FDS 2024: Field and service robotics
Homework 3

Andrea Morghen

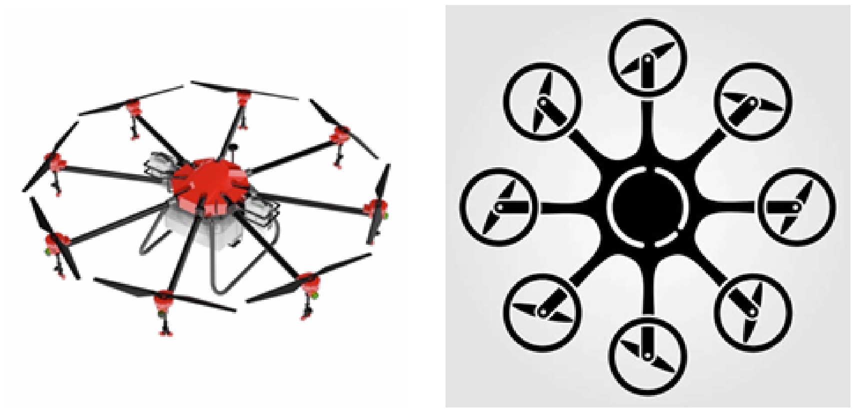

Figure 1: Octocopter

Question 1: Given the octocopter in the figures above, write the number of degrees of freedom of the system, providing a formal description of the configuration space. Besides, is the system underactuated? Briefly motivate your answer. Finally, derive the allocation matrix for the drone, considering the top view of the octocopter given in the right image above. [Hint: put the body frame aligned with one arm of the octocopter and label the propellers (counter-)clockwise].

The otocopter is equipped with eight mutually independent rotating propellers, to evaluate dofs and configuration space, we need to consider two different configurations of the system:

- **fixed on the ground**: The drone has **8 degrees of freedom**, because of 8 propellers of the drone, the configuration space is $C = T^8$.
- **Flying Body**: when flying the drone has 6 additional degrees of freedom linked to the spatial movement of the rigid body. Thus drone has a total of **14 degrees of freedom** and its configuration space is $C = T^8 \times \mathbb{R}^3 \times S^2 \times S^1$.

As for the full-actuation, we will discuss only the flight case, which is the one of most interest. While flying, **the otocopter is underactuated** because it can only generate instantaneous acceleration along the vertical axis (trust) and cannot generate instantaneous acceleration in the horizontal plane. For full-actuation, tilting wings would have to be used.

To calculate the Allocation matrix, the NED configuration was adopted for the body frame, indicating that the z-axis points inward on the page. The rotors were numbered counterclockwise.

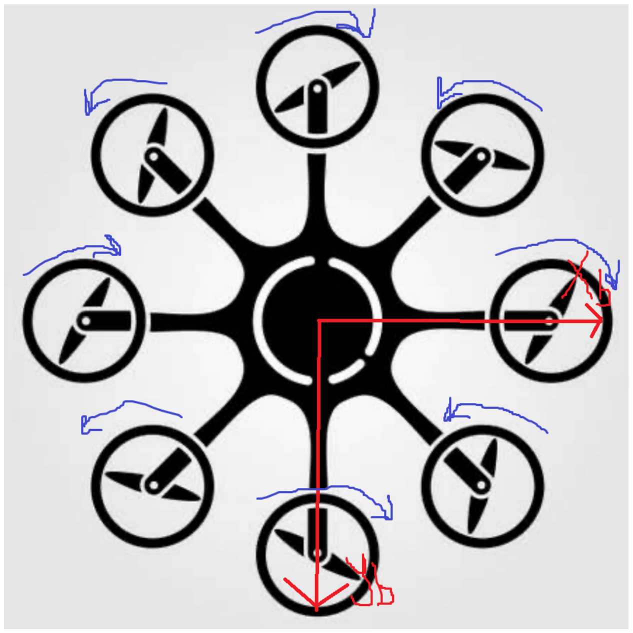

Figure 2: Octocopter

The total thrust uT and the torques generated along the x-axis, y-axis and z-axis were computed as the sum of the contribution (negative and positive) of the various rotor, :

$$\tau_x = l * (-\sin(45) * T_2 - T_3 - \sin(45) * T_4 + \sin(45) * T_6 + T_7 + \sin(45) * T_8)$$
$$\tau_y = l * (T_1 + \sin(45) * T_2 - \sin(45) * T_4 - T_5 - \sin(45) * T_6 + \sin(45) * T_8)$$
$$\tau_z = -Q_1 + Q_2 - Q_3 + Q_4 - Q_5 + Q_6 - Q_7 + Q_8$$

The relations between the thrust of a single propeller and the velocity can be approximated as follows:

$$T_i = c_T * \omega_i^2$$
$$Q_i = c_Q * \omega_i^2$$

It is obtained that the allocation matrix is as follows (the value $\frac{\sqrt{2}}{2}$ has been substituted for the $\sin(45)$):

$$A_m = \begin{bmatrix} c_T & c_T & c_T & c_T & c_T & c_T & c_T & c_T \\ 0 & -\frac{\sqrt{2}}{2} * l * c_T & -l * c_T & -\frac{\sqrt{2}}{2} * l * c_T & 0 & \frac{\sqrt{2}}{2} * l * c_T & l * c_T & \frac{\sqrt{2}}{2} * l * c_T \\ l * c_T & \frac{\sqrt{2}}{2} * l * c_T & 0 & -\frac{\sqrt{2}}{2} * l * c_T & -l * c_T & -\frac{\sqrt{2}}{2} * l * c_T & 0 & \frac{\sqrt{2}}{2} * l * c_T \\ -c_Q & c_Q & -c_Q & c_Q & -c_Q & c_Q & -c_Q & c_Q \end{bmatrix}$$

Where A_m is the allocation matrix that establishes the relationship between control inputs and propeller rotational speeds:

$$\begin{pmatrix} u_T \\ \tau_x \\ \tau_y \\ \tau_z \end{pmatrix} = A_m * \begin{pmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{pmatrix}$$

Question 2: Briefly and qualitatively describe the main differences between the hierarchical controller, the geometric controller and the passivity-based controller presented in the course. What are the main advantages and drawbacks of each control method?

We will explore the differences and characteristics of the three control methods: the hierarchical controller, the geometric controller, and the passivity-based controller. All these three types of controllers are used for position and attitude tracking of an aerial robot.

The **hierarchical controller** organises the control tasks in different levels and is structured with two loops, one external for the linear part and one internal for the angular part. The two linear subsystems are coupled by a non-linear interconnection term $\delta(\eta_{b,d}, e_\eta)$. This structure is characterised by the separation of time scales, which allows angular dynamics to be faster than linear dynamics. Feedback linearization, which is present in the inner loop, is fine if one knows the parameters well or in the presence of small errors and uncertainties, otherwise one may have problems with robustness. This controller is based on a dynamic RPY model, which is inherently prone to singularity due to the use of the RPY orientation representation. In particular, the pitch angle encounters a singularity at $\theta = \pm\frac{\pi}{2}$, where the control input becomes theoretically infinite, making it physically impractical. So the controller is not suitable for acrobatic flights.

the **geometric controller**, although more complex conceptually than the hierarchical one, operates without singularity issues, which allows for acrobatic flights. It is entirely designed in the model-configuration space R^3x$SO(3)$ using rotation matrices, avoiding any representation singularities. It offers a smooth controller by relying solely on implicit orientation representation. The drawbacks of geometrical controller are:
• the difficulty of interpreting the attitude error, because the rotation matrix is computationally heavier than the one with the Euler angles;
• Lyapunov theory can give us the exponentially convergence of the attitude error to zero only if the initial attitude error is less then 90◦
• It is also based on the linearization of feedback, like hierarchical control, which can lead to robustness problems if the model is not known.

Passivity-based controllers avoid feedback linearization of the angular part and are robust to uncertainties, at the same time, external disturbance estimation is used to compensate for unknown terms. The design of a passivity-based controller can be complex, as the optimisation of a larger number of control parameters is required due to the coupling parameters that strongly influence the dynamics (of the tracking error and there is no feedback linearization). Also like the hierarchical controller it is based on an RPY representation which makes it subject to singularities.

Common to all three techniques is that trajectory planning along the z-axis must always avoid the desired acceleration close to gravitational acceleration. In such a situation, all propellers would be switched off, rendering the robot uncontrollable.

Question 3: Briefly and qualitatively describe the differences between the ground effect and the ceiling effect.

The ground effect is an aerodynamic phenomenon that occurs when an aircraft or drone flies close to the ground. The interaction between the airflow and the earth's surface alters the pressure field around the aircraft. This effect is characterized by an increase in lift and a reduction in aerodynamic drag experienced by the aircraft's wings. This can affect the flight stability and control characteristics of the aircraft

The ceiling effect is the aerodynamic impact experienced by an aircraft or drone when it flies close to an overhead surface, such as the ceiling of a tunnel or the roof of a confined space. In this case, the air resistance is less the closer they are to the ceiling, causing the propellers to increase velocity (vacuum effect)

The most employed theoretical result for the ground effect and ceiling effect is based on the so-called potential **aerodynamic assumption**. This assumption considers the fluid:

- Inviscid
- Incompressible
- Irrotational
- Steady

In order to model the effect of a surface like the ground or the ceil, the **method of images** is employed. This method consists in placing a virtual rotor on the opposite side of the surface and at the same distance.
If one rotor is placed at $[0, 0, h]$, the other rotor is placed at $[0, 0, -h]$.
For a single rotor subject to the ground effect, it holds:

$$\frac{T_{IGE}}{T_{OGE}} = \frac{1}{1 - (\frac{\rho}{4z})^2}$$

Where T_{IGE} and T_{OGE} are the single thrust inside and outside the ground effect, ρ the radius of the propeller, and z the height of the propeller from the ground.
For a single rotor subject to the ceiling effect, it holds:

$$\frac{T_{ICE}}{T_{OCE}} = \frac{1}{1 - \frac{1}{k_1}(\frac{\rho}{z + k_2})^2}$$

Where T_{ICE} and T_{OCE} are the single thrust inside and outside the ceiling effect, k_1 and k_2 are parameters to tune.

Both ground effect and ceiling effect lead to changes in aerodynamic performance with an additional lift force due to proximity to a surface. Both effects lead to problems that are difficult to manage when the ceiling or the ground has asymmetries, in which case both ground effect and ceiling effect produce different aerodynamic effects in each rotor. The ground effect generally increases lift and efficiency, while the ceiling effect can be a challenge as the increase in thrust (a consequence of the increase in propeller speed caused by the vacuum effect) could lead to accidents. On the other hand, due to this effect, it is possible to develop more thrust for the same amount of power spent, but as mentioned, it is essential to take this effect into account to avoid accidents, so it requires more control to be used safely.

Question 4: Consider the workspace file attached as ws_homework_3_2024.mat. Within this file, you can find the values of a flight with a quadrotor with the commanded thrust (thrust) and torques (tau), the measured linear velocity (linear_vel), attitude expressed as Euler angles (attitude) and the time derivative of such angles (attitude_vel). The employed quadrotor has a supposed mass of 1.5 kg, and an inertia matrix referred to the body frame equal to diag([1.2416 1.2416 2*1.2416]). Implement yourself the momentum-based estimator of order r to estimate the external disturbances acting on the UAV during the flight. Suppose the sampling time of the estimator is equal to 1 ms. Compare the obtained estimation with the following disturbances applied during the flight:
• 0.5 N along the x- and y-axis of the world frame;
• 0.2 Nm around the yaw axis.
Compare the estimation results with different values of r. Try to answer the following questions.
• From which value of r the estimation results do not improve too much?
• Compute the real mass of the UAV from the estimated disturbance along the z-axis.

The matlab script contains the solution with a brief explanation of the code. The file is on the github repository: https://github.com/Andremorgh/FSR2024.git [1]
The file is in the folder HW3 and it is named **hw3_eserc_4.mlx.**
A high-level abstraction explanation of the implemented programming logic is given here. Run the matlab program directly to view the full operation of the code.

The 'momentum-based estimator' is a method to estimate the external forces and moments acting on a drone, using angular velocity and linear acceleration measurements provided by sensors (IMU, GPS,...). It is so called because it is based on the generalised momentum vector $q \in R^6$:

$$q = \begin{bmatrix} mI_3v & O_3 \\ O_3 & M(\eta_b) \end{bmatrix} \begin{bmatrix} \dot{p}_b \\ \dot{\eta}_b \end{bmatrix}$$

The objective of the estimator is to estimate the $\begin{bmatrix} f_e \\ \tau_e \end{bmatrix}$ through $\begin{bmatrix} \hat{f}_e \\ \hat{\tau}_e \end{bmatrix}$.
To simplify things, we want a linear relationship between the external wrench and its estimation in the Laplace domain though $G(s) \in C^{6x6}$,it is a diagonal matrix whose elements are transfer function of order r:

$$G_i(s) = \frac{(k_0)^r}{(s+c_0)^r} = \frac{k_0}{s^r + a_{r-1}s^{r-1} + ... + a_1 s + a_0}, i = 1, ..., 6$$

The denominator is Hurwitz polynomial so the poles of the transfer function have $Re(\lambda_i) < 0$, so the system is asymptotically stable. It should be noted that because we don't want amplification in a filter these functions should have unit gain, so $k_0 = c_0$ $(G(0) = \frac{k_0}{c_0} = 1)$.
The formulas in the continuous time of the r-th estimators are the following:

$$\gamma_1(t) = K_1 \left(q - \int_0^t \begin{bmatrix} \hat{f}_e \\ \hat{\tau}_e \end{bmatrix} + \begin{bmatrix} mge_3 - u_T R_b e_3 \\ C^T(\eta_b, \dot{\eta}_b)\dot{\eta}_b + Q^T(\eta_b)\tau^b \end{bmatrix} dt \right)$$

$$\gamma_i(t) = K_i \int_0^t - \begin{bmatrix} \hat{f}_e \\ \hat{\tau}_e \end{bmatrix} + \gamma_{i-1} dt, i = 2, ..., r$$

With the hypotheses that $\begin{bmatrix} \hat{f}_e(0) \\ \hat{\tau}_e(0) \end{bmatrix} = 0$ and $q(0) = 0$, this means that the estimation should start before the UAV's take-off.
Given the inputs from the sensors with a sampling time of 1 ms, the estimate of the external wrench value can be calculated using the recursive discrete-time formula described as follows:

$$\gamma_1(k+1) = \gamma_1(k) + K_1 \left(q(k+1) - q(k) - T_s \begin{bmatrix} \hat{f}_e \\ \hat{\tau}_e \end{bmatrix} - T_s \begin{bmatrix} mge_3 - u_T R_b e_3 \\ C^T(\eta_b, \dot{\eta}_b)\dot{\eta}_b + Q^T(\eta_b)\tau^b) \end{bmatrix} \right)$$

$$\gamma_i(t) = \gamma_i(k) + K_i T_s \left(- \begin{bmatrix} \hat{f}_e \\ \hat{\tau}_e \end{bmatrix} + \gamma_{i-1} \right), i = 2, ..., r$$

$$\prod_{i=j+1}^{r} K_i = c_j, j = 0, ..., r-1$$

where the discrete-time equation was used to calculate momentum :

$$q(k+1) = \begin{bmatrix} mI_3 v & O_3 \\ O_3 & M(\eta_b) \end{bmatrix} \begin{bmatrix} \dot{p}_b(k+1) \\ \dot{\eta}_b(k+1) \end{bmatrix}$$

In the composition of G(s) there are two parameters to be defined, r i.e. the filter degree r and the time constant c_0, Increasing c_0 results in faster estimation, but also results in greater under-shoot and over-shoot at the beginning.
Below are several simulations with three values of c_0 (1,10, 50, 100) but with the constant r equal to 1:

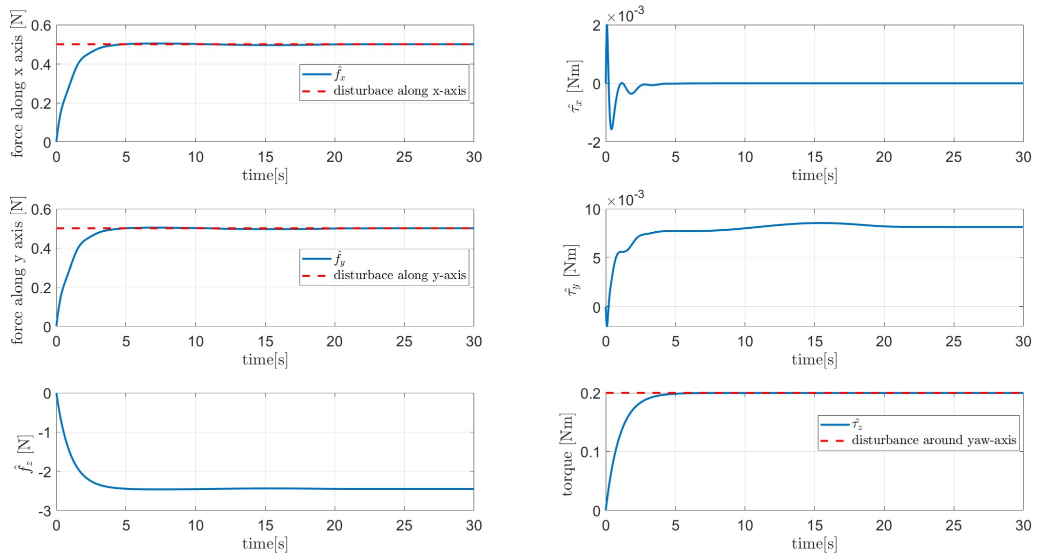

Figure 3: estimated external wrench with r = 1 and c_0 = 1

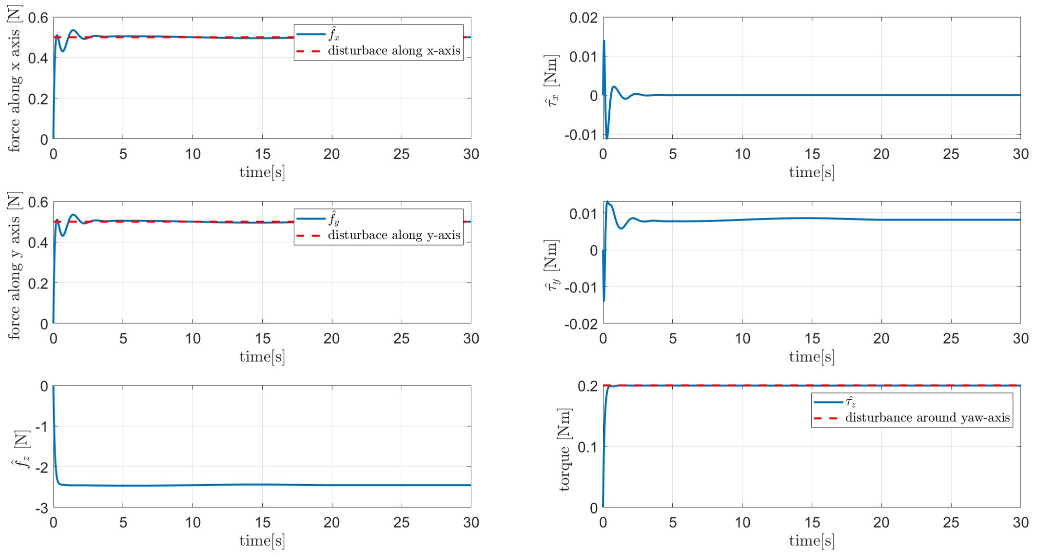

Figure 4: estimated external wrench with r = 1 and c_0 = 10

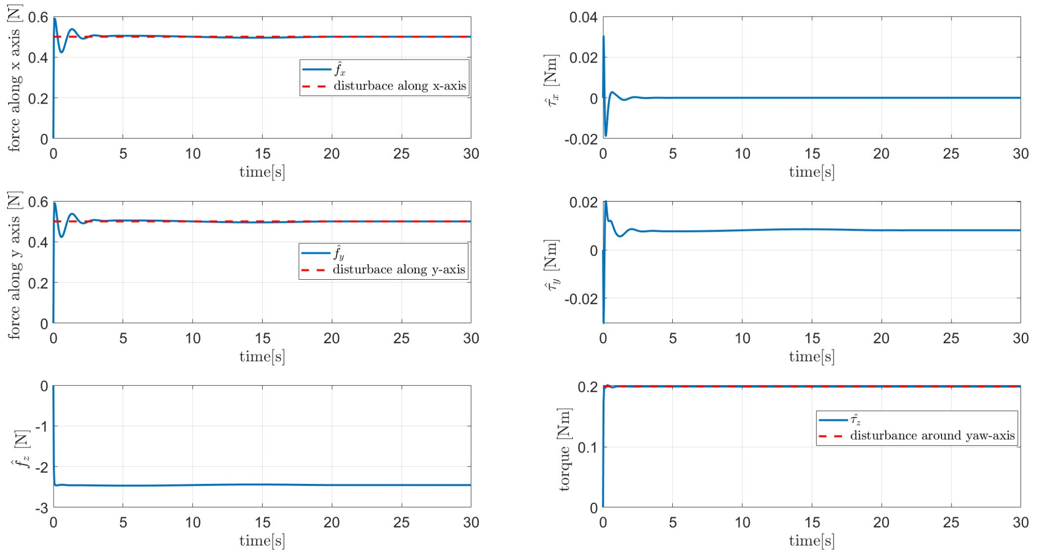

Figure 5: estimated external wrench with r = 1 and c_0 = 50

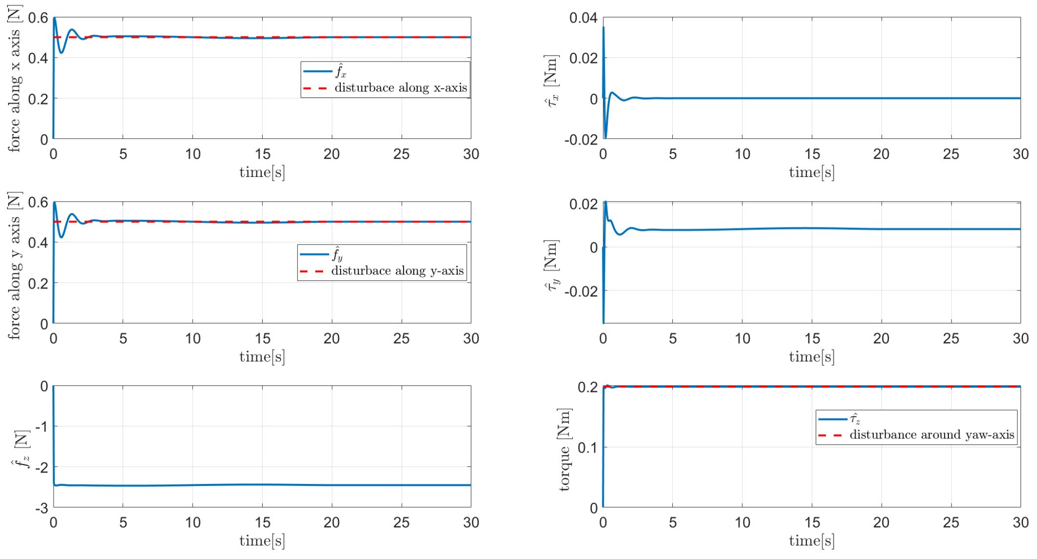

Figure 6: estimated external wrench with r = 1 and c_0 = 100

As predicted, it is evident how in the simulations with higher c_0 values the estimation is faster, but with more overshoot at the beginning, the system with $c_0 = 100$ converges in less than 3 seconds, but with more overshoot. The same estimator with $c_0 = 1$, converges in about 5 seconds, but with less overshoot than in the previous case.

In subsequent simulations, a constant value of $c_0 = 10$ will be used and the value of r (i.e. the order of the estimator) will be varied instead. With r = 1, the estimator goes to steady state around 3 seconds, giving us the correct value of the disturbance constants defined on the drone:

$$\hat{f}_e = [0.5, 0.5, -2.45]$$

$$\hat{\tau}_e = [0, 0, 0.2]$$

Increasing r results in more delay, because increasing the order of the estimator increases the number of integrators, which results in delay. So the more integrators we use for estimation, the slower the estimator becomes at the same c_0, which has to be taken into account because the estimator should be the fastest system in the control loop. Since the disturbances to be estimated are constant, the order 1 estimator would be sufficient. We can see that as r increases, undershoot and overshoot decrease, but above a certain value of r we will encounter an instability (exceeded order 100).

The following are simulations with $c_0 = 10$ and r varying (30 100 120):

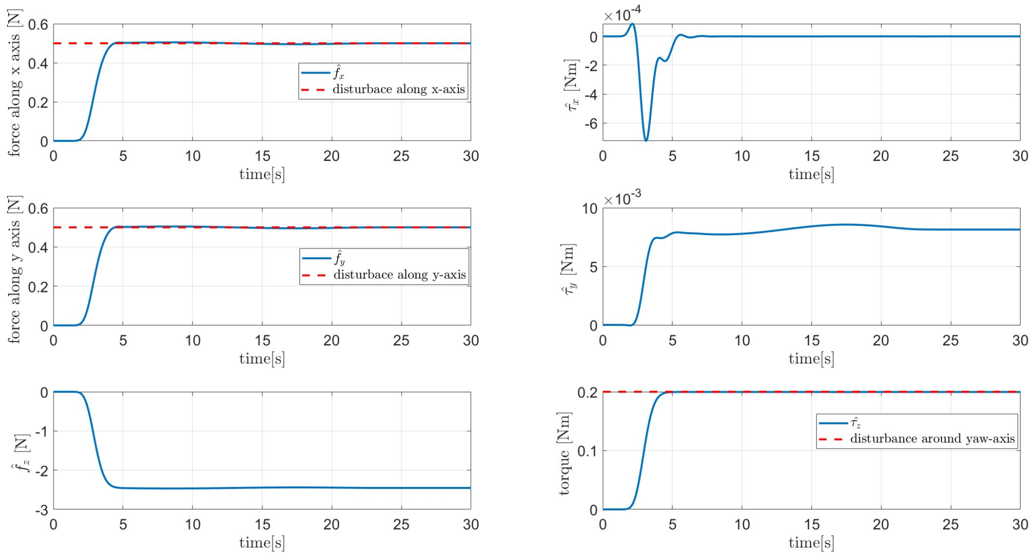

Figure 7: estimated external wrench with r = 30 and c_0 = 10

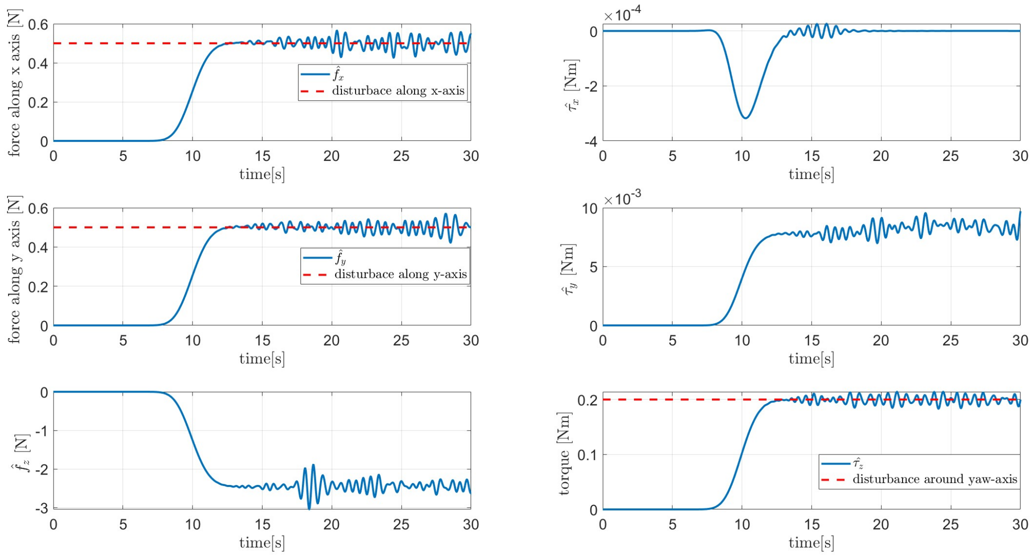

Figure 8: estimated external wrench with r = 100 and c_0 = 10

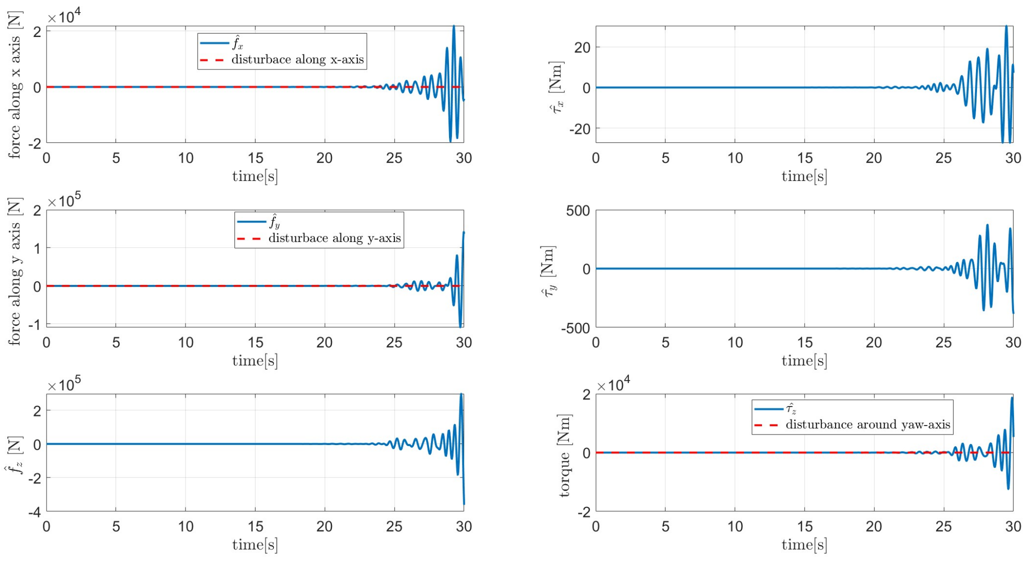

Figure 9: estimated external wrench with r = 120 and c_0 = 10

Since no external disturbances along the z-axis were given by the problem, it is possible to assume that the estimated external force along the z-axis is related to an uncertainty in the model, an incorrect knowledge of the mass. To calculate the real mass m_r, consider the equation at equilibrium of the UAV :

$$\bar{m}g - u_T R_b e_3 - \hat{f}_z = 0$$

where $\bar{m}$ is the nominal mass, $\tilde{m}$ is the estimated mass and $\hat{f}_z$ is the estimated force along the z axis:

$$\bar{m} = 1.5kg$$

$$\hat{f}_z = \tilde{m}g,$$

$$\tilde{m} = \frac{\hat{f}_z}{g} = 0.25kg$$

The real mass m_r is then:

$$m_r = \bar{m} - \tilde{m} = 1.5 - 0.25 = 1.25kg$$

Question 5: Consider the Simulink file attached as geometric_control_template.slx. Within this file, you can find a template to implement yourself the geometric control. You must fill in the inner and outer loops. Simulate the scheme and report the plots you believe are most interesting. You may add further scopes to the scheme to extract data you believe most interesting to show.

The matlab and simulink scripts contain the solution with a brief explanation of the code. The files are on the github repository:
https://github.com/Andremorgh/FSR2024.git [1]
The files are in the folder HW3 and they are named **HW3_es5.mlx** and **geometric_control_template.slx.**

A high-level abstraction explanation of the implemented programming logic is given here. Run the matlab program and the simulink model directly to view the full operation of the code.

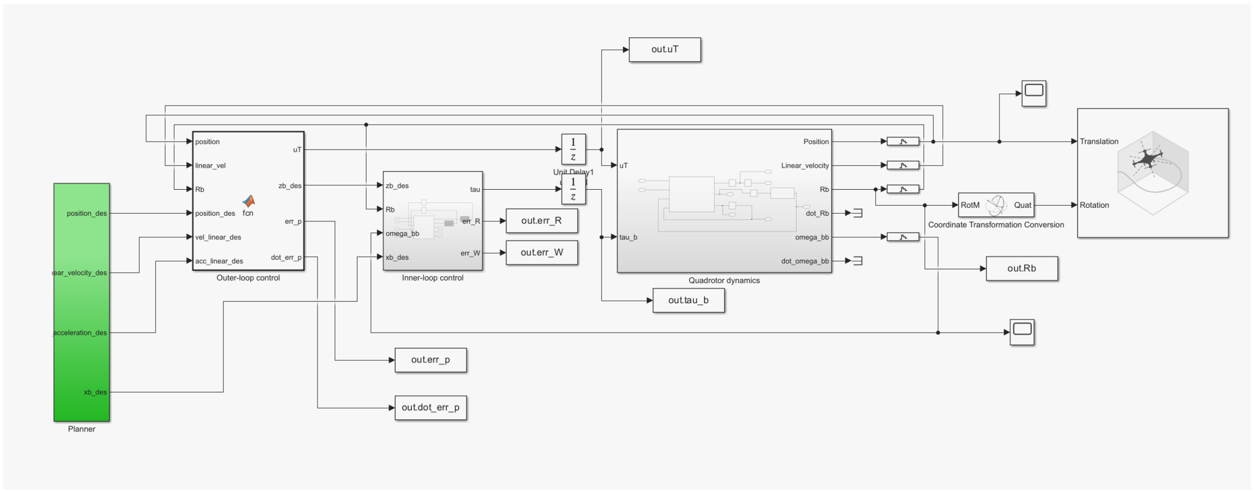

Figure 10: Symulink scheme

The geometric controller consists of two interlocking loops: the inner loop for the angular part and the outer loop that takes care of the linear part. The errors that the controller handles are as follows:

$$e_p = p_b - p_{b,d}, \quad \dot{e}_p = \dot{p}_b - \dot{p}_{b,d}$$

$$e_R = \frac{1}{2}(R_{b,d}^T R_b - R_b^T R_{b,d})^\vee, \quad e_\omega = \omega_b^b - R_b^T R_{b,d} \dot{\omega}_{b,d}^{b,d}$$

Where $\vee$ is the vee operator which is the inverse of the skew-symmetric operator and $\omega_{b,b,d}$ can be calculated as $(R_{b,d}^T \dot{R}_{b,d})^\vee$.

In the **outer loop** the linear errors are managed, u_T and $z_{b,d}$ are computed as follow:

$$u_T = -(-K_p e_p - K_v \dot{e}_p - mg\mathbf{e}_3 + m\ddot{p}_{b,d})^T R_b \mathbf{e}_3$$

$$z_{b,d} = \frac{-K_p e_p - K_v \dot{e}_p - mg\mathbf{e}_3 + m\ddot{p}_{b,d}}{\| - K_p e_p - K_v \dot{e}_p - mg\mathbf{e}_3 + m\ddot{p}_{b,d}\|}$$

K_p and K_v are the gains of the PD controller.

The **inner loop** is given input by the planner $x_{b,d}$, together with $z_{b,d}]$ We can compute $R_{b,d}$ as:

$$R_{b,d} = [S(y_{b,d})z_{b,d},\ \frac{S(z_{b,d})x_{b,d}}{|S(z_{b,d})x_{b,d}|},\ z_{b,d}]$$

Once $R_{b,d}$ has been calculated, angular errors can be handled and τ_b is computed as follows:

$$\tau_b = -K_R e_R - K_\omega e_\omega + S(\omega_b^b)I_b\omega_b^b - I_b(S(\omega_b^b)R_b^T R_{b,d}\dot{\omega}_{b,d}^{b,d} - R_b^T R_{b,d}\dot{\omega}_{b,d}^{b,d})$$

K_R and K_{omega} are gains of the PD controller.
The duration of the trajectory is 20 seconds with an additional 10 seconds to study the system's stationarity . The aerial robot toolbox was used to simulate the 3D movement of a drone. The following gains were chosen:

$$K_p = diag([50\ 50\ 400]), \quad K_v = diag([10\ 10\ 50]),$$

$$K_R = diag([10\ 10\ 1500]), \quad K_\omega = diag([10\ 10\ 200])$$

The following results were obtained:

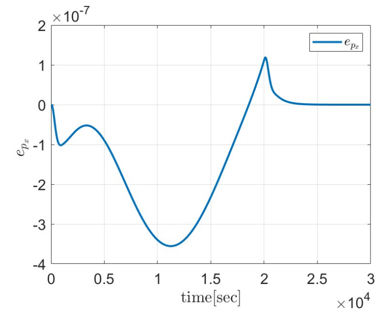
Figure 11: Error e_{p_x}

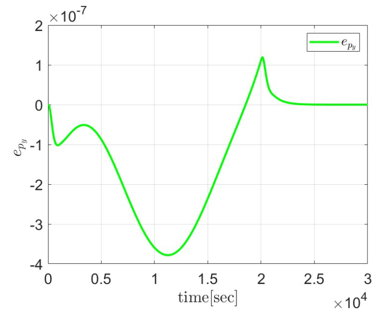
Figure 12: Error e_{p_y}

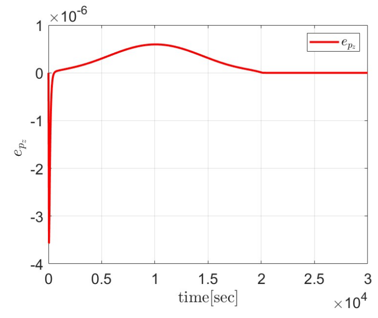
Figure 13: Error e_{p_z}

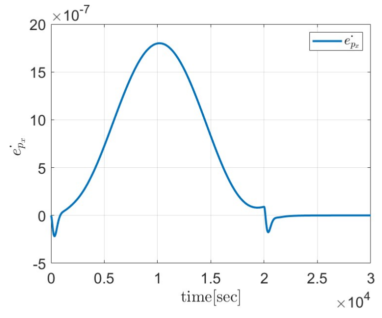
Figure 14: Error $\dot{e}_{p_x}$

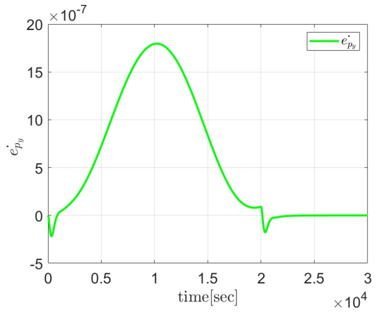
Figure 15: Error $\dot{e}_{p_y}$

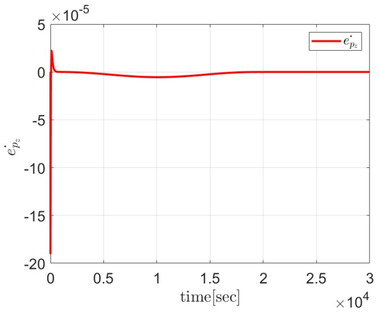
Figure 16: Error $\dot{e}_{p_z}$

Regarding the errors of the linear part we can see that there is an initial error on the z component of the order 10^-5. Once this initial error is recovered , it can be said that all three components of e_p have an error of 10^-7 during the trajectory. A similar argument applies to the components of $\dot{e}_p$. This initial error is due to the initial value of u_t, in fact at the beginning of the trajectory there is the assumption of stationarity of the drone, that corresponds to the value of u_t of around 11.77 N, while the value present in the data is 12 N. So the trajectory of the linear part along the z-component has this initial error that must be recovered, with the choice of K_p and K_v made (value of the third component higher than the other two) such recovery occurs in 1 second.

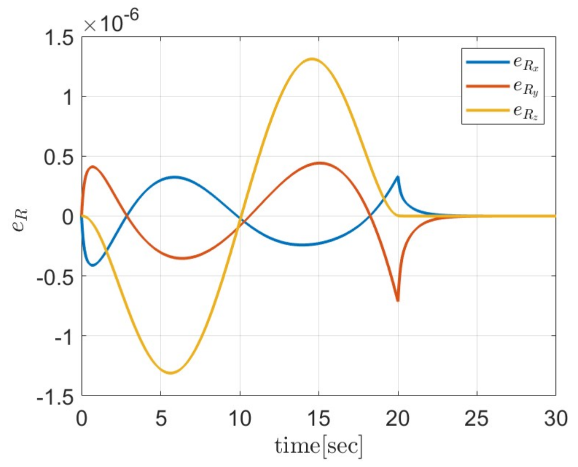

Figure 17: Unicycle v

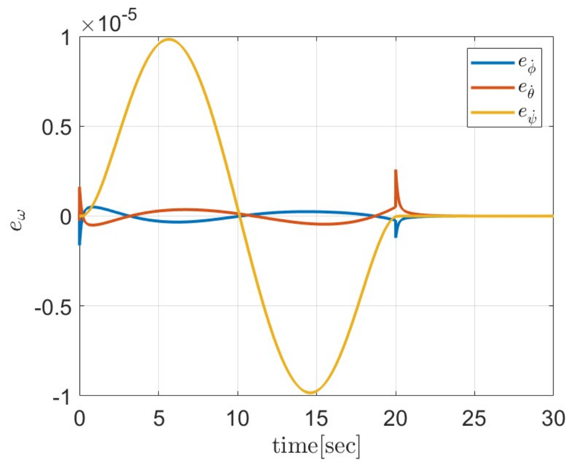

Figure 18: Unicycle w

As far as the errors e_R and e_w are concerned, there is an oscillation of the order of 10^-6 and 10^-5 respectively, with the chosen values of K_R and K_ω. It is interesting to note that only the z-component requires a greater value than the other two as all of the drone's rotation takes place around this axis. Despite the tuning, in fact, this component remains the one with the largest error in both e_R and e_w.

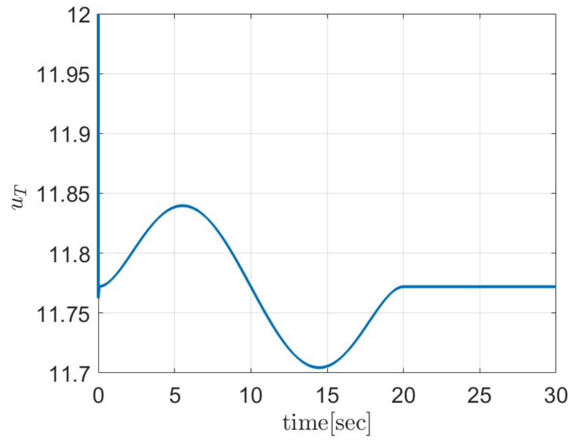

Figure 19: Unicycle v

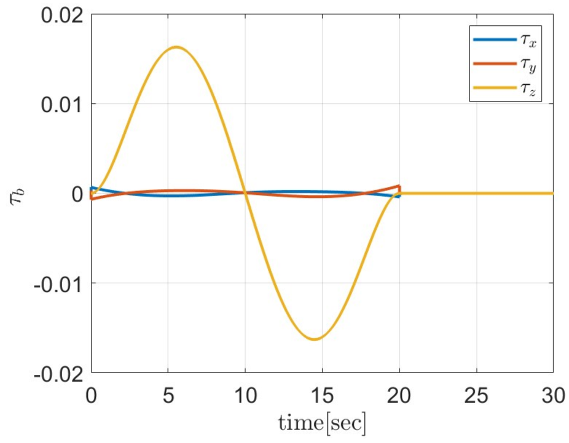

Figure 20: Unicycle w

For the total thrust, as mentioned above, it starts with a value of 12 N,in around 1 second it reaches a value of 11.77 N which corresponds to the equilibrium of forces. Subsequently there is an increase up to 11.85N which corresponds to the thrust required to reach a greater height (4 meters), followed by a decrease phase to reduce the acceleration along the z axis. Finally u_T settles at 11.77 N, again creating a balance of forces that corresponds to a stationary position of the drone in space. As far as τ_b is concerned, we note τ_z with respect to the other components, which reaches a maximum value of 0.017 Nm and goes to zero. This occurs because the desired attitude of the drone only involves rotation along the z axis.

References

[1] Andrea Morghen. FSR2024. 2024. URL: `https://github.com/Andremorgh/FSR2024.git`.