕ”) that the drone has, the higher the module of velocity will be along the inertial axis y. If the pitch is elevated (“θ”), a higher velocity along the inertial axis x will be obtained. However, it should be noted that yaw (“ψ”) does not interfere with any of the drone’s inertial velocities.

- $\dot{\lambda}_{eq}(t) = Q(\lambda_{eq})\omega_{eq} = 0 \Leftrightarrow \lambda_{eq} = \lambda_{eq}(0)$.

As it’s not wanted to impose relatively high velocities to the vehicle, the Euler angles at the equilibrium point were defined as the following:

- $\boldsymbol{\lambda_{eq}} = \lambda_{eq}(0) = \begin{bmatrix} \frac{\pi}{360} \\ \frac{\pi}{360} \\ 0 \end{bmatrix} rad.$

Knowing that the drone at the point of equilibrium defined has “${}^{I}v_{z_{eq}} = 0$” and the Euler angles relevant to this matter, it’s possible to deduce the inertial velocities associated with this set of conditions as well as the thrust necessary to ensure the respective set of parameters, solving the following system of equations:

- $\dot{v}_{eq}(t) = -S(\omega_{eq})v_{eq} - g * R^{T} * (\lambda_{eq}) * \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{f_{a_{eq}}}{m} + \frac{f_{p_{eq}}}{m} \Leftrightarrow$

$$\Leftrightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = R^{T}(\lambda_{eq}) * \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} - \frac{cte_{v}}{2m} * C_{D} * S * R^{T} * (\lambda_{eq}) * \begin{bmatrix} {}^{I}v_{x_{eq}} \\ {}^{I}v_{y_{eq}} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{T_{eq}}{m} \end{bmatrix} \Leftrightarrow$$

$$\Leftrightarrow \begin{bmatrix}0\\0\\0\end{bmatrix} \pm \begin{bmatrix}0.0856946 - 0.0000023{}^{I}v_{y_{eq}} - 0.0297284{}^{I}v_{x_{eq}}\\ -0.0346832{}^{I}v_{y_{eq}} - 0.0856913\\ 28.5714286T_{eq} + 0.0004779{}^{I}v_{x_{eq}} - 0.0004779{}^{I}v_{y_{eq}} - 9.8192522\end{bmatrix} \Leftrightarrow$$

$$\Leftrightarrow \begin{matrix}\boldsymbol{{}^{I}v_{x_{eq}}} = 2.88276\ {}^{m}/_{s}\\ \boldsymbol{{}^{I}v_{y_{eq}}} = -2.4706\ {}^{m}/_{s}\\ \boldsymbol{T_{eq}} = 0.34358428\ N\end{matrix}$$

Knowing the Euler angles and respective inertial velocities, "v_{eq}" can be deduced, allowing to consequently know the change in position during the equilibrium trajectory "$v_{eq}t + p_{eq}(t=0)$":

- $v_{eq} = R^{T}(\lambda_{eq})^{I}v_{eq} = \begin{bmatrix}2.8827\\-2.4704\\0.0467\end{bmatrix} {}^{m}/_{s}.$

With this defined, there are only three components in our plant's input left to calculate ("$n_{p_{eq}}$"):

- $\dot{\omega}_{eq}(t) = J^{-1}S(\omega_{eq})J\omega_{eq} + J^{-1}n_{p_{eq}} \Leftrightarrow n_{p_{eq}} = \begin{bmatrix}0\\0\\0\end{bmatrix} N.m.$

3.7. Goal number 1.7

Obtain the linearized models ("$\dot{\tilde{x}} = A_{OPi}\tilde{x} + B_{OPi}\tilde{u}$") around the equilibrium points and trajectories, considering $Z - Y - X$ Euler angles kinematics.

For the linearization of each mode of operation, the group needs to calculate the matrices "A", "B", "C" and "D" that will represent the linearization of each mode. Through Taylor's polynomial of first order, it's made an approximation of the "f" function that describes the state variation.

The result of this operation is given as "$\frac{df}{dx^T}(x_{eq}, u_{eq})$", which corresponds to the matrix "A". The result for the matrix "B" is "$\frac{df}{du^T}(x_{eq}, u_{eq})$".

By doing the derivatives in question, the generic matrices of "A" and "B" are given as a function of the equilibrium states and inputs[5]. Through the use of MATLAB R2021a, it's possible to quickly obtain the matrices of the respective linearizations.

Acknowledging that it's a simulated environment and it's not limited by sensors, the matrix "C" corresponds to an identity matrix 12×12 for both modes, allowing the observation of all states.

Meanwhile, the matrix "D" corresponds to a 12×4 matrix of zeros for both operating modes since the inputs don't have a direct implication on the output.

0	0	0	0	0	0	1.0000	0	0	0	0	0
0	0	0	0	0	0	0	1.0000	0	0	0	0
0	0	0	0	0	0	0	0	1.0000	0	0	0
0	0	0	0	0	0	0	0	0	1.0000	0	0
0	0	0	0	0	0	0	0	0	0	1.0000	0
0	0	0	0	0	0	0	0	0	0	0	1.0000
0	0	0	0	9.8200	0	-0.0325	0	0	0	0	0
0	0	0	-9.8200	0	0	0	-0.0379	0	0	0	0
0	0	0	0	0	0	0	0	-0.0599	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Figure 10 – Matrix "A" on the operating mode "Hovering"

1.0e+04 *

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0.0031	0	0	0
0	7.1685	0	0
0	0	6.9638	0
0	0	0	4.6019

Figure 11 – Matrix "B" while on the operating mode "Hovering"

0	0	0	-0.0216	-0.0934	2.4701	1.0000	0.0001	0.0087	0	0	0
0	0	0	0.0683	0	2.8818	0	1.0000	-0.0087	0	0	0
0	0	0	-2.4700	-2.8818	0	-0.0087	0.0087	0.9999	0	0	0
0	0	0	0	0	0	0	0	0	1.0000	0.0001	0.0087
0	0	0	0	0	0	0	0	0	0	1.0000	-0.0087
0	0	0	0	0	0	0	0	0	0	0.0087	1.0000
0	0	0	0	9.8196	0	-0.0325	0	0	0	0.0467	-2.4706
0	0	0	-9.8193	0.0007	0	0	-0.0379	0	-0.0467	0	-2.8825
0	0	0	0.0857	0.0857	0	0	0	-0.0599	2.4706	2.8825	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Figure 12 - Matrix "A" while on the operating mode "Horizontal Flight"

1.0e+04 *

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0.0031	0	0	0
0	7.1685	0	0
0	0	6.9638	0
0	0	0	4.6019

Figure 13 – Matrix "B" while on the operating mode "Horizontal Flight"

3.8. Goal number 1.8

Decompose each linearized model into modes and discuss the results.

In order to decompose each linearized model, it's important to divide this study into the two different operation modes: "Hovering" and "Horizontal Flight".

To analyse the different linearized models, it was considered that the drone in the beginning of the simulation is in a fixed point, distant from the ground below.

3.8.1. Decomposition of the linearized model of "Hovering"

For the simulation of this linear mode to be analysed, it's important to determine certain initial conditions regarding the state of the drone. These are as follows:

- $p(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} m;$
- $\lambda(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} rad;$
- $v(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} {}^{m}/_{s};$
- $\omega(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} {}^{rad}/_{s}.$

In the case for the operating mode "Hovering" all the conditions defined above are null due to the fact that this mode requires the stable position of the drone on the air, only with a constant thrust that counteracts the force of Earth's gravity.

The total thrust input was defined has constant and equal to the equilibrium thrust calculated on the point of equilibrium of the hover linearization ("$T_{eq} = mg$").

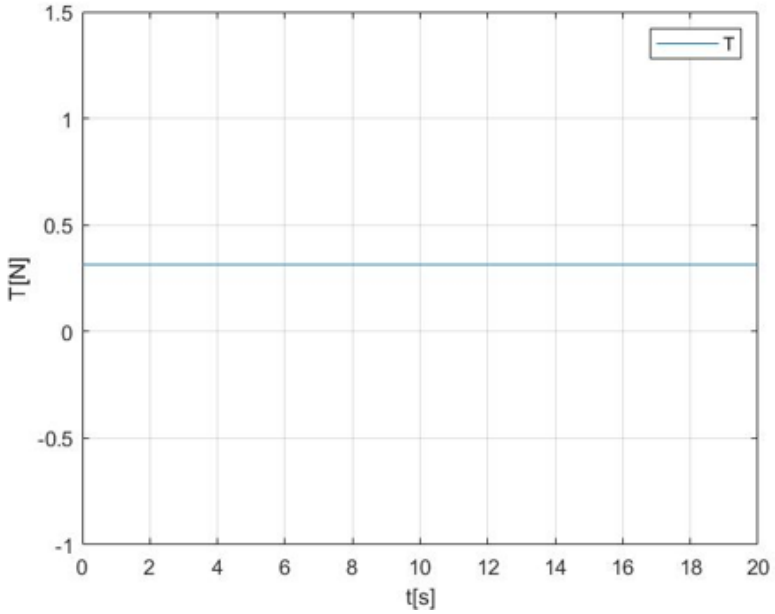

Figure 14 – Total Thrust ("$T[N]$") over Time ("$t[s]$") during "Hovering"

The other input required is the moment generated by the drone along the different axis of the body ("$n_{p_{eq}} = 0$", as it was calculated in the linearization). This also means that there's equal thrust in all four rotors, thus maintaining its hover position.

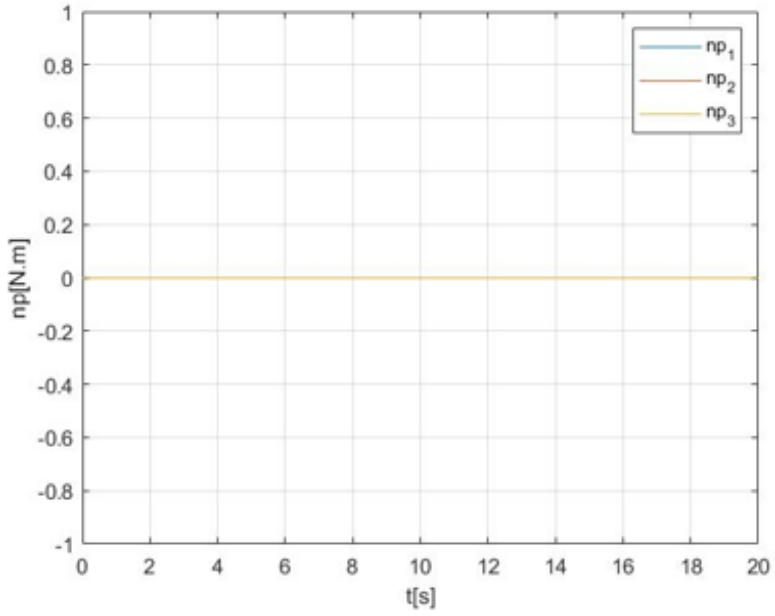

Figure 15 – Moment generated by the drone ("$np[N.m]$") over Time ("$t[s]$") during "Hovering"

The next set of graphics are the results from the simulation of "Hovering".

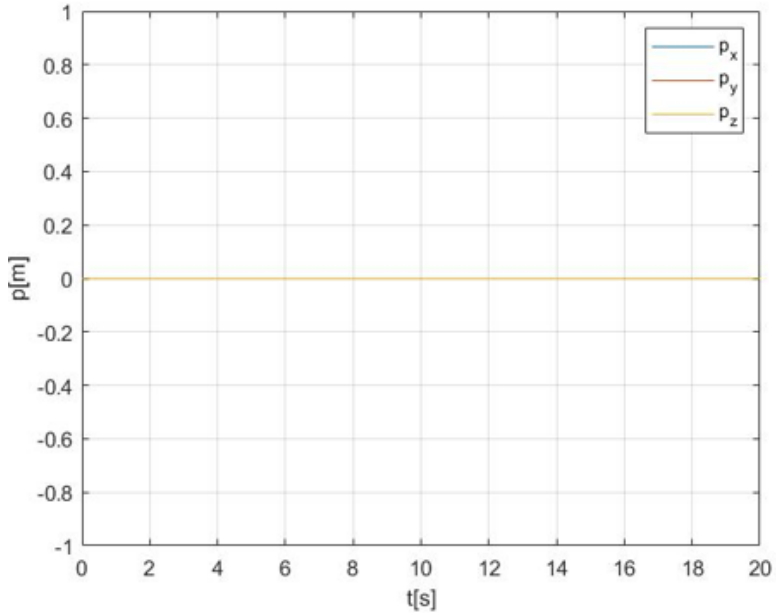

Figure 16 – Position ("$p[m]$") over Time ("$t[s]$") during "Hovering"

As shown by the last figure, the drone remains in the same position throughout the simulation. This is expected due to the fact that a variation in position isn't wanted, the goal is the drone to stay in position.

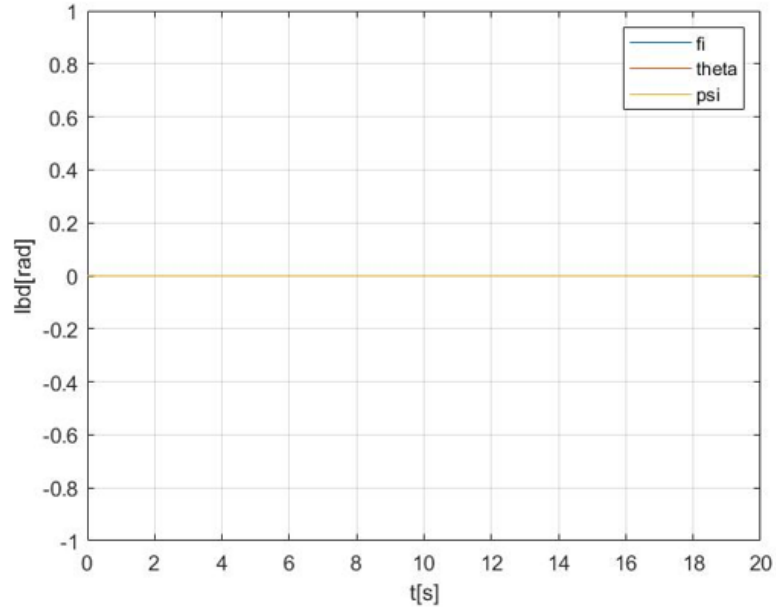

Figure 17 – Euler angles ("$lbd[rad]$") over Time ("$t[s]$") during "Hovering"

The Euler angles in this linearized mode are null, justifying the lack of variation in any velocity and angular velocity, with both being null as can be seen in the following figures.

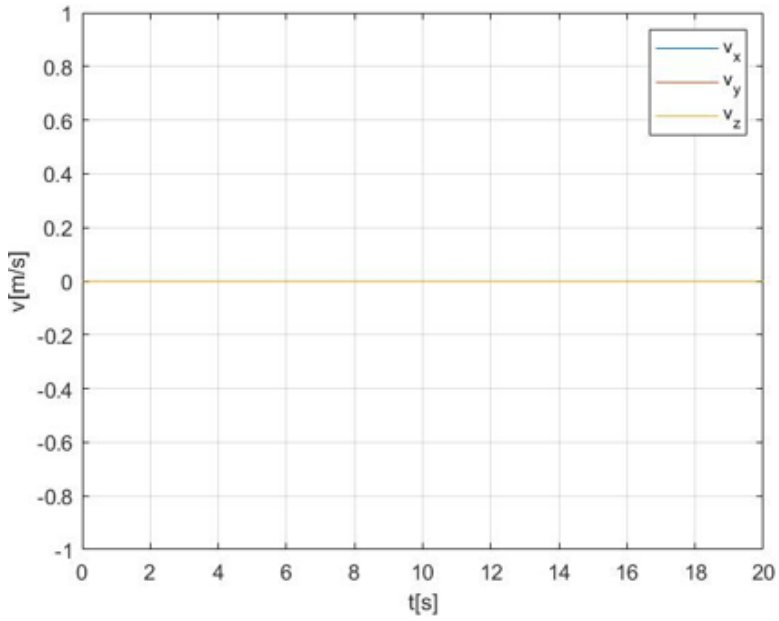

Figure 18 – Velocity ("$v[m/s]$") over Time ("$t[s]$") during "Hovering"

Although there is a propulsion force, this force only counteracts the force of Earth's gravity, causing no acceleration, therefore there's no velocity applied to the drone.

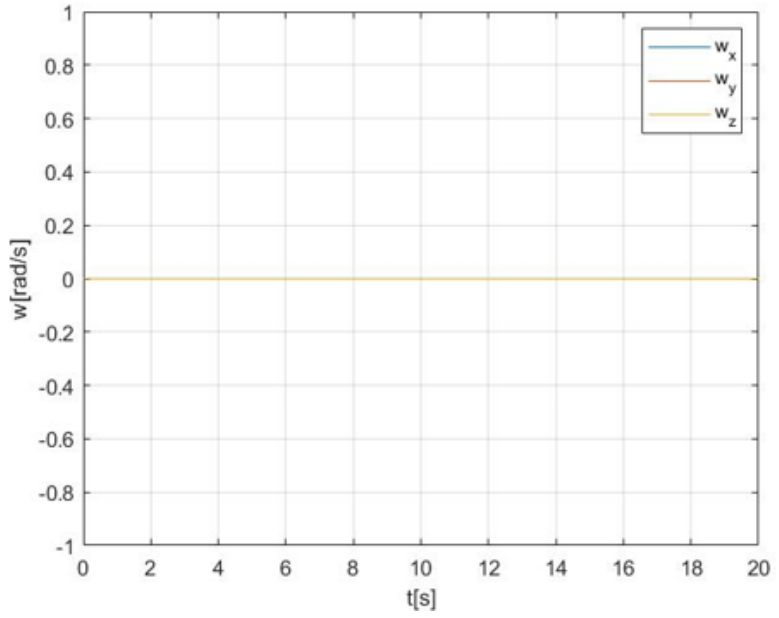

Figure 19 – Angular Velocity ("$w[rad/s]$") over Time ("$t[s]$") during "Hovering"

The angular velocity in null due to the lack of moment mentioned before.

3.8.2. Decomposition of the linearized model of "Horizontal Flight"

The linearized model of "Horizontal Flight" is essentially a mode where the drone realizes a horizontal movement, so, in order to evaluate the linearized model, the equilibrium points previously obtained and simulate the operating mode in action, certain initial conditions were assumed regarding the state of the drone at the start of the simulation:

- $p(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} m;$
- $\lambda(0) = \begin{bmatrix} \frac{\pi}{360} \\ \frac{\pi}{360} \\ 0 \end{bmatrix} rad;$
- $v(0) = \begin{bmatrix} 5.7649 \\ -4.9412 \\ -0.0934 \end{bmatrix} {}^{m}/_{s};$
- $\omega(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} {}^{rad}/_{s}.$

As it can be observed, all the values given for the initial state of the model are equivalent to the values of the equilibrium states obtained previously, with the exception of the velocity which, as a vector, it has the same direction and orientation but has values with the double of the size.

Furthermore, in addition to defining the states at the first instant, the inputs were also defined for the interval that governs the simulation. The total thrust is constant and equal to the equilibrium thrust defined previously on the respective equilibrium point of the operating mode "Horizontal Flight" ("$T(t) = 0.34358428\ N$").

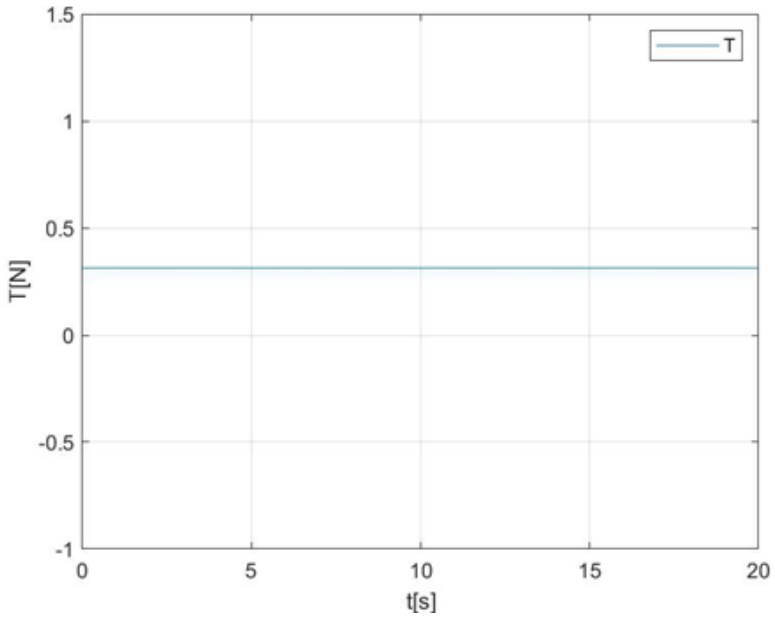

Figure 20 – Total Thrust ("$T[N]$") over Time ("$t[s]$") during "Horizontal Flight"

The remaining component of the input refers to the moment generated by the drone along the different axis of the body, which is null throughout the entire range, not differing from its equilibrium point, and consequently implying that the thrust of all rotors are equal in order to possibly generate the expected horizontal movement.

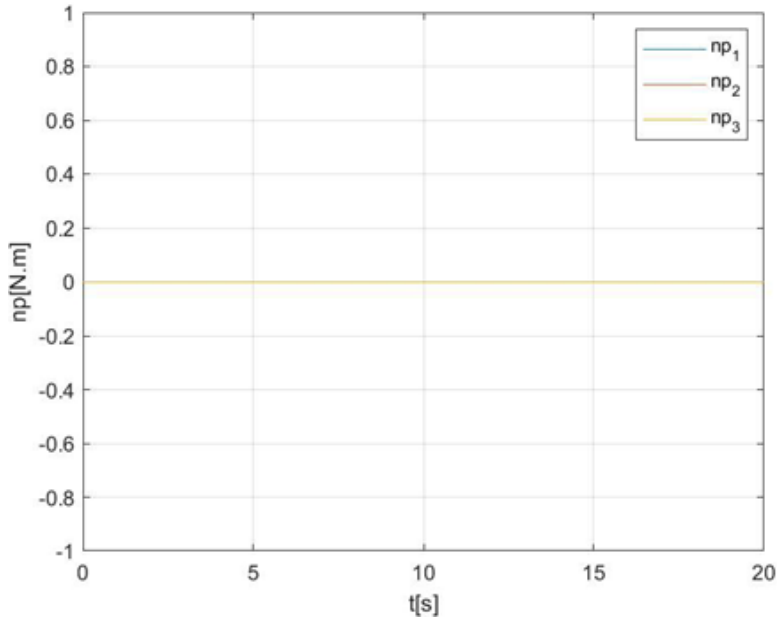

Figure 21 – Moment generated by the drone ("$np[N.m]$") over Time ("$t[s]$") during "Horizontal Flight"

The next set of graphics are the results from the simulation of "Horizontal Flight".

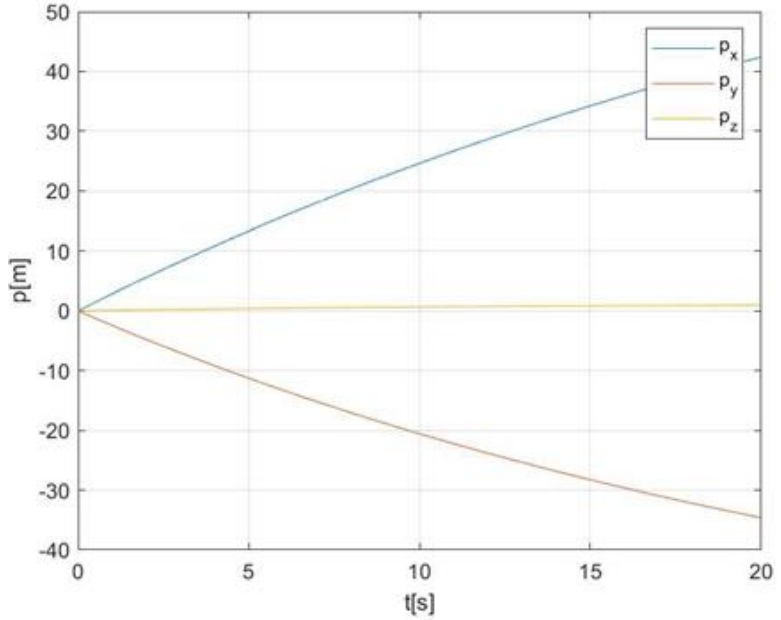

Figure 22 – Position ("$p[m]$") over Time ("$t[s]$") during "Horizontal Flight"

A logarithmic growth can be observed in relation to the drone's position "p_x" associated with the velocity along "X_I" being positive. However, "v_{x_I}" decreases due to the inclination of the Crazyflie 2.1 being equal to the inclination in the equilibrium, which for the linearization made and for the equilibrium point chosen, it implies a lower and constant velocity equivalent to the velocity of equilibrium along "X_I".

The parameter "p_y" shows a similar progression to "p_x", but negative since the velocity according to the inertial "Y_I" is negative. The slope is similar to the slope of "p_x" and the velocity module along "Y_I" is lower than "X_I".

Regarding "p_z", it remains practically null, with a slight variation due to the linearization, more specifically due to the respective matrix "A", where it's possible to observe the relationship between the components of velocity and "p_z".

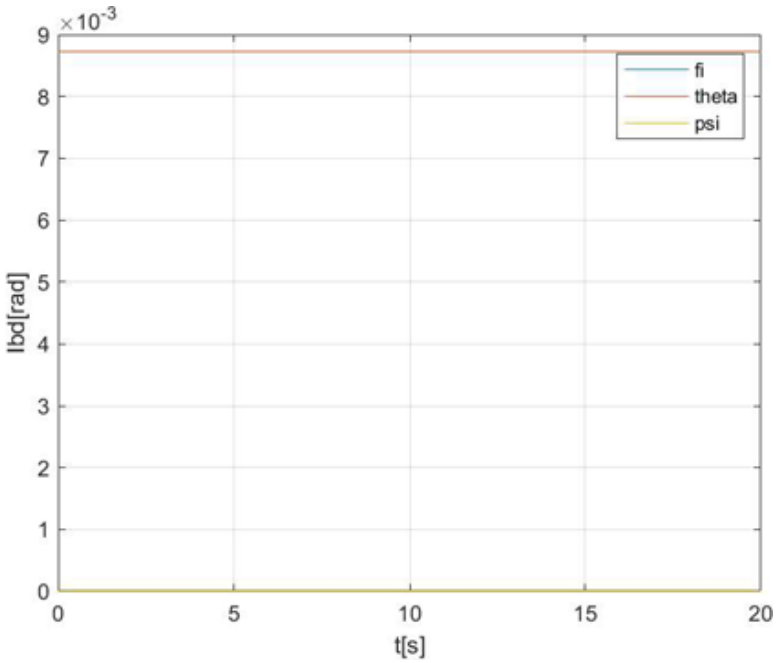

Figure 23 – Euler Angles ("$lbd[rad]$") over Time ("$t[s]$") during "Horizontal Flight"

The Euler angles remain constant as expected (the drone has null yaw while pitch and roll have a value of $0.5º \approx 8.72665 \times 10^{-3}\ rad$) since there is no variation in angular velocity, which is null throughout the simulation, as it can be seen below.

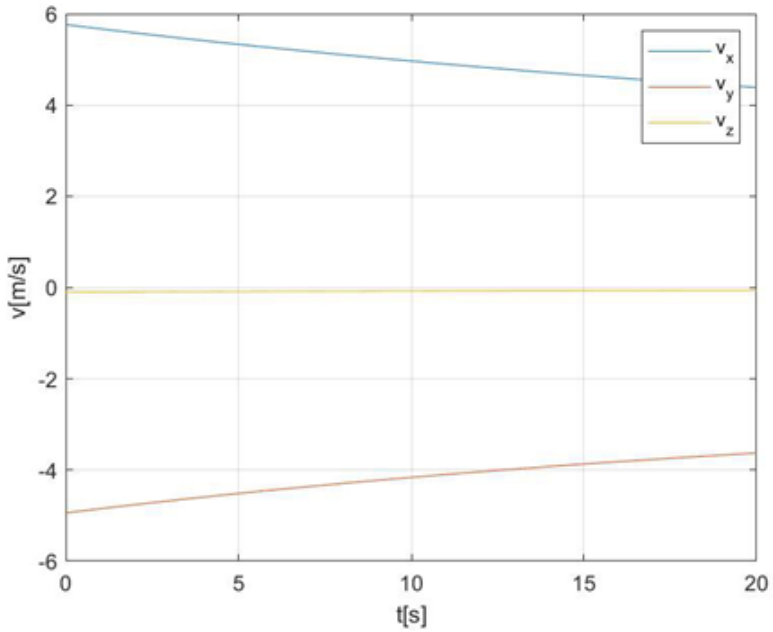

Figure 24 – Velocity ("$v[m/s]$") over Time ("$t[s]$") during "Horizontal Flight"

Although realistically the propulsion forces impose an acceleration on the drone, the velocity in relation to the Euler angles that describe the inclination of the drone tends to converge to the equilibrium velocity that this model has associated with the current inclination. This can be justified due to the linearization made around the equilibrium point and the dynamics it represents.

With these results, it's possible to verify that over time the model may become inadequate to represent the system in question, especially for the velocity of this UAV.

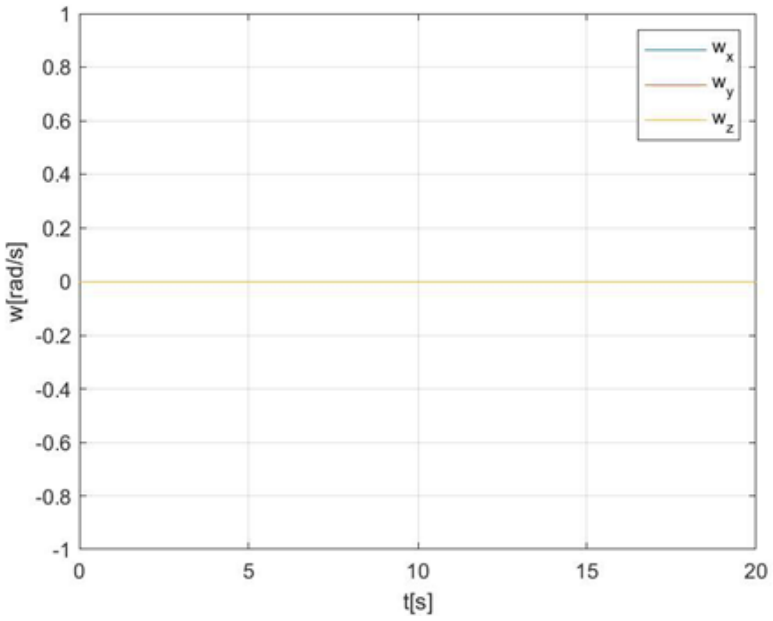

Figure 25 – Angular Velocity ("$w[rad/s]$") over Time ("$t[s]$") during "Horizontal Flight"

The angular velocity, as expected and mentioned before, is null due to the lack of moment for its entire range of time.

3.9. Goal number 1.9

Analyse each linearized model regarding their controlability, observability and stability.

In order to analyse the linearized model regarding their controllability, observability and stability, it's necessary to separate the analysis into the two different operating modes.

In order to analyse the stability for any operating mode, it's crucial to determine the Jordan matrix. Only from this matrix it can be verified the existence of the eigenvalues of "A" for the different modes of operation and consequently the stability of the model for the respective equilibrium point.

3.9.1. Analysis of the linearized model "Hovering"

```
JJ1 =

         0    1.0000         0         0         0         0         0         0         0         0         0         0
         0         0    1.0000         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0   -0.0325         0         0         0         0         0         0         0         0
         0         0         0         0   -0.0599         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0    1.0000         0         0         0         0
         0         0         0         0         0         0         0         0    1.0000         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0   -0.0379         0         0
         0         0         0         0         0         0         0         0         0         0         0    1.0000
         0         0         0         0         0         0         0         0         0         0         0         0
```

Figure 26 – Jordan matrix for "Hovering"

Analysing the eigenvalues "λ_i", this operating mode is unstable since "$\lambda_1 = \lambda_2 = \lambda_3 = \lambda_6 = \lambda_7 = \lambda_8 = \lambda_9 = \lambda_{11} = \lambda_{12} = 0$" and a Jordan block with more than two null values, with "$\lambda_4 = -0.0335$", "$\lambda_5 = -0.0599$" and "$\lambda_{10} = -0.0379$".

Regarding the controlability, a linearized system is controllable if "$rankM_c(n) = n$", with "$M_c(n) = [B \quad AB \quad \dots \quad A^{n-1} \quad B]$". Since this linearized system confirms this requisite ("$rankM_c(12) = 12$"), this means that "Hovering" is an observable system.

For a linearized system to be observable, the system requires that "$rankM_o(n) = n$", with "$M_o(n) = \begin{bmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{bmatrix}$". Since this linearized system confirms this requisite ("$rankM_o(12) = 12$"), operating mode is controllable.

3.9.2. Analysis of the linearized model "Horizontal Flight"

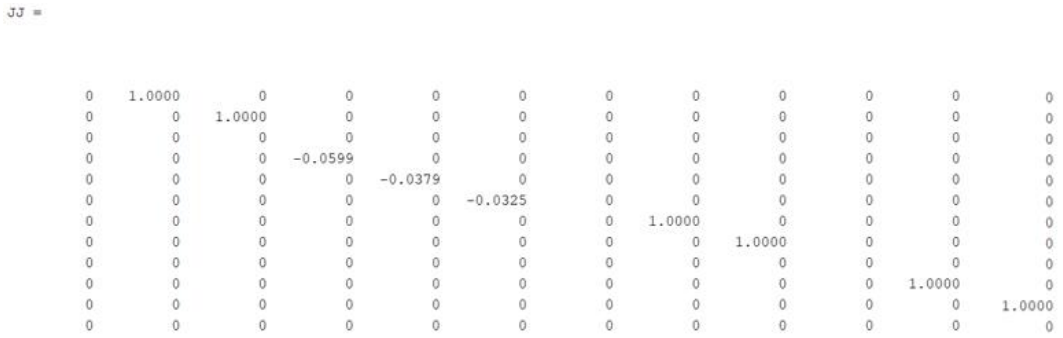

Figure 27 – Jordan matrix for "Horizontal Flight"

Analysing the eigenvalues "λ_i", this operating mode is unstable since "$\lambda_1 = \lambda_2 = \lambda_3 = \lambda_7 = \lambda_8 = \lambda_9 = \lambda_{10} = \lambda_{11} = \lambda_{12} = 0$" and a Jordan block with more than two null values, with "$\lambda_4 = -0.0599$", "$\lambda_5 = -0.0379$" and "$\lambda_{10} = -0.0325$".

Regarding the controlability, a linearized system is controllable if "$rank M_c(n) = n$", with "$M_c(n) = [B \quad AB \quad \dots \quad A^{n-1} \quad B]$". Since this linearized system confirms this requisite ("$rank M_c(12) = 12$"), this means that "Hovering" is an observable system.

For a linearized system to be observable, the system requires that "$rank M_o(n) = n$", with "$M_o(n) = \begin{bmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{bmatrix}$". Since this linearized system confirms this requisite ("$rank M_o(12) = 12$"), this operating mode is controllable.

3.10. Goal number 1.10

Deduce the transfer functions and state-space models related with the inner and outer dynamics loops defined by the following conditions:

- "$\boldsymbol{G_{n_x,\phi}(s)} = {\phi(s)}/{n_x(s)}$"
- "$\boldsymbol{G_{n_y,\theta}(s)} = {\theta(s)}/{n_y(s)}$"
- "$\boldsymbol{G_{n_z,\psi}(s)} = {\psi(s)}/{n_z(s)}$"
- "$\boldsymbol{G_{\phi,p_y}(s)} = {p_y(s)}/{\phi(s)}$"
- "$\boldsymbol{G_{\theta,p_x}(s)} = {p_x(s)}/{\theta(s)}$"
- "$\boldsymbol{G_{T,p_z}(s)} = {p_z(s)}/{T(s)}$"

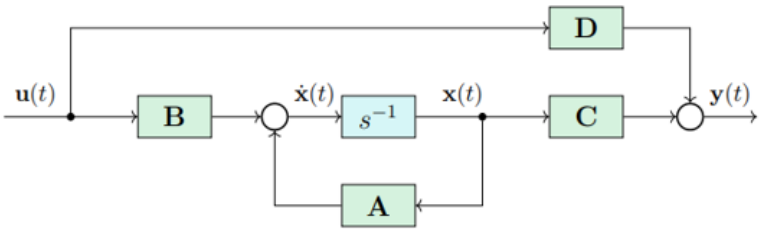

Figure 28 – State-space model block diagram representation

In order to represent the linearized time-invariant system, the equations necessary are the following:

- $\dot{x}(t) = Ax(t) + Bu(t)$;
- $y(t) = Cx(t) + Du(t)$.

Applying a Laplace transform into both equations:

- $\begin{cases} sX(s) = AX(s) + BU(s) \\ Y(s) = CX(s) + DU(s) \end{cases}$.

Following this system of equations by solving it:

- $H(s) = \frac{Y(s)}{U(s)} = C(sI - A)^{-1}B + D$.

The obtain function "$H(s)$" corresponds to a matrix of transfer functions (MIMO) with a number of rows corresponding to the number of outputs ("$y(t)$") and a number of columns equal to the number of inputs ("$u(t)$"). Each position to a transfer function (SISO) that describes the relationship between the inputs (columns) and the outputs (rows).

Both "$H(s)$" matrices from "Hovering" and from "Horizontal Flight" have a 12×4 configuration and the transfer functions representing the respective relationships between the specific inputs and outputs can be found in the same positions and relationships between different outputs (or states) through the use of the same operations.

The outer dynamic functions can be found in "$H(s)$" in the following positions of the matrix:

- $G_{n_x,\phi} = {\phi(s)}/{n_x(s)}$, in the 4^{th} line, 2^{nd} column;
- $G_{n_y,\theta} = {\theta(s)}/{n_y(s)}$, in the 5^{th} line, 3^{rd} column;
- $G_{n_z,\psi} = {\psi(s)}/{n_z(s)}$, in the 6^{th} line, 4^{th} column;
- $G_{T,p_z} = {p_z(s)}/{T(s)}$, in the 3^{rd} line, 1^{st} column.

The inner dynamic functions can be obtained indirectly through operations such as:

- $G_{\phi,p_y} = \frac{p_y(s)}{\phi(s)} = \frac{p_y(s)}{n_x(s)} * \frac{n_x(s)}{\phi(s)} = \frac{G_{n_x,p_y}}{G_{n_x,\phi}}$, where $G_{n_x,p_y} = H(s)$ in the 2^{nd} line, 2^{nd} column;
- $G_{\theta,p_x} = \frac{p_x(s)}{\theta(s)} = \frac{p_x(s)}{n_y(s)} * \frac{n_y(s)}{\theta(s)} = \frac{G_{n_y,p_x}}{G_{n_y,\theta}}$, where $G_{n_y,p_x} = H(s)$ in the 1^{st} line, 3^{rd} column.

The transfer functions obtained from the linear system model used in "Hovering" are the following:

- $G_{n_x,\phi} = \frac{7.168\times10^4}{s^2}$;
- $G_{n_y,\theta} = \frac{6.964\times10^4}{s^2}$;
- $G_{n_z,\psi} = \frac{4.602\times10^4}{s^2}$;
- $G_{T,p_z} = \frac{31.25}{s^2+0.0599s}$;
- $G_{\phi,p_y} = \frac{-7.039\times10^5 s^2}{7.168\times10^4 s^4+2719s^3}$;
- $G_{\theta,p_x} = \frac{6.838\times10^5 s^2}{6.694\times10^4 s^4+2264s^3}$.

The transfer functions obtained from the linear system model used in “Horizontal Flight” are the following:

- $G_{n_x,\phi} = \frac{7.168\times10^4}{s^2}$;
- $G_{n_y,\theta} = \frac{6.964\times10^4}{s^2}$;
- $G_{n_z,\psi} = \frac{4.602\times10^4}{s^2}$;
- $G_{T,p_z} = \frac{31.25}{s^2+0.0599s}$;
- $G_{\phi,p_y} = \frac{-7.039\times10^5 s^3-4.215\times10^4 s^2}{7.168\times10^4 s^5+7013s^4+162.9s^3}$;
- $G_{\theta,p_x} = \frac{\left(6.838\times10^5 s^6+6.688\times10^4 s^5+1553s^4-4.896\times10^{-14}s^3+2.677\times10^{-31}s^2\right)}{6.964\times10^4 s^8+9077s^7+379.8s^6+5.145s^5}$.

The results show that the outer dynamics remain the same in both systems, while the inner dynamics show different results with higher degree of complexity in both cases (for example, in both systems the inputs applied influence their progress in a similar way, while the system states cause different progress depending on the system in question.

3.11. Goal number 1.11

Discuss the closed loop stability of the modes defined by "$\boldsymbol{G_{T,p_z}(s)}$" and "$\boldsymbol{G_{n_x,p_y}(s)}$", for instance, considering the root-locus arguments for a given control structure.

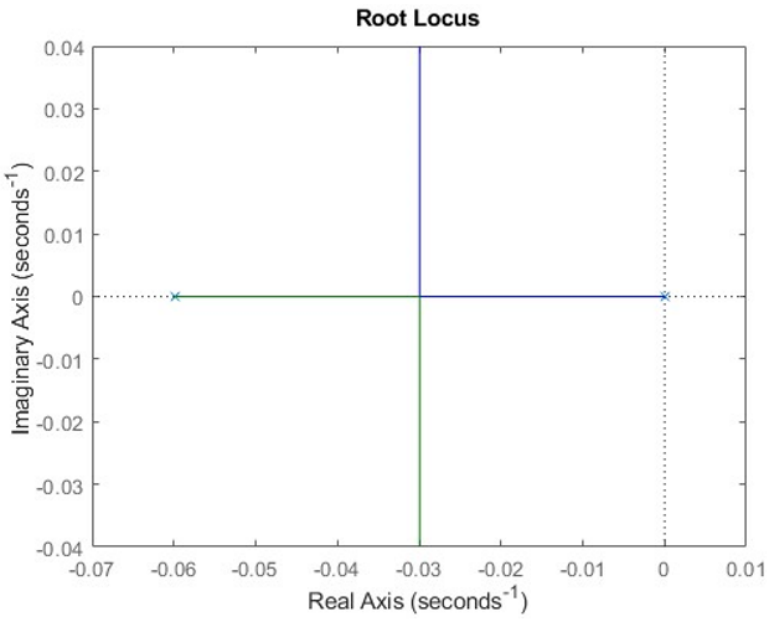

Figure 29 – Root-Locus of G_{T,p_z} on "Hovering"

From this root-locus graphic it's possible to observe that this SISO system is marginally stable for a controller gain of 0 (open-loop) since there is one negative pole and other one at the origin. For any controller gain "k" (closed-loop, where "$k \in]0,+\infty[$") the SISO system is stable since the real part of the poles of this second order function are always negative.

Being the open-loop a transfer function of 2^{nd} order, defined by one negative pole and one pole at the origin (meaning that "$m = 2$") and no zeros ("$n = 0$"), there are two asymptotes ("$m - n = 2$") centred at $-0.03\,Hz$ ("$assymptotic\ centre = \frac{\sum poles - \sum zeros}{m-n}$") and with angles of $90º$ and $270º$ ("$assympote\ angles = \frac{180º + 360º(l-1)}{m-n}, l = 1, \dots, m-n \iff assymptote\ angle = 90º \wedge assymptote\ angle = 270º$") where each one of the respective conjugated poles will converge for bigger control gains.

As it can be observed, up to "$-0.03\,Hz$" the poles are real and at this frequency the poles can overlap, which would require a gain "$k = 2.87 \times 10^{-5}$". For greater values of "k", the poles would become conjugates of each other, while following a vertical asymptote (characteristic of the existence of two poles than zeros), the damping factor becomes different from 1. If the damping factor goes towards infinity, the asymptote will tend to 0 and consequently overshoot to 100%, but never taking these exact values because this would imply that the poles are located on the imaginary axis and marginally stable.

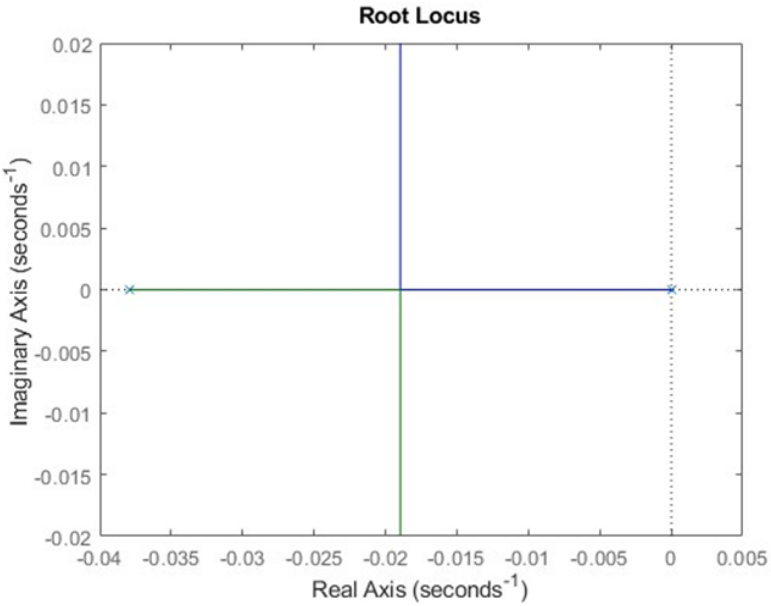

Figure 30 – Root-Locus of G_{ϕ,p_y} of "Hovering"

Being a transfer function of the 4^{th} order, defined by one negative pole and three poles at the origin ("$m = 4$"), it's possible to conclude that the open-loop system is marginally stable. Since it has 2 zeros at the origin ("$n = 2$"), there are two asymptotes ("$m - n = 2$"), centred at "$-0.019\,Hz$" located at $90º$ and $270º$ where each of the respective conjugated poles will converge for bigger control gains.

Due to the equivalent difference of poles and zeros and since the two zeros cancel two out of the three poles located at the origin, the root-locus shape shows to be similar to the previous root-locus, differentiating in the asymptotic centre and the controller gains and respective poles positions. Consequentially, the analysis is similar, where essentially Is possible to conclude that the SISO system is marginally stable for null gain (on open-loops) of the controller ("k") and stable for bigger gains (on closed-loops).

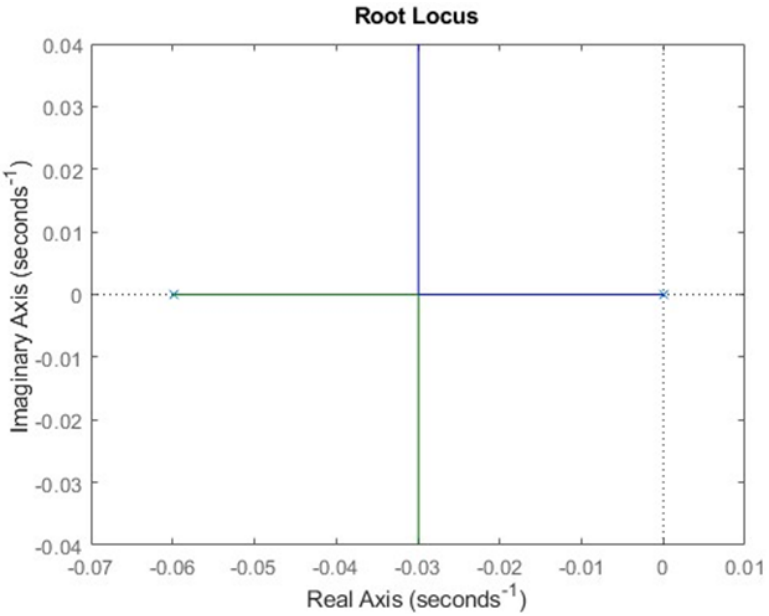

Figure 31 – Root-Locus of G_{T,p_z} of "Horizontal Flight"

As expected from the equal transfer functions for both operating modes, the root-locus is similar, and the analysis is exactly the same as the one for "Hovering".

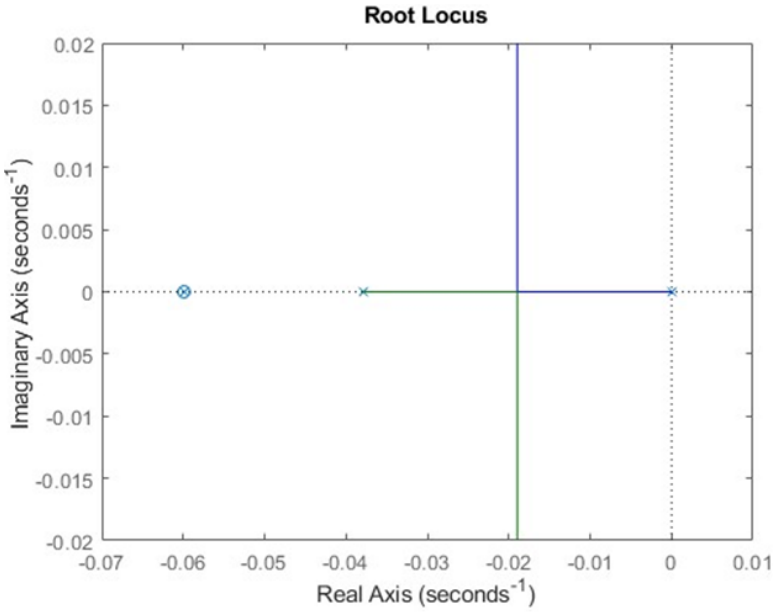

Figure 32 – Root-Locus of G_{ϕ,p_y} of "Horizontal Flight"

Considering a 5^{th} order transfer function, defined by two negative poles and three poles at the origin ("$m = 5$"), it's possible to conclude that the open-loop system is marginally stable. Since the system has 2 zeros at the origin and 1 zero negative ("$n = 3$"), there are two asymptotes ("$m - n = 2$"), centred at "$-0.019\,Hz$" and located at $90º$ and $270º$, where each one of the respective conjugated poles will converge for bigger control gains.

Due to the equivalent difference of poles and zeros and, since there are 2 zeros that cancel two out of the three poles at the origin and the other zero is really close to the less significant (regarding dynamics) and most negative pole, the root-locus shape shows to be similar to the previous root-locus, differentiating in the asymptotic centre and in the controller gains and respective poles position. Consequentially, the analysis is similar, where it's possible to conclude that the SISO system is marginally stable for null gain of the controller ("k") and stable for bigger gains.

3.12. Goal number 2.1

Analyse the data and divide it in three sets, considering motions in each of the position degrees of freedom (axis "x", "y" and "z").

In this set of goals for the identification of the drone, the data given by Professor Bruno Guerreiro will be analysed and decomposed into three operating modes used before ("Hovering" and "Horizontal Flight", the last one being divided into "Horizontal Flight in Axis x" and "Horizontal Flight in Axis y"). After running the script "$razyflie\_show\_usdlog"$ it was obtained graphics for position, velocity, Euler angles, angular velocity and the thrust for each rotor.

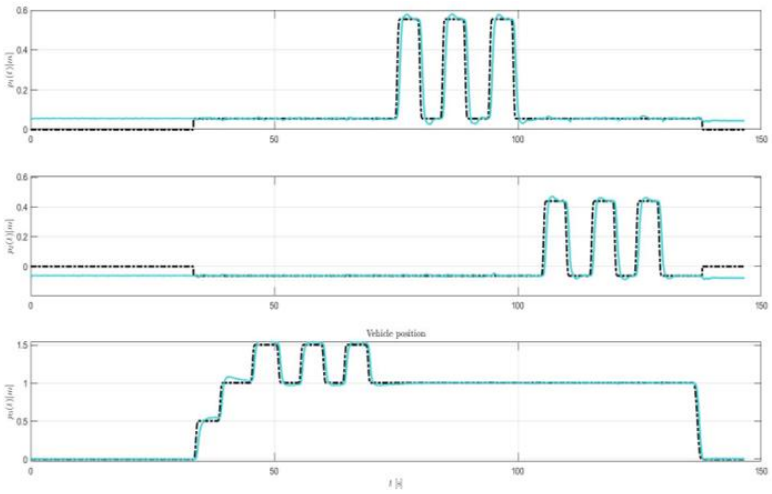

Figure 33 – Analysis of the Position of the drone

Looking at the figure, it's possible to observe the changes that occurred along the time of the simulation. In the first seconds the drone was in a resting position until around the 30 seconds mark, where there's a change of position in the axis z, which is the result of an inflicted velocity, marked by the steps it takes until half of the flight. Afterwards, the drone remains in its position regarding the axis z, having changes of position in the other axis (first in axis x and afterwards in axis y). During this period, it's possible to observe the operating mode "Horizontal Flight", where there is movement in the axis x and y, while the drone remains in the same position in the axis z.

Ater analysing the figure, it's possible to divide the data into three different phases:

- The first phase goes from the resting position of the drone into different stages of "Hovering", where there's no movement of the drone regarding the axis x and y, and the rotors only provide thrust to counteract the force of gravity and occasional changes of thrust (spikes) for vertical movement between the stages (from instant $37.5\ s$ to instant $72.1\ s$);
- The second phase can be determined as the operating mode "Horizontal Flight in Axis x", having only movement in the axis x with the pairs of rotors providing thrust in inverse proportion around an equilibrium thrust in order to apply the rotation needed to move along the axis (from instant $72.1\ s$ to instant $102.4\ s$);
- The third phase can be determined as the operating mode "Horizontal Flight in Axis y", having only movement in that axis from the same concept actions of the previous operating mode (from instant $102.4\ s$ to instant $150\ s$).

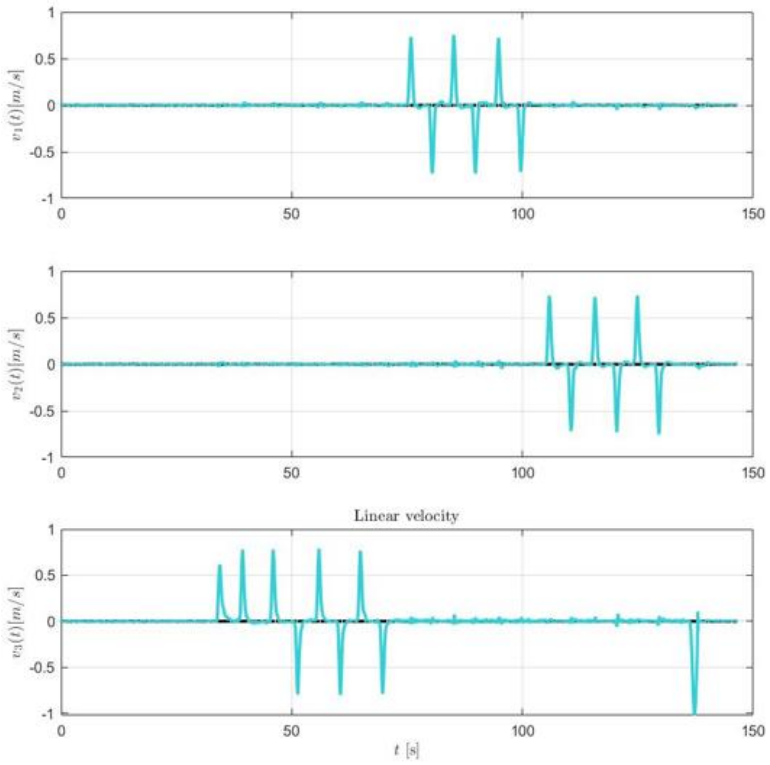

Figure 34 – Analysis of the Velocity of the drone

The velocity graphic correlates to the graphic of position, since every time there is a change in velocity (velocity no longer becomes null), occurs a change in the position as it's showed in the figures.

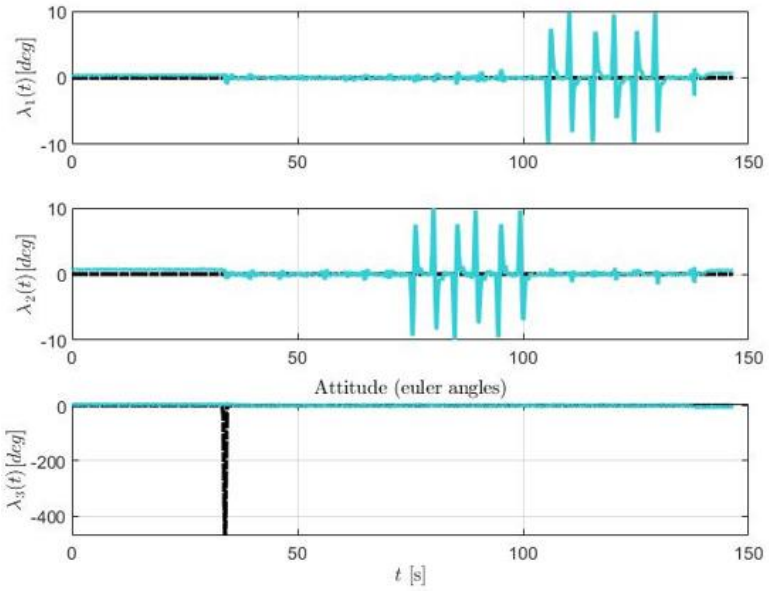

Figure 35 – Analysis of the Euler Angles of the drone

There is variation of the Euler angles in real conditions due to the fact that, in reality, the drone needs to tilt to change its position in “Horizontal Flight”.

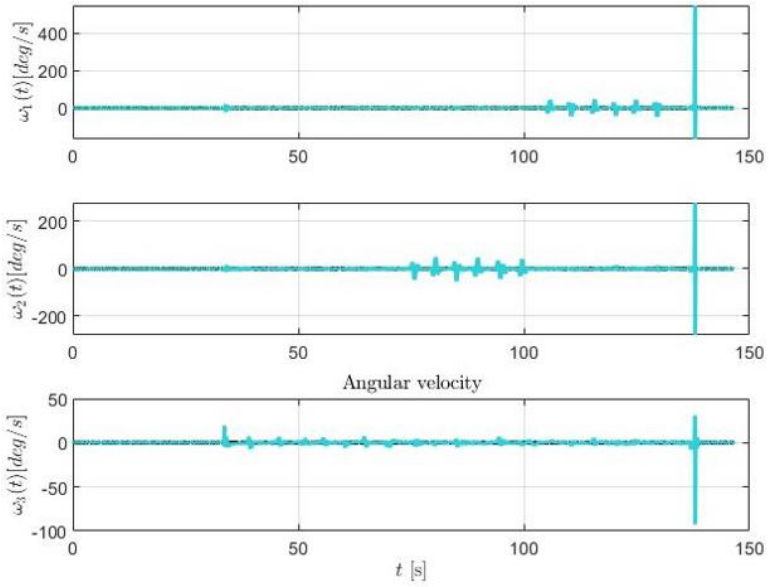

Figure 36 – Analysis of the Angular Velocity of the drone

The angular velocity doesn’t change much during the flight, the spikes seen are just the angular velocity needed to create the tilt to make “Horizontal Flight” possible.

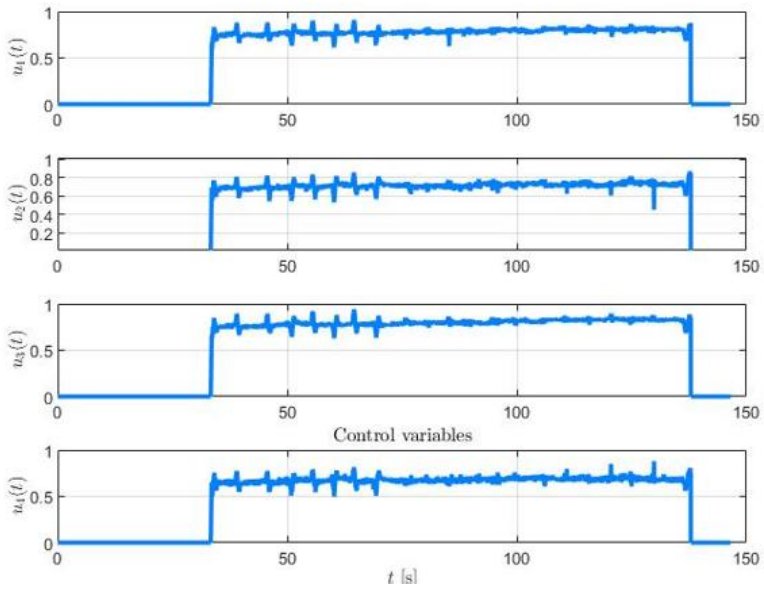

Figure 37 – Analysis of the Thrust of the drone

The graphic of the thrust shows very low variation between rotors since the values are so close to each that it does not seem to cause variation. Nevertheless, by analysing more closely the graphic, the small spikes are the increase of thrust in certain rotors to provoke the movement between the different stages of “Hovering” and in “Horizontal Flight”.

The big spikes in the beginning and in the end of the simulation mark the drone exiting and returning its resting periods (when the thrust is null).

Observing the graphic, the drone is initially at rest on the ground and has no input being applied to it. Due to limitations in the programme, it’s unable to tell when the drone starts working just by looking at the data. The data is interpreted as being in a position equal to zero when faced with a null input, not varying its position, which is a dynamic that doesn’t describe what happens when the drone is in the air. Since a null input means that the drone is descending, the dataset is divided, making it relevant for estimating and validating the model only from the moment it’s already in the air, so that the dynamics detected and the consequent estimations of the model in operation are more accurate.

3.13. Goal number 2.2, 2.3 and 2.4

2.2. Show the relation between the inputs and outputs of the transfer functions:

- "$\boldsymbol{G_{\theta,p_x}(s)} = {p_x(s)}/{\theta(s)}$";
- "$\boldsymbol{G_{\phi,p_y}(s)} = {p_y(s)}/{\phi(s)}$";
- "$\boldsymbol{G_{T,p_z}(s)} = {p_z(s)}/{T(s)}$".

2.3. Use the tools provided by MATLAB R2021a for model identification for each of these modes.

2.4. Compare the outputs of the models using the same inputs from the dataset.

With the data divided into modes, it's possible to analyse each operating mode individually, identify them and get each transfer function.

For "Hovering", with the drone hovering with movements in the axis z, the references from the following figure were used.

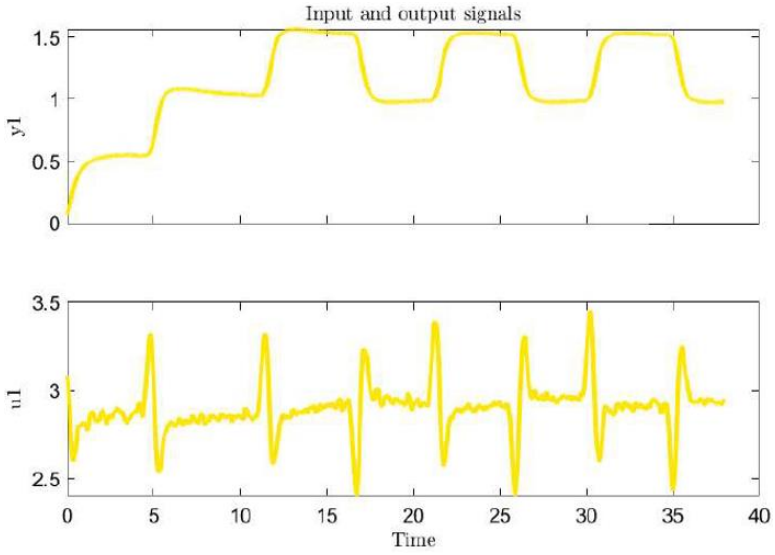

Figure 38 – Position variation with the Thrust input in "Hovering"

Model Structure refers to the structure of the transfer function and its order, more precisely the number of poles and zeros.

Realistically, there are transfer functions that represent with a certain degree of proximity the dynamics, trying to model in certain instants (the system is not linear, so the models used can change) and find the best model structure possible (which is a big step to achieve the best model that fits the system that's pretended to control).

Since it was already obtained in the first part the transfer function for the following dynamics, knowing already the possible number of poles and zeros, serving as a base for the development of the model structures wanted.

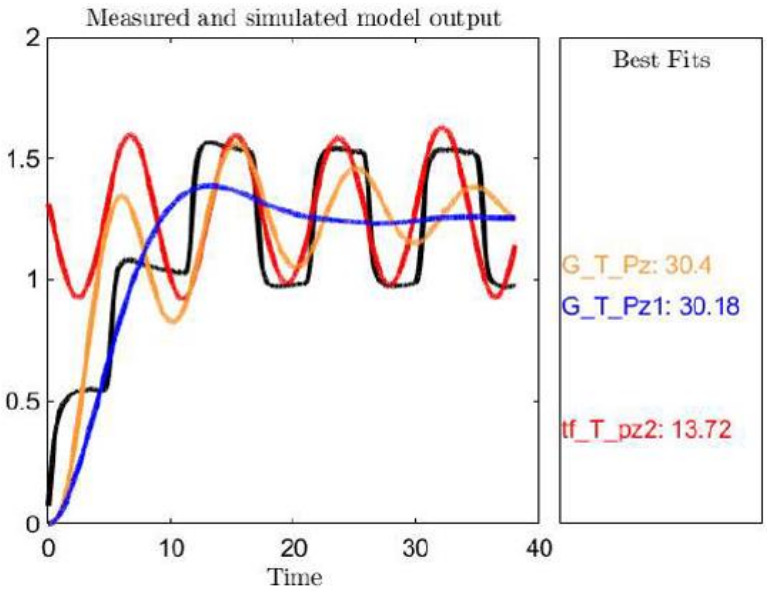

Figure 39 – Different attempts to get a model fit to use in "Hovering"

The last graphic showed various attempts to get a fitting model:

- The blue line is a model that was fed with the total reference, but it doesn't have enough poles and zeros to be a competent model;
- The red line is a model that was only given a single up-and-down wave, which could be a fitting model for this movement only;
- The orange line, similar to the blue line, was fed with the total reference but with a structure of 6 poles and 3 zeros. It doesn't have a good behaviour in the beginning, and, through the duration of the simulation, it can be seen developing a stabilizing pattern with up-and-down part passes, giving the impression that, if the up-and-down section is continuous, the model won't respond as well as it should.

The model chosen and more appropriated is shown in the next figure.

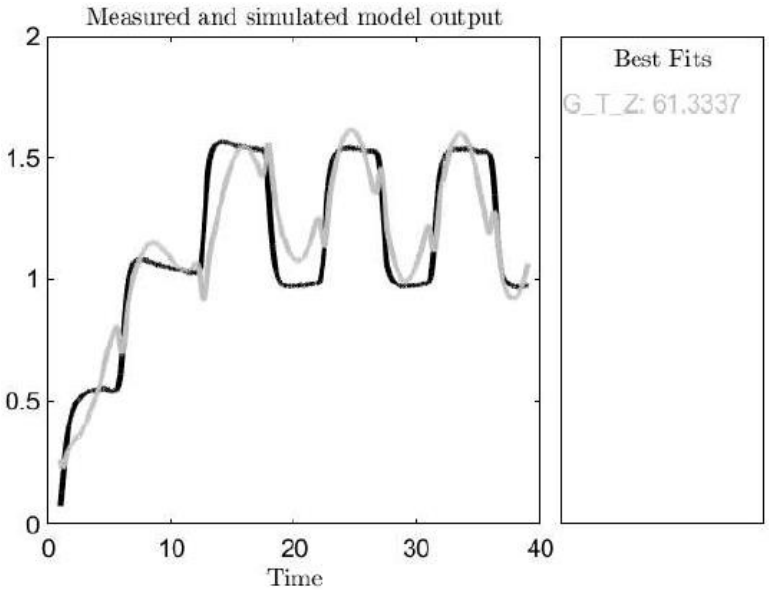

Figure 40 – The most appropriate model to use in "Hovering"

The model shown above gives the most appropriate response, being able to describe the mode wanted with bigger efficiency.

The model structure used as a reference was the one obtained for "Hovering", which consisted in two poles and no zeros, but given the unknown dynamics perceived and the estimations previously made, it can be seen that the structure wasn't the most suitable, due to the possible existence of noise in the input and the reflex of unknown dynamics. Due to this, and in order to obtain this model, it was estimated the use of 3 poles and 2 zeros as parameters.

This model was obtained through the data used for the validation of the model seen in the figure. This data was used because the model wasn't wanted neither with low amounts of data and neither with all the data, consequentially showing bias towards a sequential type of movement (for example in the waves seen before). The intention with selection of this mode was to give a correct response if it wants to go up or down, therefore giving the correct amount of data relatively to the simple movement.

What allowed the biggest improvement was the removal of some non-linearities of the dataset, more precisely in the relation between the input (thrust) and the output ("p_z"), since the change of position was needed to offset from the "Hovering" thrust. Previously, what was estimated was something similar to "$\delta p_z = (\delta T + T_{eq}) * G_{T,p_z}$" with "$T = \delta T + T_{eq}$", while "$T_{eq} * G_{T,p_z} = 0$". Removing the offset "$T_{eq}$" from the inputs and estimating "$G_{\delta T,p_z}$" around that equilibrium point ("$\delta p_z = \delta T * G_{\delta T,p_z}$") to use the model and the total thrust as an input in a simulation would need to always subtract the input by "T_{eq}" before applying the transfer function obtained and to always add "T_{eq}" for the output of a controller designed for the use of this model.

The transfer function estimated for this dynamic for “Hovering” and movement up-and-down in the axis z is:

- $G_{T,p_z} = \frac{-1.224s^2 - 0.1956s - 0.4597}{s^3 + 0.05147s^2 + 0.5759s + 0.02964}$.

For the second operating mode (“Horizontal Flight in Axis x”), the references used are represented in the following figure.

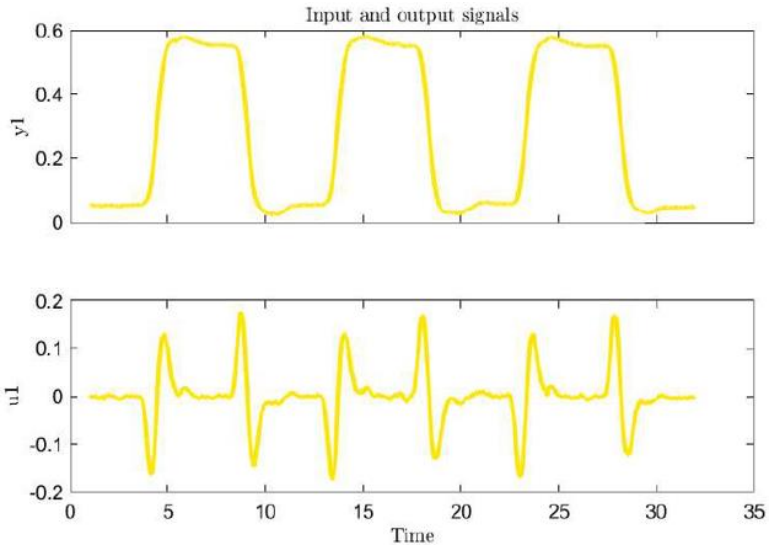

Figure 41 – Position variation with Thrust input in “Horizontal Flight in Axis x”

As occurred in the previous operating mode, it was estimated the model structure for this mode.

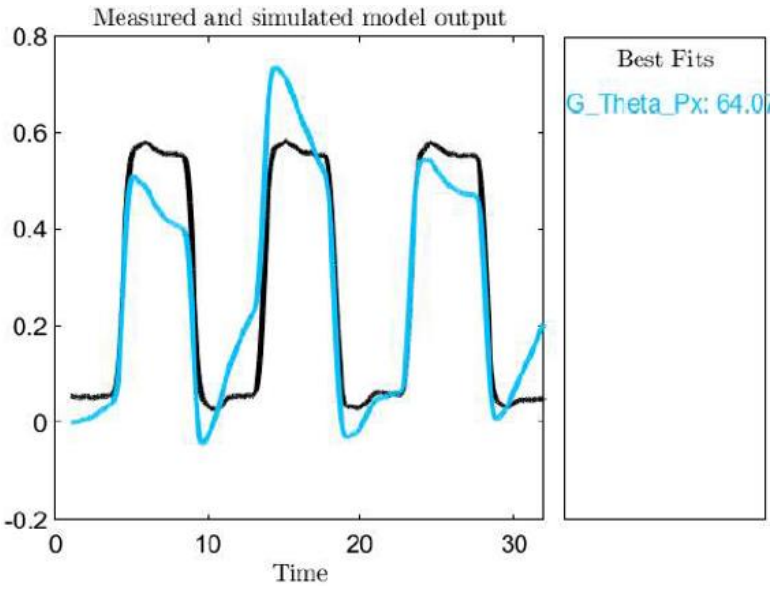

Figure 42 – The most appropriate model to use in “Horizontal Flight in Axis x”

Using the data acquired on the first part of the operating mode “Horizontal Flight”, its acknowledged that this data gives a base for a good approximation for what could be the best model structure and the reference for this flight. The graphic presents the best model developed, with 5 poles and 3 zeros, even though this model can only be used in “Horizontal Flight in Axis x”. The model has a stable behaviour, getting to move in almost the same instant that the reference, therefore being a good model to achieve the goal.

The transfer function estimated for the dynamics of “Horizontal Flight in Axis x” is the following:

- $G_{\theta,p_x} = \frac{14.86s^6 - 202.8s^5 - 6749s^4 - 4035s^3 - 1.067\times10^4 s^2 - 172.7s - 1687}{328s^7 + 746.5s^6 + 574.5s^5 + 1315s^4 + 162.6s^3 + 204.3s^2 + 2.116s + 2.549}$.

For the third and final operating mode (“Horizontal Flight in Axis y”), the references used are in the following figure.

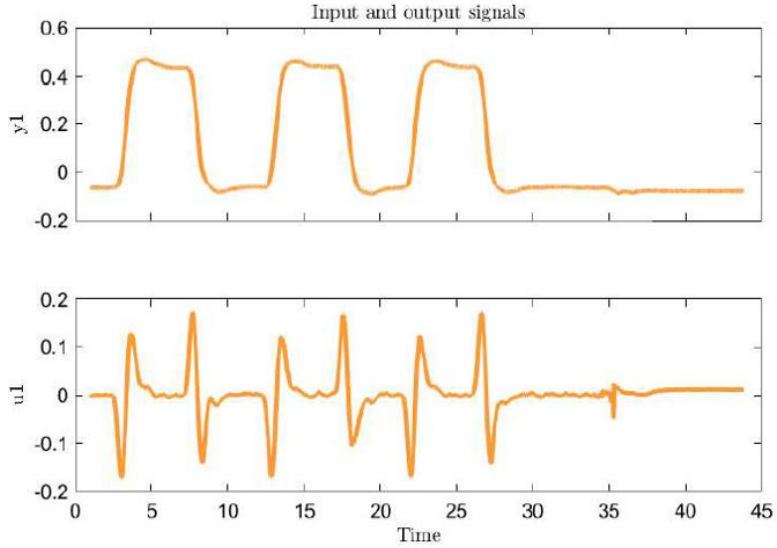

Figure 43 – Position variation with Thrust input in “Horizontal Flight in Axis y”

Estimating the model structure for this operating mode, it was obtained the following figure.

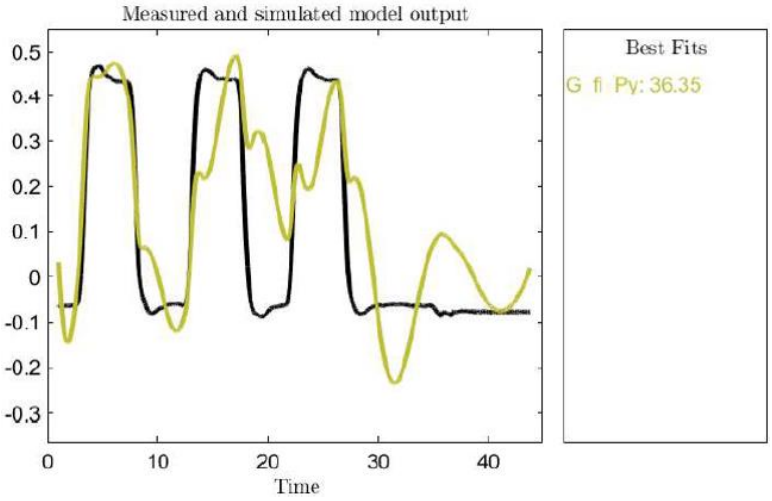

Figure 44 – The most appropriate model to use in "Horizontal Flight in Axis y"

Using the data acquired in "Horizontal Flight in Axis x" to obtain a base for a good approximation for what could be the best model structure, and with reference for this operating mode, the graphic above presents the best model acquired, with a total of 5 poles and 3 zeros.

In the start of the model, the model has good behaviour in approximation, but afterwards it starts to have some problems in getting into the following peak positions.

The transfer function estimated for the dynamics of "Horizontal Flight in Axis y" is the following:

- $G_{\phi,p_y} = \frac{-6.274s^3+4.148s^2-4.574s+1.496}{s^5+0.9972s^4+0.8588s^3+0.2858s^2+0.02944s+0.008948}$.

4. Other Comments

In order to improve the results, an error can be obtained between the result obtained from the model and the results from reality during the duration of the simulation. This error can be correlated to disturbances (for example, Gaussian noise for the effects of simulation). If the correlation is large, it can mean that the drone is being affected with external noise, if the correlation is small, the error could be related with the input, meaning that this error can come from unknown dynamics (there's always a chance of not acknowledging all the dynamics regarding the simulation, meaning that there's always a chance for errors to appear).

It's believed that the data may not be the best and most trustworthy for the process and practice of the model in order to present the most adequate dynamics for the respective situations.

There's a delay from the input to the output that could make the task harder to estimate (even though that it doesn't seem to be relevant enough in this situation). Regardless, this delay can be verified through a closer look into the graphics (through the overlap of the plots of input and output or through the function "$delayest()$")). In order to simplify the estimation, there's a need to pre-process the data, applying an advance on the data of the output equivalent to that delay (presenting a certain number of indexes forward that can be given by "$\frac{Delay}{Sample\ Period}$") and multiply it to the transfer function afterwards ("$e^{-(s*SP)}$").

5. Conclusions

During this project, the group got to understand the complexity of the systems control of unmanned autonomous vehicles and experience the kind of work that goes into every project in this area.

Even though this work is a very simplified project in comparison to the work of aerospace corporations worldwide, the challenge of this project allowed the students to learn and practice about the control models and modelling techniques that are used in aerospace engineering.

The results obtained were as expected, since the linearized models obtained are in accordance with what was expected, representing what was believed to be the dynamics of the drone during “Hovering” and “Horizontal Flight” around certain equilibrium points. The models estimated with real data had an acceptable and approximated behaviour to the real dynamics.

This project will have another two chapters that will enhance the skills of the students in order to train them into better engineers for tomorrow’s work environment in aerospace engineering and other markets with similar needs, either in Portugal or anywhere else in the world.

6. Webgraphy

[1] – Captain Brian P. Tice, "Unmanned Aerial Vehicles" – https://web.archive.org/web/20090724015052/http://www.airpower.maxwell.af.mil/airchronicles/apj/apj91/spr91/4spr91.htm

[2] – EASA (European Union Aviation Safety Agency) website, "Open Category – Low Risk – Civil Drones" where we checked the classification of a Micro Aerial Vehicle – https://www.easa.europa.eu/en/domains/civil-drones-rpas/open-category-civil-drones

[3] – NASA's Earth Fact Sheet where we obtained the Bulk Parameter of Mean Surface Gravity ("$\boldsymbol{g_{earth}} = 9.82\ [^{m}/_{s}]$") and the Terrestrial Atmosphere Surface Density ("$\boldsymbol{\rho_{earth}} = 1.217\ \left[^{kg}/_{m^3}\right]$") – https://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html

[4] – Carlos Luis, "Design of a Trajectory Tracking Controller for a Nanoquadcopter" where we defined the coefficients of drag and thrust of each rotor – https://arxiv.org/pdf/1608.05786.pdf

7. Attachments

Github repository by José Corvo:

- Link for the Main Folder – https://github.com/Jose-Corvo-Lv99/UAV-s_Grupo3_FCT_2023-24/tree/main/Project1_UAV_2023_24/Parte%201
  - Link for the Main Code Folder – https://github.com/Jose-Corvo-Lv99/UAV-s_Grupo3_FCT_2023-24/tree/main/Project1_UAV_2023_24/Parte%201/Main%20Code
  - [5] – Link for the Important Images Folder – https://github.com/Jose-Corvo-Lv99/UAV-s_Grupo3_FCT_2023-24/tree/main/Project1_UAV_2023_24/Parte%201/Imagens
  - Link for Auxiliar Code to help produce the Main Code – https://github.com/Jose-Corvo-Lv99/UAV-s_Grupo3_FCT_2023-24/tree/main/Project1_UAV_2023_24/Parte%201/Auxiliares