Homework 3

Fiorella Maria Romano
FIELD AND SERVICE ROBOTICS (FSR)

May 28, 2025

The MATLAB implementation files used for this homework are available in the following GitHub repository:

`https://github.com/fioreromano/Field-ServiceRobotics/tree/main/FSR_HW3`

Exercise 1

Degrees of Freedom and Configuration Space The number of degrees of freedom (DOFs) and the structure of the configuration space depend on the operating condition of the system:

- **Ground-fixed configuration:** If the base of the octocopter is fixed to the ground, the only actuated variables are the eight propeller angular velocities. In this scenario, the system possesses **eight degrees of freedom**, corresponding to the independent control of each motor. The configuration space can be formally described as $\mathbb{C} = \mathbb{T}^8$
- **Flying configuration:** When the octocopter is free to fly, the rigid-body motion in three-dimensional space must be considered in addition to the actuation of the propellers. This introduces six additional degrees of freedom-three for translation and three for rotation-associated with the vehicle's position and orientation. The overall configuration space is thus $\mathbb{C} = \mathbb{R}^3 \times \mathbb{SO}(3) \times \mathbb{T}^8$and the system has a total of **14 degrees of freedom** (8 from the propellers and 6 from the rigid-body pose).

Underactuation Despite having eight actuators, the octocopter is structurally underactuated. This is due to the fact that all propellers are mounted in a single plane and can only generate thrust along the body-fixed z-axis. As a result, the system cannot produce arbitrary forces in all directions independently: it is not possible to apply a force in the plane of the propellers without simultaneously generating a torque. Consequently, only four independent control inputs (total thrust and three torques) can be realized at any instant, while the remaining degrees of freedom are not directly actuated.

Allocation matrix The control of a coplanar octocopter's motion is achieved by mapping the angular velocities of the eight rotors, $\omega_1, \ldots, \omega_8$, to the total thrust u_T and the body-fixed torque vector $\boldsymbol{\tau}_b = [\tau_x, \tau_y, \tau_z]^\top$. This mapping is described by the *allocation matrix* $G_q \in \mathbb{R}^{4\times 8}$, which encodes the geometric configuration and rotational properties of the propulsion system.

The allocation matrix for the octocopter is 4×8 rather than 6×8 because, with coplanar and non-tiltable propellers, the system can only generate force along the body z-axis and torques about the x, y, and z axes.Therefore, the first two rows of the full 6×8 matrix, corresponding to f_x and f_y, are always zero and are typically omitted, leaving only the four controllable components (f_z, τ_x, τ_y, τ_z).

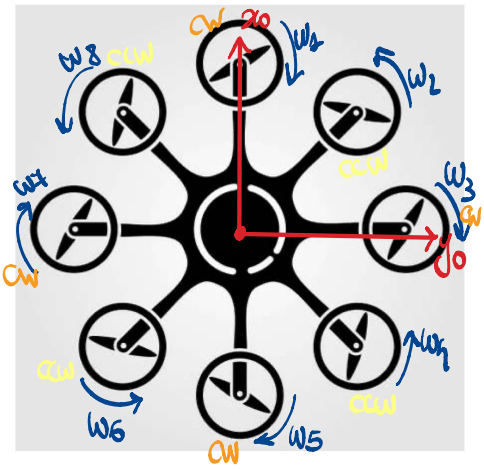

Figure 1: Octocopter

In the next equation, α_i and β_i represent the tilt angles of each propeller with respect to the body x_b- and y_b-axes, respectively. The body frame is aligned with the North-East-Down (NED) convention, with the z_b-axis pointing downward. All rotors are placed in the x_b-y_b plane, generating thrust along the $-z_b$ direction. The eight rotors are equally spaced at 45° intervals, each at a distance l from the center of mass. Rotors alternate between clockwise (CW) and counterclockwise (CCW) rotation, denoted by $k_i = \pm 1$, to balance the net yaw torque in hover.

$$\begin{bmatrix} f_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = G_q \begin{bmatrix} u_{w1} \\ u_{w2} \\ \vdots \\ u_{w8} \end{bmatrix} \implies \begin{bmatrix} f^b \\ \tau^b \end{bmatrix} = f_u(\alpha_i, \beta_i, u_{wi})$$

Force and Torque Contributions Each rotor i produces:

- **Thrust:** $T_i = c_f \, \omega_i^2$, where c_f is the thrust coefficient.
- **Drag Torque:** $Q_i = c_m \, k_i \, \omega_i^2$, where c_m is the drag coefficient and $k_i = +1$ (CCW, descending chord) or $k_i = -1$ (CW, ascending chord). The k_i coefficient alternates with the rotation direction of the propellers.

The total thrust and torque are given by:

$$f^b = \sum_{i=1}^{n} c_f |w_i| w_i z_{pi}^b \qquad \tau_b = \sum_{i=1}^{n} |w_i| w_i (-k_i c_m z_{pi}^b + c_f S(p_{pi}^b) z_{pi})$$

In our case, $\alpha_i = 0$, $\beta_i = 0$, $z_{pi} = [0, 0, -1]^T$ $\theta_i = (i-1)45°$ for rotors $i = 1, \ldots, 8$.

$$G_q = \begin{bmatrix} -c_f & \cdots & -c_f \\ -lc_f \sin\theta_1 & \cdots & -lc_f \sin\theta_8 \\ lc_f \cos\theta_1 & \cdots & lc_f \cos\theta_8 \\ c_m k_1 & \cdots & c_m k_8 \end{bmatrix} \implies G_q = \begin{bmatrix} -c_f & -c_f & -c_f & -c_f & -c_f & -c_f & -c_f & -c_f \\ 0 & -lc_f\frac{\sqrt{2}}{2} & -lc_f & -lc_f\frac{\sqrt{2}}{2} & 0 & lc_f\frac{\sqrt{2}}{2} & lc_f & lc_f\frac{\sqrt{2}}{2} \\ lc_f & lc_f\frac{\sqrt{2}}{2} & 0 & -lc_f\frac{\sqrt{2}}{2} & -lc_f & -lc_f\frac{\sqrt{2}}{2} & 0 & lc_f\frac{\sqrt{2}}{2} \\ -c_m & c_m & -c_m & c_m & -c_m & c_m & -c_m & c_m \end{bmatrix}$$

Physical Interpretation

- **Thrust (u_T):** All rotors contribute equally to the vertical force, which acts along $-z_b$ (downward in NED).

$$T_i = c_f\,\omega_i^2 \quad \Rightarrow \quad u_T = -(T_1 + T_2 + T_3 + T_4 + T_5 + T_6 + T_7 + T_8)$$

- **Roll (τ_x):** $\tau_x = -l\sin(45°)T_2 - lT_3 - l\sin(135°)T_4 - l\sin(225°)T_6 - lT_7 - l\sin(315°)T_8$
- **Pitch (τ_y):** $\tau_y = lT_1 + l\cos(45°)T_2 + l\cos(135°)T_4 + lT_5 + l\cos(225°)T_6 + l\cos(315°)T_8$
- **Yaw (τ_z):** The alternating sign in the last row, determined by k_i, reflects the alternating drag torque due to the rotation direction of each rotor.

$$Q_i = c_m k_i \omega_i^2 \quad \Rightarrow \quad \tau_z = -Q_1 + Q_2 - Q_3 + Q_4 - Q_5 + Q_6 - Q_7 + Q_8$$

The resulting allocation matrix reflects the physical limitations and symmetry of the propulsion system. With fixed, coplanar propellers, the drone can only generate thrust along the body z-axis, while torques arise from both the lever arms of each rotor and their spinning direction.

Exercise 2

Ground Effect The ground effect occurs when a UAV flies close to the ground, typically within about one rotor diameter. The presence of the ground alters the airflow around the rotors, as the air is reflected by the ground and interacts with the UAV, affecting its stability. This phenomenon leads to three key phenomena: (1) thrust increase up to 20% at constant power, (2) reduced induced power requirements during hover, and (3) improved passive stability through damping of vertical oscillations especially during takeoff and landing.

One common technique for modeling the ground effect is the *method of images*, which considers a virtual mirror rotor below the ground plane. The thrust ratio between flight inside the ground effect (IGE) and outside the ground effect (OGE), assuming constant power, is given by:

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

where ρ is the rotor radius and z is the vertical distance to the ground. This effect peaks at $z \approx \rho$ and becomes negligible beyond $z > 2\rho$.

Within the Ground Effect zone, the thrust is significantly increased due to the ground interaction. To maximize energy efficiency during flight close to the ground, it is advisable to reduce the total thrust output, as the ground effect naturally increases lift. This adjustment can result in significant energy savings, especially during takeoff and landing phases.

When multiple rotors are present, the ground effect is influenced by the distance between the rotors. As the distance increases, the mutual aerodynamic influence decreases, and the behavior approaches that of a single rotor. The extended formula takes into account these interactions, which can lead to non-uniform thrust distribution and affect the UAV's stability.

Ceiling Effect The ceiling effect occurs when a UAV flies close to an overhead surface, such as a ceiling or structure, typically within a distance comparable to the rotor diameter. In this situation, the airflow generated by the propellers is constrained by the surface above, altering the aerodynamic environment.

When a UAV approaches the ceiling, three primary effects occur: (1) the restricted airflow above the propellers creates increased pressure, leading to significantly higher thrust output; (2) this results in unintended upward acceleration and additional lift generation; (3) the altered aerodynamic conditions frequently cause flight instability, particularly in attitude control.

This phenomenon must be taken into account to avoid collisions. The increase in thrust is due to the so-called *vacuum effect*, which decreases propeller drag, allowing the propellers to rotate faster at the same power.

A model for the thrust ratio under ceiling effect is given by:

$$\frac{T_{CE}}{T_{OGE}} = \frac{1}{1 - \frac{1}{k_1}\left(\frac{\rho}{z+k_2}\right)^2}$$

where ρ is the rotor radius, z is the vertical distance to the ceiling,k_1 and k_2 are positive constants to be properly tuned.

Table 1: Comparison between Ground Effect and Ceiling Effect

Aspect	**Ground Effect**	**Ceiling Effect**
Location	Near ground (below UAV)	Near ceiling (above UAV)
Airflow alteration	Reduced induced drag, enhanced lift	Restricted upward airflow, increased pressure
Effect on thrust	Increased thrust at constant power	Increased thrust, possible overshoot
Impact on control	Improved stability near ground	Potential instability and loss of altitude control
Modeling approach	Method of images (virtual rotor)	Empirical models with tuning constants

In summary, while both the ground and ceiling effects result from altered airflow near surfaces, the ground effect increases lift near the ground, and the ceiling effect increases lift near an upper boundary, each requiring specific control strategies to maintain stable and efficient flight.

Exercise 3

A momentum-based estimator was implemented to analyze the UAV's flight behavior. Its primary objective is to estimate external disturbances acting on the system, including effects from inaccurate modeling of aerodynamic forces, unknown variations in mass or inertia, and physical interactions with the environment.

To formulate the estimator, we first introduce the generalized momentum vector $q \in \mathbb{R}^6$, then we differentiate it in order to get the wrench expression:

$$\begin{bmatrix} f_e \\ \tau_e \end{bmatrix} = \dot{q} - \begin{bmatrix} mge_3 - u_T R_b e_3 \\ C^T(\eta_b, \dot{\eta}_b)\dot{\eta}_b + Q^T(\eta_b)\tau_b \end{bmatrix} \tag{1}$$

We assume a linear relationship between the real external wrench and the estimated one in the Laplace domain:

$$\mathcal{L}[\begin{bmatrix} \hat{f}_e \\ \hat{\tau}_e \end{bmatrix}] = \mathcal{G}(s)\mathcal{L}[\begin{bmatrix} f_e \\ \tau_e \end{bmatrix}]$$

where $\mathcal{G}_i(s)$ is chosen as an $r-th$ order low-pass filter with repeated poles at $-c_0$:

$$\mathcal{G}_i(s) = \frac{k_0}{s^r + c_{r-1}s^{r-1} + \cdots + c_1 s + c_0}$$

with $i = 1 \dots 6,\ c_r, k_0 > 0$. Since we don't want a scaled solution, we suppose $k_0 = c_0$. This filter structure is employed because it guarantees system stability due to its poles being located in the left half of the complex plane. In particular, the use of repeated poles at the same location is motivated by the presence of constant or slowly varying external disturbances. Repeated poles near the origin allow the estimator to integrate persistent signals over time, making it well-suited for rejecting constant biases or low-frequency disturbances. Additionally, this configuration provides a smooth attenuation of high-frequency noise, effectively reducing measurement noise and enhancing the robustness of the disturbance estimation.

In the code, the transfer function $\mathcal{G}_i(s) = \frac{k_0}{(s+c_0)^r}$ is computed and then the coefficients c_r are extracted.

When we go back in the time domain, we obtain a system of recursive differential equations defining the variables γ_i, and recalling (1), we get:

$$\gamma_1 = k_1(q - \int_0^t \begin{bmatrix} \hat{f_e}(t) \\ \hat{\tau_e}(t) \end{bmatrix} + \begin{bmatrix} mge_3 - u_T R_b e_3 \\ C^T(\eta_b, \dot{\eta}_b)\dot{\eta}_b + Q^T(\eta_b)\tau_b \end{bmatrix} dt)$$

$$\gamma_i = k_i(q - \int_0^t \begin{bmatrix} \hat{f_e}(t) \\ \hat{\tau_e}(t) \end{bmatrix} + \gamma_{i-1}\ dt) \qquad i = 2 \dots r$$

At the end, we get that the estimation is given by the last γ_r:

$$\begin{bmatrix} \hat{f_e}(t) \\ \hat{\tau_e}(t) \end{bmatrix} = \gamma_r(t) \qquad\qquad \begin{bmatrix} \hat{f_e}(0) \\ \hat{\tau_e}(0) \end{bmatrix} = 0\ , \quad q(0) = 0$$

Choice of c_0 The filter gain c_0 and the filter order r play a pivotal role in the performance of the momentum-based estimator. Increasing the gain c_0 shifts the filter poles further to the left in the complex plane, resulting in a faster estimator response and reduced delay. However, this also leads to larger oscillations in the estimator's transient response, as a higher c_0 makes the filter more reactive, amplifying overshoot and the amplitude of initial oscillations before reaching steady state. Additionally, a higher c_0 permits more high-frequency noise to pass through. Conversely, increasing the filter order r improves the attenuation of high-frequency noise, yielding a smoother estimate. Nevertheless, a higher order introduces greater phase lag and slows the estimator's responsiveness to changes in the actual disturbance. Consequently, there is a trade-off: low values of c_0 and r produce a noisy but fast estimate, whereas higher values result in a smoother but delayed estimation.

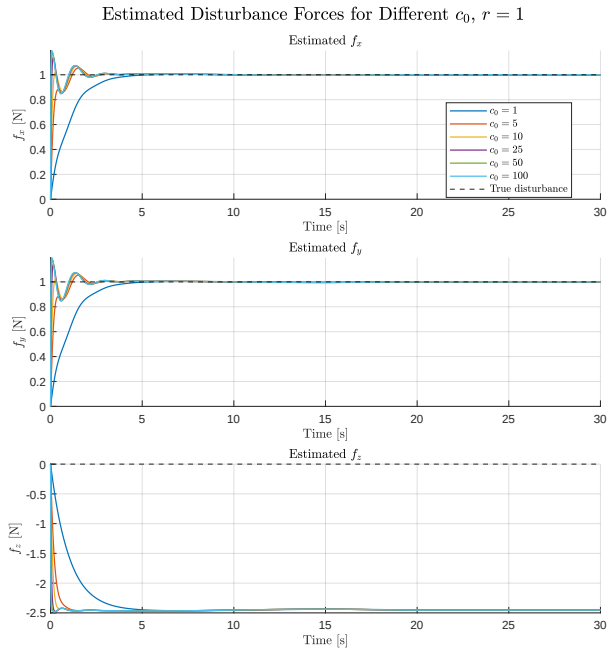

Figure 2: Estimated Forces for different c0 and r=1

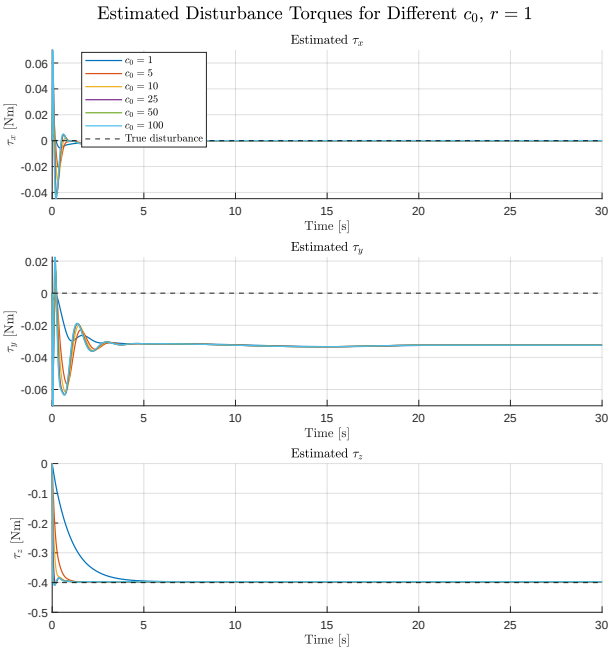

Figure 3: Estimated Torques for different c0 and r=1

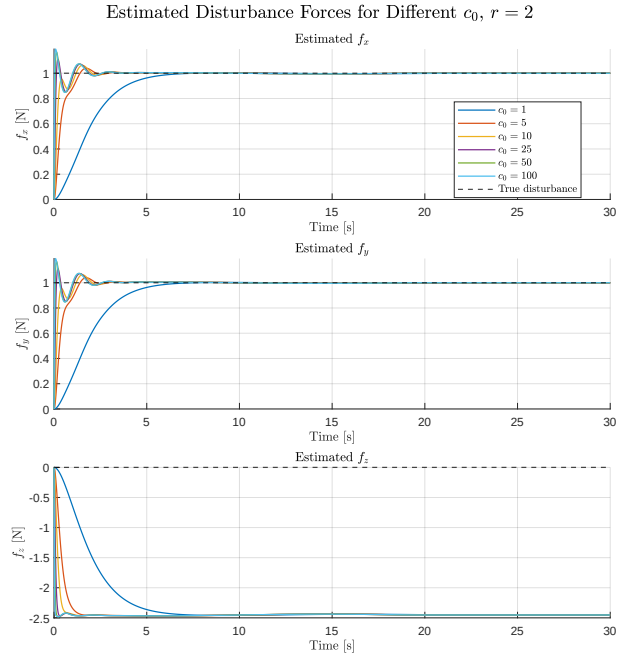

Figure 4: Estimated Forces for different c0 and r=2

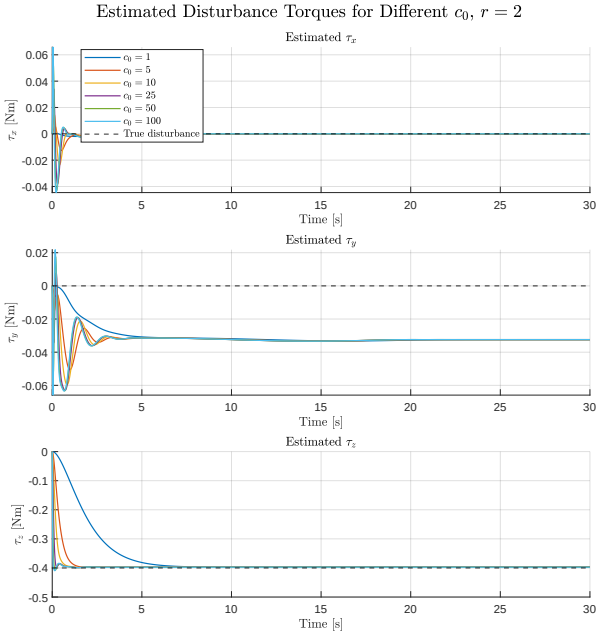

Figure 5: Estimated Torques for different c0 and r=2

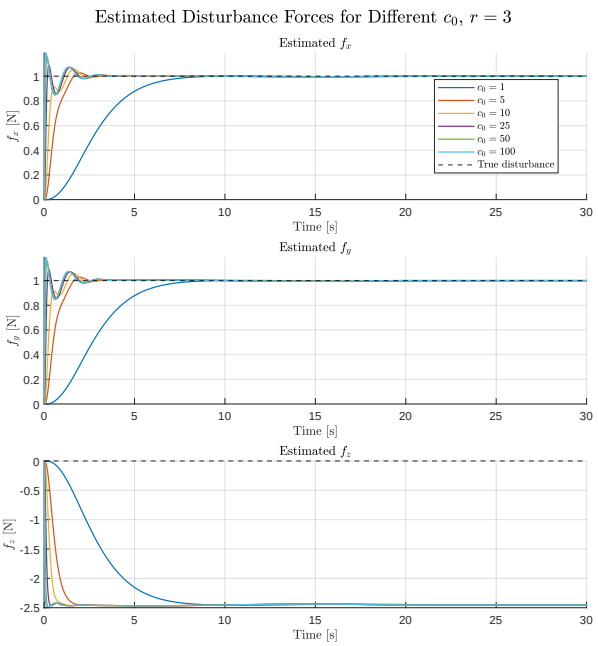

Figure 6: Estimated Forces for different c0 and r=3

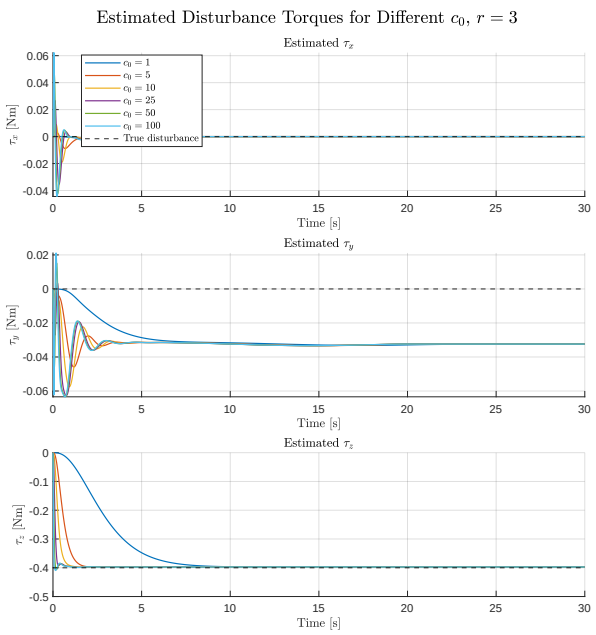

Figure 7: Estimated Torques for different c0 and r=3

Parameter	Response Speed	Noise Attenuation	Estimation Lag	Mean Error Effect
$c_0 \uparrow$	Faster	Lower	Reduced	Improves up to a point, then worsens due to noise
$r \uparrow$	Slower	Higher	Increased	Improves up to a point, then worsens due to lag

Table 2: Qualitative effects of increasing the filter gain c_0 and the filter order r in the momentum-based estimator.

Based on the analysis (Figures 2 to 7), I choose to **fix** $c_0 = 10$, as the plots indicate that this value achieves a sufficiently fast estimation response without causing excessive overshoot. This selection represents a balanced compromise between estimator speed and transient behavior.

Filter's order evaluation As shown in the Figure 8, the estimation error converges to zero at different rates depending on the filter order. However, the estimation quality is better assessed by examining the Mean Absolute Error (MAE) depicted in Figure 10. Specifically, as the filter order r increases, the average estimation error also rises-from approximately 0.004 for $r = 1$ to about 0.009 for $r = 5$ in the case of τ_z, and from roughly 0.008 to 0.02 for f_x and f_y. Furthermore, Figure 11 illustrates that the convergence time (computed as the 3% settling time) increases with the filter order, highlighting the trade-off between noise attenuation and estimator responsiveness.

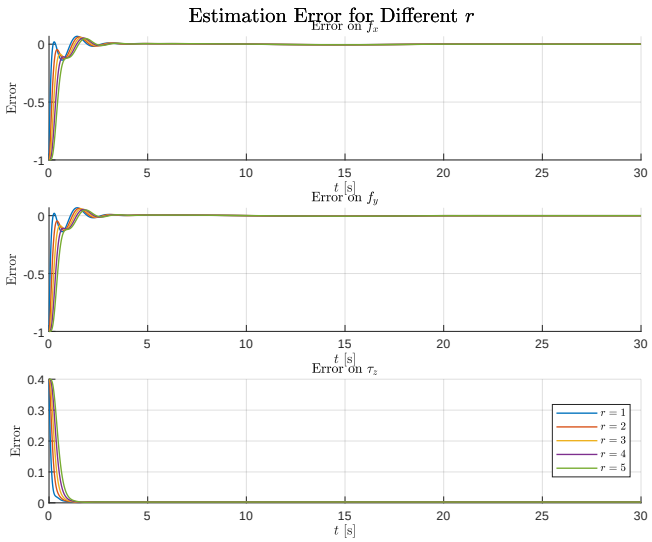

Figure 8: Error for different r, $c_0 = 10$

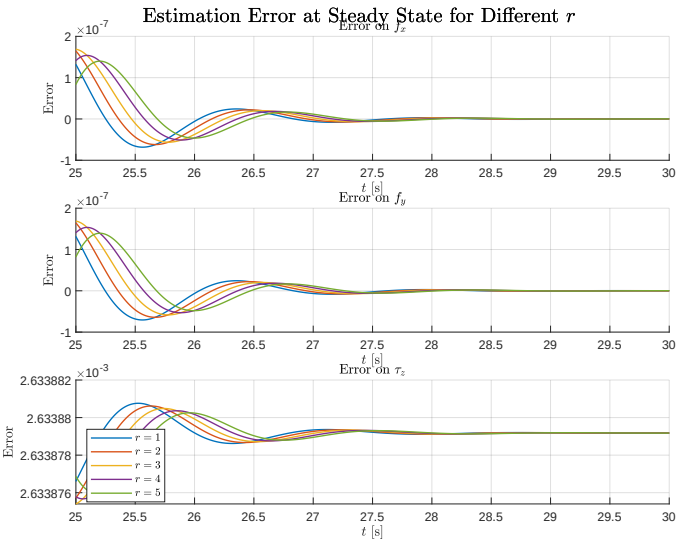

Figure 9: Steady State error for different r, $c_0 = 10$

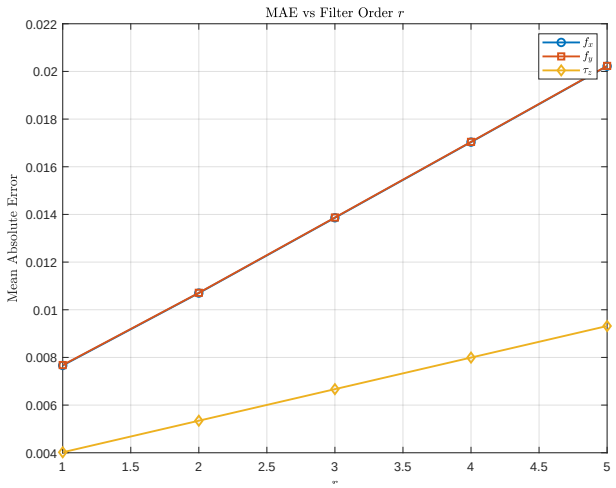

Figure 10: Mean Absolute Error for different r, $c_0 = 10$

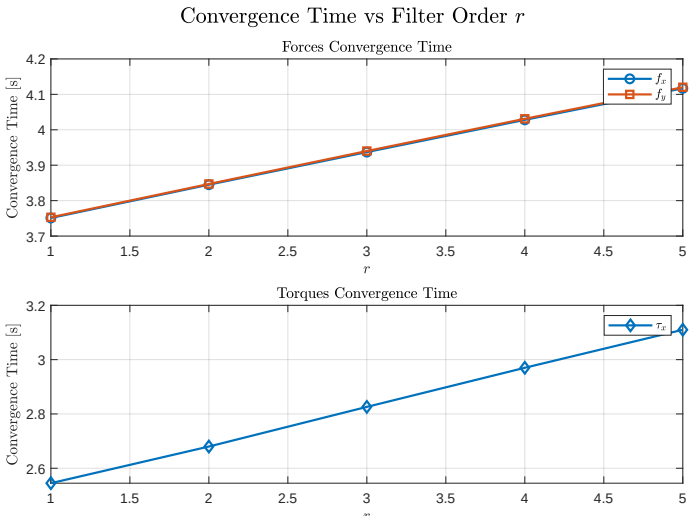

Figure 11: Convergence time for different r, $c_0 = 10$

Filter Order r	**Convergence Time f_x [s]**	**Convergence Time f_y [s]**	**Convergence Time τ_z [s]**
1	3.751	3.753	2.545
2	3.845	3.847	2.680
3	3.937	3.940	2.826
4	4.028	4.031	2.970
5	4.117	4.120	3.110

Table 3: Convergence times for force components f_x, f_y and torque component τ_z at different filter orders r.

Filter Order r	**MAE** f_x **[s]**	**MAE** f_y **[s]**	**MAE** τ_z **[s]**	$\mathbf{max}_{err} f_x$	$\mathbf{max}_{err} f_y$	$\mathbf{max}_{err} \tau_z$
1	0.0077	0.0077	0.0040	0.132517e-6	0.131980e-6	0.002633880761213
2	0.0107	0.0107	0.0053	0.164746e-6	0.164600e-6	0.002633880615723
3	0.0139	0.0139	0.0067	0.168812e-6	0.168662e-6	0.002633880483493
4	0.0170	0.0170	0.0080	0.153803e-6	0.153426e-6	0.002633880363335
5	0.0202	0.0202	0.0093	0.139914e-6	0.139336e-6	0.002633880254139

Table 4: MAE and maximum errors at different filter orders r.

From the analysis of the tables and graphs, two key trends emerge:

- As the filter order r increases:
  - The convergence time grows linearly (from 2.545 s for $r = 1$ to 3.110 s for $r = 5$ in τ_z).
  - The mean absolute error (MAE) worsens (from 0.004 to 0.009 for τ_z), while the noise decreases.
- Optimal trade-off:
  - $r = 3$ offers the best compromise:
    - **Smoothness**: Significant noise reduction compared to $r = 1$.
    - **Contained delay**: 2.826 s for τ_z (only +0.28 s vs. $r = 1$).
    - **Acceptable MAE**: 0.0067 for τ_z.
  - Beyond $r = 3$, benefits become marginal: MAE increases (+34% from $r = 3$ to $r = 5$) and delay becomes excessive.

Therefore, $r = 3$ is the optimal choice for this system.

Mass Estimation The real mass of the UAV can be estimated by analyzing the disturbance force component along the vertical axis. Assuming the UAV is in a steady hovering condition with negligible vertical acceleration, the disturbance force along z, denoted as $f_{z,\text{est}}$, primarily compensates for the gravitational force acting on the UAV.

We can therefore compute an estimate of the additional mass component Δm_{UAV} affecting the UAV dynamics:

$$\Delta m_{\text{UAV}} = \frac{f_{z,\text{est}}(\text{end})}{g} = -0.25kg$$

Therefore, the real mass od the drone is $m_{real} = m_{model} + \Delta m_{\text{UAV}} = 1.25kg$

Exercise 4

The geometric control approach is based on the idea of avoiding local parametrizations of orientation (such as Euler angles or quaternions) and instead working directly on the nonlinear manifold SO(3). This avoids singularities and ambiguities, and leads to globally valid and coordinate-free control laws.

The geometric controller decouples the dynamics into an outer loop that computes thrust and desired orientation from position errors, and an inner loop that ensures accurate attitude tracking.

OUTER LOOP

The outer loop computes the position error e_p and its time derivative $\dot{e}_p$, based on the difference between actual and desired position and velocity. These quantities are then used to generate the total thrust u_T and the desired body z-axis direction $z_{b,d}$ as follows:

$$u_T = -(-k_p e_p - k_v \dot{e}_p - mge_3 + m\ddot{p}_d)^\top R_b e_3, \qquad z_{b,d} = -\frac{-k_p e_p - k_v \dot{e}_p - mge_3 + m\ddot{p}_d}{\| - k_p e_p - k_v \dot{e}_p - mge_3 + m\ddot{p}_d\|} \quad (2)$$

where $e_3 = [0,\ 0,\ 1]^\top$, k_p and k_v are the gain matrices for position and velocity errors respectively, and $\ddot{p}_d$ is the desired acceleration from the trajectory planner. The outer loop thus provides both the magnitude of the total thrust and the desired orientation of the thrust direction, encoded in $z_{b,d}$, which is later used by the inner loop to construct the full desired attitude $R_{b,d}$.

INNER LOOP

The inner loop computes the torque τ_b to be applied to the quadrotor, using the following control law:

$$\tau_b = -k_R e_R - k_\omega e_\omega + S(\omega_b^b) I_b \omega_b^b - I_b \left(S(\omega_b^b) R_b^T R_{b,d} \omega_{b,d}^{b,d} + R_b^T R_{b,d} \dot{\omega}_{b,d}^{b,d} \right) \quad (3)$$

The desired rotation matrix $R_{b,d}$ is built from the desired body z-axis $z_{b,d}$ (from the outer loop), by first computing $y_{b,d}$ and then obtaining $x_{b,d}$ via orthonormalization:

$$R_{b,d} = \left[y_{b,d} \times z_{b,d},\ \frac{z_{b,d} \times x_{b,d}}{||z_{b,d} \times x_{b,d}||},\ z_{b,d} \right] \quad (4)$$

with x_c a reference direction. This reconstruction ensures $R_{b,d} \in \mathrm{SO}(3)$.

Results

The trajectory starts from an initial position $p_i = [0\ \ 0\ \ 1]$, $\psi_i = 0$ to reach a final configuration $p_f = [1\ \ 1\ \ 4]$, $\psi_f = 0.3490\ rad$. The following gain matrices have been chosen:

$$k_p = diag([20,\ 20,\ \ 100]), \quad k_v = diag([20,\ 20,\ \ 100])$$

$$k_R = diag([10,\ 10,\ \ 100]), \quad k_\omega = diag([10,\ 10,\ \ 100])$$

The control gains along the z-axis are set higher to effectively counteract the gravitational force acting vertically. In contrast, the gains in the horizontal xy-plane are tuned to balance accuracy and responsiveness, ensuring stable and precise lateral motion.

Regarding orientation control, the gains prioritize the yaw axis due to the higher moment of inertia around the z-axis, which requires stronger corrective actions to maintain desired heading stability.

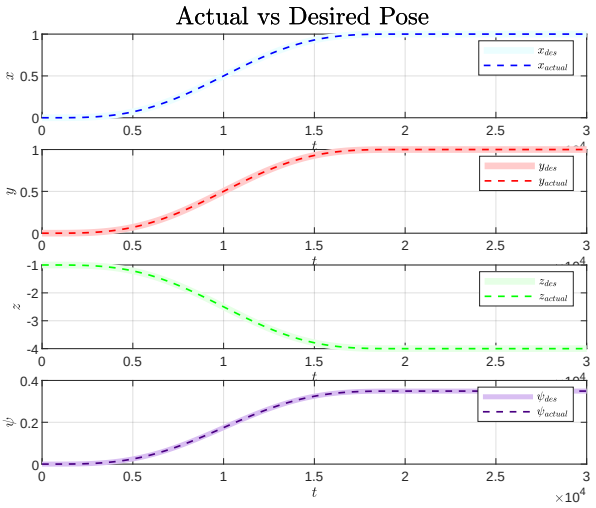

Figure 12: Actual vs Desired values for position and yaw angle

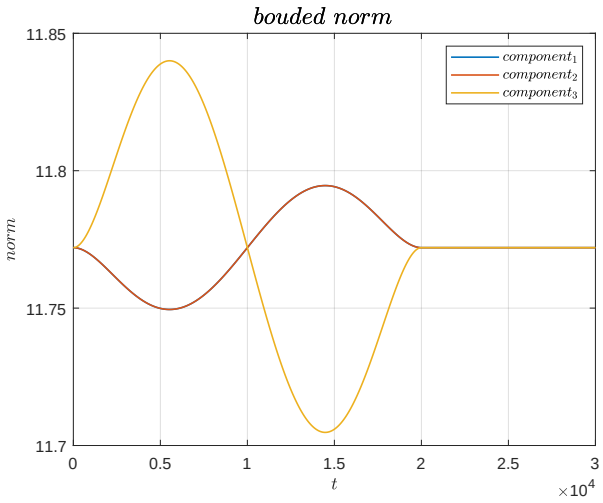

Figure 13: $\|-mge_3 + m\ddot{p}_{b,d}\| < \mathrm{K}$

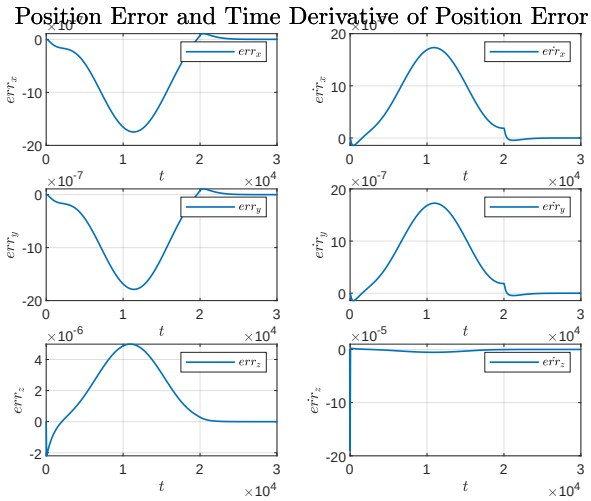

Figure 14: Errors on linear position and linear velocity

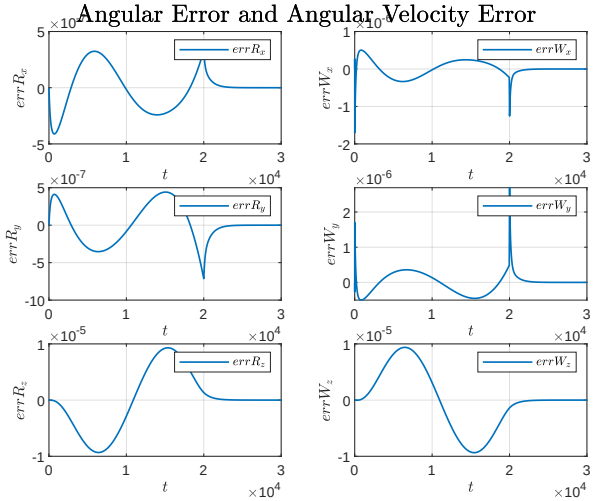

Figure 15: Errors on angular position and angular velocity

Tracking Errors The translational and rotational tracking errors exhibit excellent convergence and well-behaved transient dynamics. The position error e_p and velocity error $\dot{e}_p$ show smooth responses, with the largest initial deviation along the z-axis due to displacement from $[0,\ 0,\ 1]^\top$ to $[1,\ 1,\ 4]^\top$. The system displays slightly underdamped vertical dynamics with minimal overshoot, indicating proper tuning of the k_p gains. Velocity errors decay rapidly, with peaks between 10^{-6} and 10^{-5} m/s, confirming effective tracking. Similarly, the attitude error e_R and angular velocity error e_ω converge quickly to zero, with small bounded transients in the x and y directions (within 10^{-7} to 10^{-6} rad) and angular velocity peaks below 10^{-5} rad/s. These results demonstrate that the k_R and k_ω gains are well tuned, ensuring precise and stable rotational tracking alongside reliable outer-loop performance.

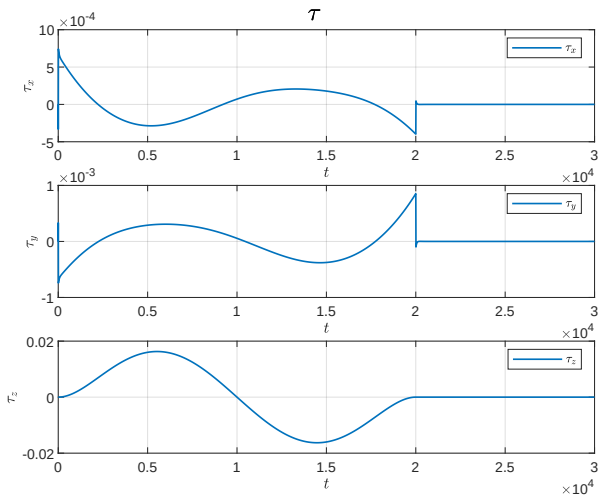

Figure 16: τ

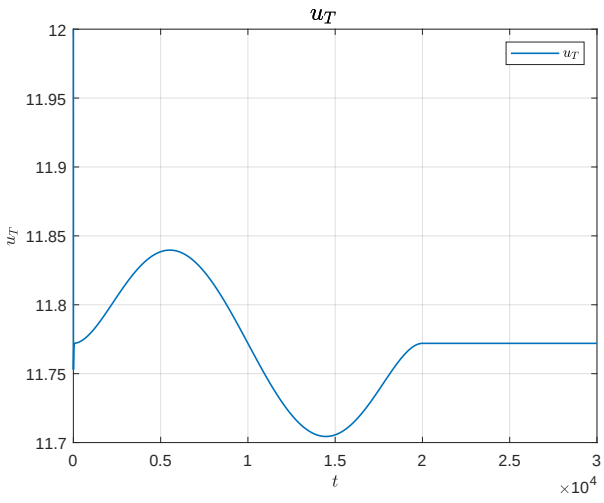

Figure 17: u_T

Inputs The torque commands exhibit smooth and bounded behavior across all axes. The τ_x and τ_y components remain within $[-5 \cdot 10^{-4},\ 10^{-3}]$ Nm, consistent with the expected effort for roll and pitch stabilization, while τ_z reaches up to $2 \cdot 10^{-2}$ Nm due to the more significant yaw dynamics in the reference trajectory.

The total thrust u_T shows a transient response, initially rising above 12 N during ascent and then slightly decreasing before stabilizing around 11.78 N. This final value aligns with the projection of the gravitational force onto the body z-axis, which varies during flight. The smooth, continuous evolution of both torque and thrust confirms that the controller maintains stability and accuracy without actuator saturation or instability, even during dynamic maneuvers.

For this kind of controller, it is essential to ensure that the desired acceleration remains bounded, as the computation of the desired thrust direction $z_{b,d}$ becomes ill-posed when the net acceleration exceeds the physical capabilities of the system. The geometric controller relies on the assumption:

$$\|-mge_3 + m\ddot{p}_{b,d}\| < \text{constant}$$

where this constant is determined by the maximum collective thrust that the UAV's propellers can generate. It's possible to observe that this condition is fulfilled; in fact, in Figure 13 we can see that the norm remains bounded.

Exercise 5

The Voliro approach is a control strategy for drones with tilting rotors. Instead of controlling each rotor in a fixed direction, this method allows the rotors to tilt, giving the drone much more flexibility and agility. By adjusting both the speed and the angle of each rotor, the drone can move and rotate in any direction. The control system calculates the best combination of rotor speeds and tilting angles to achieve the desired movement.

The control inputs u_ω and α can be computed by transforming the nonlinear allocation problem into a linear one via a variable transformation:

$$u_{v,i} = c_f u_{\omega,i} c_i, \quad u_{l,i} = c_f u_{\omega,i} s_i \qquad i = 1, \ldots, 4$$

where c_f is the thrust coefficient, and c_i, s_i are the cosine and sine of the tilting angle for rotor i. This leads to a static allocation matrix $G_{q,static}$.

The actuator commands $u(\alpha, u_\omega)$ can be retrieved in the following way:

$$u_{\omega,i} = \frac{1}{c_f}\sqrt{u_{l,i}^2 + u_{v,i}^2}, \qquad \alpha_i = \text{atan2}(u_{l,i}, u_{v,i}), \qquad i = 1, \ldots, 4 \tag{5}$$

The Simulink block function implements a feedback linearization controller that computes the position and orientation errors between the current and desired states, then applies proportional-derivative (PD) control laws to generate the desired force $\mathbf{f}^b$ and torque $\boldsymbol{\tau}^b$, along with a feed-forward action on the desired velocity. These auxiliary control inputs are concatenated into an input vector $u_{\text{auxiliary}}$ and mapped to actuator commands by solving a linear allocation equation. It is then inverted as $u = A^\dagger(-b + u_{\text{auxiliary}})$ where $A^\dagger$ denotes the Moore-Penrose pseudoinverse of the allocation matrix A.

The auxiliary control inputs $u_{\text{auxiliary}}$ are composed of:

$$\begin{aligned} v &= -K_p\,\mathbf{e}_p - K_d\,\dot{\mathbf{e}}_p + \mathbf{a}_{\text{linear,des}} \\ w &= -K_R\,e_R - K_w\,e_\omega + I_b\left(\text{skew}(\boldsymbol{\omega}_{bb})R_b^\top R_{b,\text{des}}\boldsymbol{\omega}_{bb,\text{des}} + R_b^\top R_{b,\text{des}}\dot{\boldsymbol{\omega}}_{bb,\text{des}}\right) \end{aligned}$$

with the gain matrices set as $K_p = \text{diag}(50, 50, 100)$, $K_d = \text{diag}(10, 10, 50)$, $K_R = \text{diag}(100, 100, 100)$, K $\text{diag}(10, 10, 50)$.

Nullspace Exploitation Since the allocation matrix A has a nontrivial nullspace due to actuator redundancy, the controller exploits this property to improve actuator command feasibility and efficiency. The general solution to the allocation is expressed as:

$$u = A^\dagger(-b + u_{\text{auxiliary}}) + Nz$$

where $N = \text{null}(A)$ is a basis for the nullspace of A, and z is a free parameter vector.

In this implementation, z is chosen to minimize the norm of the actuator commands, yielding the minimum-norm solution within the nullspace:

$$z = -N^\top u_p, \quad \text{with} \quad u_p = A^\dagger(-b + u_{\text{auxiliary}}).$$

Thus, the final actuator command vector is: $u = u_p + Nz$. The computed actuator commands $\mathbf{u}$ are then converted into rotor speeds $u_{\omega,i}$ and tilting angles α_i for each propeller via Eq. (7)

Results The actual pose (position and orientation) closely follows the desired trajectory, as shown in Figure 18. The apparent discrepancy in the attitude, particularly in the yaw angle ψ, is merely a representation artifact, since an error of 2π radians is equivalent to a zero-degree error. As is possible to see from Figure 20 and 21, the position and velocity errors remain extremely small, with peak values reaching 3×10^{-5} m and 6×10^{-6} m/s, respectively. Similarly, the orientation and angular velocity errors exhibit maximum values of 2×10^{-4} rad and 1×10^{-4} rad/s, respectively.

This behavior confirms that the feedback linearization controller is highly effective in ensuring accurate trajectory tracking and stabilization of the UAV, both in terms of position and orientation.

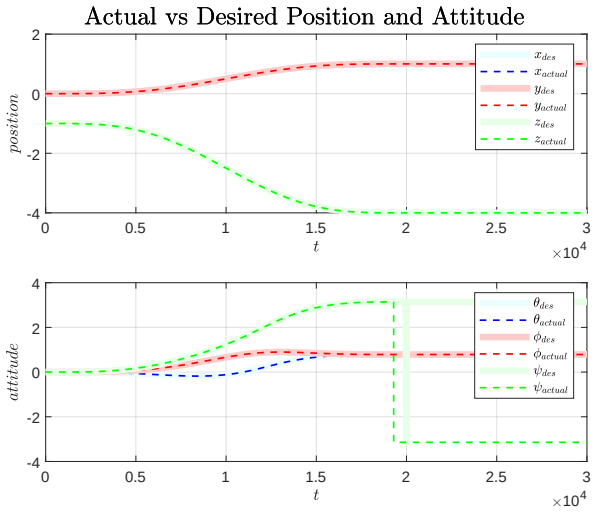

Figure 18: Desired and Actual Position and Attitude

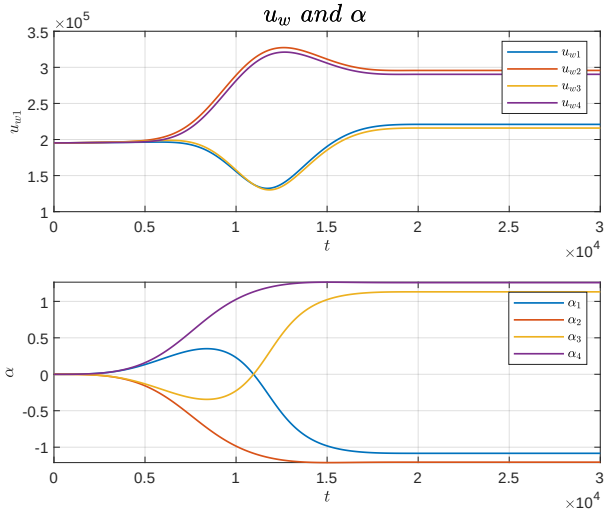

Figure 19: u_w and α

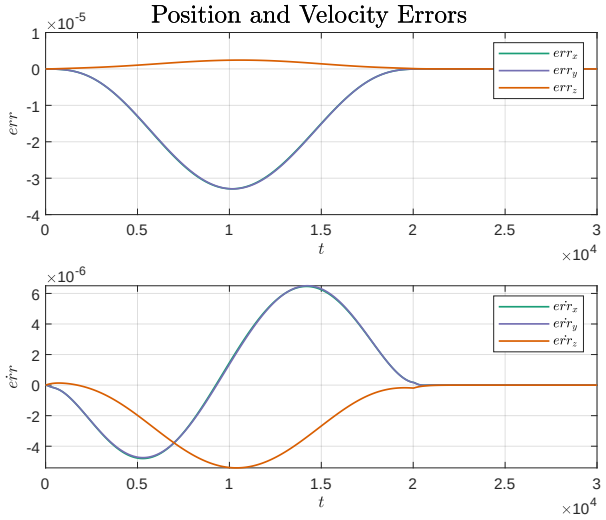

Figure 20: Position and Velocity Errors

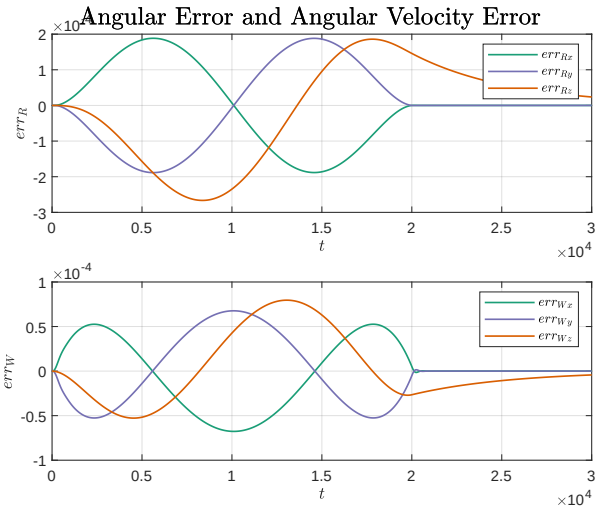

Figure 21: Angular Position and Angular Velocity Errors

However, it is important to note that this excellent tracking performance is achieved at the expense of extremely high control inputs. The required actuator efforts (u_w) can be far beyond realistic limits, highlighting a critical trade-off: while the controller can, in theory, drive the errors to zero, the physical feasibility of the resulting control actions is not guaranteed. In a real system, actuator saturation or failure would likely prevent such perfect tracking, and the errors would remain larger. In fact, despite the introduction of nullspace exploitation in the control allocation—intended to minimize the actuator commands by projecting the solution onto the nullspace of the allocation matrix A—the values of u_w remain extremely high, reaching the order of 2×10^5 or more for all four propellers. Since u_w corresponds to the square of the angular velocity, this implies that the required speed for each propeller is approximately $\approx 447\,\text{rad/s}$,

which translates to about $\approx 4268\,\text{RPM}$. Such high speeds demand a significant effort from the actuators, potentially pushing them close to or beyond their operational limits.

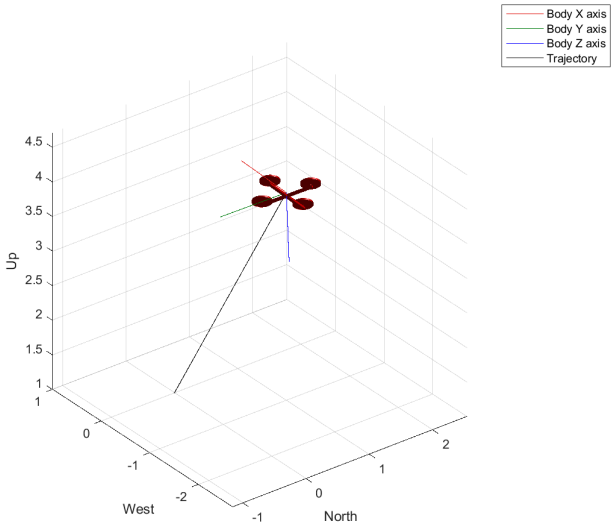

Figure 22: 3D drone final configuration

Trajectory The tilting quadrotor starts from an initial position of $p_0 = [0,\ 0,\ -1]$ in the NED reference frame and follows a smooth trajectory toward the final position $p_f = [1.0,\ 1.0,\ -4.0]$. During this motion, it also adjusts its orientation to reach a roll-pitch-yaw configuration of $[45°,\ 45°,\ 0°]$. In the final seconds of the trajectory, the quadrotor performs a stable hover in this target pose, maintaining both position and attitude with high precision. From the error plots, it can be observed that after reaching the final position, the tilting quadrotor enters a stable hover configuration. During this phase, the position and linear velocity errors converge to near-zero values. The remaining attitude corrections are primarily concentrated on the yaw angle ψ, which continues to be finely adjusted in the final seconds.