Project of *Control Problems in Robotics*

Modeling and controlling the Mars Helicopter *Ingenuity*

Davide De Santis
1886037
DIAG Department
University of Rome "La Sapienza"
desantis.1886037@studenti.uniroma1.it

Simone Orelli
1749732
DIAG Department
University of Rome "La Sapienza"
orelli.1749732@studenti.uniroma1.it

Antonio Rapuano
2044902
DIAG Department
University of Rome "La Sapienza"
rapuano.2044902@studenti.uniroma1.it

Mert Tekyurt
2057350
DIAG Department
University of Rome "La Sapienza"
tekyurt.2057350@studenti.uniroma1.it

April 2024

Abstract

This paper details the process of developing a control system for Ingenuity, a drone helicopter that operated from 2021 to 2024 on Mars as part of NASA's Mars 2020 mission. Said process consists first of creating a mathematical model capable of representing the behavior of the drone in flight over Mars, and then designing a controller capable of imparting desired behavior and performance (trajectory tracking) to the model. The problem is approached through two different control strategies: first, with a static input-to-output feedback linearization controller, and then, through the backstepping technique. Among the two, only the latter returns satisfactory results, as it is able to achieve the objectives by exploiting the fact that Ingenuity is an underactuated robot.

Contents

1 Introduction

In NASA's Mars 2020 mission, the helicopter *Ingenuity* was deployed to demonstrate the first powered flight on another planet. The helicopter was designed to be a technology demonstrator, and its main goal was to prove that powered flight in the thin Martian atmosphere is possible. By the date of the end of its mission (caused by the rupture of one of the rotor blades), Ingenuity had far exceeded every expectation: originally designed to perform up to five experimental test flights over 30 days, instead it flew for almost three years, performing 72 flights, and traversing more than 14 times the planned distance.

In the upcoming sections of the document we will present a mathematical model of Ingenuity, and we will discuss the control strategies that can be used to stabilize and control it, possibly around a planned trajectory, during flight in the Martian atmosphere. The theoretical results will be validated through simulated experiments.

1.1 Helicopter structure

At its core, Ingenuity features a fuselage designed for minimal weight and maximum strength, a sturdy framework for housing critical components such as the avionics, power system, and rotor assembly.

The most prominent feature is its coaxial rotor system, comprising two horizontally-overlapping counter-rotating (to mitigate reaction torque effects) blades, ensuring stability and precise maneuverability during flight. Attached to the rotor system is a central mast, which extends upward from the fuselage.

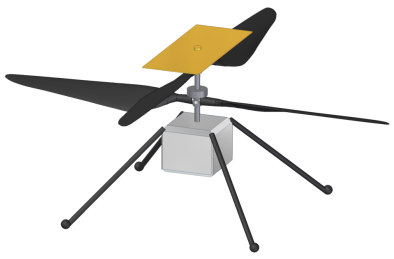

Figure 1.1: 3D render of the Ingenuity Mars Helicopter.

Complementing these is a suite of sensors, cameras, and communication equipment strategically integrated throughout Ingenuity's structure. These instruments enable it to navigate autonomously according to the instructions coming from the base of operation on Earth and capture imagery of the Martian surface.

The main parameters of the helicopter structure model are summarized in Table 1.1.

Parameter	**Symbol**	**Value**
Mass	m	$1.8\,\mathrm{kg}$
Rotor diameter	$2\,r$	$1.2\,\mathrm{m}$
Blade chord	c	$0.24\,\mathrm{m}$
COM - upper rotor distance	$d_{cm,u}$	$0.10\,\mathrm{m}$
COM - lower rotor distance	$d_{cm,l}$	$0.05\,\mathrm{m}$
Inertia	I	$diag(0.210, 0.288, 0.278)\,\mathrm{kg\,m^2}$

Table 1.1: Structural parameters of the Ingenuity model.

1.2 Martian environment

Even though the Martian gravity is only about 38% of the Earth's, its thin atmosphere, composed mainly of carbon dioxide with a ground-level pressure of about 0.6% of the Earth's, makes it difficult for aerial vehicles to fly. The athmospheric model is split in two zones: below and above the altitude of 7000 m. Since Ingenuity only performs low level flights, we are interested in the parameters of the lower layer, whose expressions (with respect to altitude) are summarized in Table 1.2.

Parameter	**Sym**	**Value**
Gravity	g	$3.71\,\mathrm{m\,s^{-2}}$
Temperature	T	$-242.1 - 9.98 \times 10^{-4}\,h\,\mathrm{K}$
Pressure	p	$0.699\,e^{-0.9\times10^{-5}h}\,\mathrm{kPa}$
Air density	ρ	$p/(0.192\,T)\,\mathrm{kg\,m^{-3}}$

Table 1.2: Martian environment parameters for altitude $h < 7000\,\mathrm{m}$.

2 Mathematical model

Each of the two rotors of Ingenuity generates a thrust vector in a commanded direction, which is applied on the rotating hub. Not only does this result in a force capable of moving the helicopter, but it also generates a torque that can be used to control its orientation.
The helicopter is treated as a simple rigid body without taking into account the aerodynamics of the blades, therefore the only external force is gravity.

From now on, we will consider an inertial reference frame (superscript ^{i}) and a non-inertial one attached to the drone body (superscript ^{b}).
The relative orientation of the latter with respect to the former is expressed in roll, pitch and yaw angles (i.e. the three components of W), or alternatively through the corresponding rotation matrix (in compact trigonometric notation with $\sin(W_i) = s_i$ and $\cos(W_i) = c_i$):

$$R(W) = R = \begin{bmatrix} c_2c_1 & s_2s_1c_3 - s_1c_3 & s_3s_2c_1 + s_1s_3 \\ c_2s_3 & s_2s_1s_3 + c_3c_1 & s_3s_2c_1 - s_1c_3 \\ -s_2 & s_1c_2 & c_2c_1 \end{bmatrix}$$

2.1 Helicopter dynamics

The magnitude of the thrust generated by each of the rotors is given by the sum of two components, one due to the lift and the other due to the drag.

$$F_l = K_l\omega^2,$$
$$F_d = K_d\omega^2,$$

where ω represents the angular velocity of the rotor, while K_l and K_r are two coefficients, whose expressions are the following:

$$K_l = \frac{1}{2}C_lcr^3\rho,$$
$$K_d = \frac{1}{2}C_dcr^3\rho.$$

In the previous formulas, c is the chord of a rotor blade and r is its hub-to-tip length; ρ is the air density, while C_l and C_d indicate the lift and drag coefficients of the rotor blades, which depend on their angle of attack. We consider the latter constant at a value such that $K_l - K_d > 0$.

This means that we can write the total thrust norm generated by each rotor as:

$$||F|| = F_l - F_d = (K_l - K_d)\,\omega^2, \tag{2.1}$$

therefore we can treat ω_u and ω_l as two of the control inputs of the system (respectively, for the upper and lower rotor).

Now consider the angles that the projections of the thrust vectors on the planes xz and yz form with the vertical axis z, called respectively α and β: this couple is the other input of the system, and it is the same for the upper and lower rotors. The two angles are illustrated in Figure 2.1.

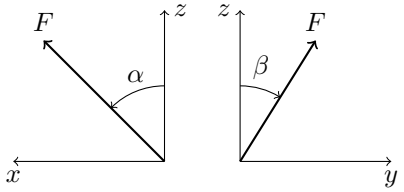

Figure 2.1: Angles α and β that the projections of the thrust vector forms with the vertical direction in the xz and yz planes of the body frame.

In the body reference frame, the thrust vector is then written by means of the auxiliary angle γ, i.e. the angle that the thrust vector forms with the vertical axis z in the three-dimensional space:

$$F^b = T_{(\alpha,\beta)}||F|| \tag{2.2}$$
$$\text{with} \quad T_{(\alpha,\beta)} = \begin{bmatrix} -\tan\alpha\cos\gamma \\ -\tan\beta\cos\gamma \\ \cos\gamma \end{bmatrix},$$
$$\tan^2\gamma = \tan^2\alpha + \tan^2\beta.$$

Since the thrust vectors are not applied in the center of mass of the helicopter, they also generate torques according to:

$$\tau^b = \begin{bmatrix} 0 \\ 0 \\ d_{cm} \end{bmatrix} \times F^b, \tag{2.3}$$

with d_{cm} being the distance between the center of mass of the helicopter and the center of the rotors. Note that this cross product results in a vector whose last component is null. In fact, the yaw rotation is

caused by the difference of the reaction torques generated by the two rotors:

$$\tau_r^b = \begin{bmatrix} 0 \\ 0 \\ Q_u - Q_l \end{bmatrix} \quad \text{with} \quad Q = 0.02\,||F||. \tag{2.4}$$

According to Newton's second law, the overall force acting on the helicopter is the sum of the forces generated by the two rotors and the force of gravity:

$$\begin{aligned} F_{tot}^b &= F_u^b + F_l^b + R^T F_g^i \\ &= T_{(\alpha,\beta)}(K_l - K_d)(\omega_u^2 + \omega_l^2) + R^T F_g^i, \end{aligned}$$

where F_{tot}^b is the total force acting on the body and F_g^i is the gravitational force. Similarly, the total torque is composed of the torques generated by the two rotor thrust vectors (roll and pitch) and the reaction torque of the two rotors (yaw):

$$\tau_{tot}^b = \tau_u^b + \tau_l^b + \tau_r^b, \tag{2.5}$$

where τ_{tot}^b is the total torque acting on the body.

By applying these laws, we compute the derivative of the velocity V in the body frame. Neglecting external forces, the cross product term accounts for the Coriolis effect. This is also done for the angular velocity Ω, where the cross product term represents the gyroscopic effect.

$$\begin{aligned} \frac{d}{dt}V^b &= \frac{F_{tot}^b - \Omega^b \times (mV^b)}{m}, \\ \frac{d}{dt}\Omega^b &= I^{-1}(\tau_{tot}^b - \Omega^b \times (I\Omega^b)). \end{aligned}$$

In the inertial frame, the position P and the orientation W of the robot are updated by integrating the velocity and the angular velocity, with:

$$\begin{aligned} \frac{d}{dt}P &= R\,V^b, \\ \frac{d}{dt}W &= R\,\Omega^b. \end{aligned}$$

2.2 Actuator dynamics

The control signals ω_u, ω_l, α and β are not instantly translated into the corresponding angular velocities or swashplate tilt angles: non-idealities arise since the physical components of Ingenuity are set in motion by actuators (typically electrical motors).

In reality, often actuators integrate low-level closed loops (sometimes even analog, e.g. through potentiometers) whose dynamics may be overlooked in the high-level control system design. To account for the effect of these components in our model and simulations, however, it is appropriate to model each of the four input channels as double integrators:

$$G_i(s) = \frac{1}{s^2}, \quad i = 1, 2, 3, 4.$$

In feedforward, the trivial solution is to control the double integrators with (physically realizable) controllers such as:

$$K_i(s) = \frac{s^2}{(1 + s\,T_i)^2}, \quad i = 1, 2, 3, 4,$$

with T_i being a (small enough) time constant that can be tuned to match the actual dynamics of the actuators.
This means that, at the end of the day, the resulting actuator open-loop transfer functions are equal to:

$$L_i(s) = G_i(s)\,K_i(s) = \frac{1}{(1 + s\,T_i)^2}, \quad i = 1, 2, 3, 4.$$

3 Control via I/O feedback linearization

From the control perspective, Ingenuity is an underactuated robot, since the number of control inputs is less than the number of degrees of freedom. Specifically, it has six degrees of freedom (three for the position and three for the orientation) and only four control inputs (the rotation speed of the two rotors ω_u and ω_l, and the thrust vector angles α and β).

In order for the robot to be controlled, it is necessary to exploit its natural dynamics by employing the four inputs to impart a certain behavior to the total system (both positional and rotational state variables) such that, at the end of the day, the three-dimensional position and the yaw angle follow the desired course in time. To grasp the concept, the reader can imagine a quadrotor or helicopter with a tail rotor, which need to tilt in the desired direction in order to move.

If we choose as controlled outputs of the system the three position components and the yaw angle,

it is possible to employ the input-to-output feedback linearization technique to design a controller that renders unobservable the nonlinear dynamics of the helicopter. As a result, the system seen between input and output will have linear dynamics and thus could be easily controlled via linear control techniques.

Consider the following nonlinear state-space representation of the robot dynamics:

$$\frac{d}{dt}\begin{bmatrix} P \\ V^b \\ W \\ \Omega^b \end{bmatrix} = \begin{bmatrix} R\,V^b \\ \frac{1}{m}R^T F_g^i - \Omega^b \times V^b + \frac{1}{m}u \\ R\,\Omega^b \\ -I^{-1}\Omega^b \times (I\Omega^b) + I^{-1}\tau_{tot}^b \end{bmatrix} \tag{3.1}$$

where the input u is the total thrust force generated by the rotors:

$$u = F_u^b + F_l^b. \tag{3.2}$$

Calculations are facilitated if we make a change of coordinates to the inertial frame.

$$\frac{d}{dt}\begin{bmatrix} P \\ \dot{P} \end{bmatrix} = \begin{bmatrix} \dot{P} \\ \frac{1}{m}F_g^i + \Omega^b \times \dot{P} - (R\Omega^b) \times \dot{P} + \frac{1}{m}Ru \end{bmatrix}.$$

The idea is to choose u to cancel the nonlinear part of the positional dynamics:

$$u = mR^T[(R\Omega^b) \times \dot{P} - \Omega^b \times \dot{P} - \frac{1}{m}F_g^i + v].$$

The transfer function relating the input v to the output P is then a double integrator:

$$\frac{d^2}{dt^2}P(t) = v(t) \iff \frac{P(s)}{v(s)} = \frac{1}{s^2}.$$

It can be controlled using a simple PD controller with a feedforward term:

$$v = k_{p,1}(P^d - P) + k_d\left(\frac{d}{dt}P^d - \dot{P}\right) + \frac{d^2}{dt^2}P^d.$$

Notice that the only input affecting the Ω_3^b dynamics is the reaction torque τ_r^b. Controlling the third component of the angular velocity in the body frame is all we can do if we want to keep the inputs u and τ_r^b decoupled, and it is sufficient if we assume that the roll and pitch angles are sufficiently small. Therefore, we proceed in an analogous way to the previous case, choosing:

$$\tau_r^b = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}[\Omega^b \times (I\Omega^b) + Iw]$$

This results in the following transfer function for the Ω_3^b dynamics:

$$\frac{d}{dt}\Omega_3^b(t) = w_3(t) \iff \frac{\Omega_3^b(s)}{w_3(s)} = \frac{1}{s},$$

where w_3 is the third component of w. The linearized system can be controlled using a simple proportional controller with a feedforward term:

$$w = \begin{bmatrix} 0 \\ 0 \\ k_{p,2}(\Omega_{3,d}^b - \Omega_3^b) + \frac{d}{dt}\Omega_{3,d}^b \end{bmatrix}.$$

Following this, a critical point is reached: once an feedback linearizing control is implemented, it is worth taking a look at how the part of the system that has been hidden behaves. We could call this part, in a non-formal way, the *forced zero dynamics* of the system. It is obtained by substituting 2.3 and 2.5 in 3.1:

$$\frac{d}{dt}\begin{bmatrix} \Omega_1^b \\ \Omega_2^b \end{bmatrix} = \begin{bmatrix} -\frac{F_{l,2}^b d_{cm,l} + F_{u,2}^b d_{cm,u}}{I_{1,1}} - \frac{I_{2,2} - I_{3,3}}{I_{1,1}}\Omega_2^b\Omega_3^b \\ \frac{F_{l,1}^b d_{cm,l} + F_{u,1}^b d_{cm,u}}{I_{2,2}} + \frac{I_{1,1} - I_{3,3}}{I_{2,2}}\Omega_1^b\Omega_3^b \end{bmatrix}.$$

The latter is an autonomous dynamics that is affected by the signals $F_{l,1}^b$, $F_{l,2}^b$, $F_{u,1}^b$ and $F_{u,2}^b$, which in turn depend on the feedback linearizing control. In practical terms, in order to enforce the designed behavior, the controller may induce pseudo-chaotic angular accelerations in the not-directly-controllable part of the system, leading to undesired and "clumsy" behaviors. It is a consequence of the fact that the system is underactuated.

3.1 Actuation

The objective of this section is to recover the actual control inputs of Ingenuity, i.e. the rotation speed of the upper and lower blades ω_u and ω_l and the tilt angles of the swashplate α and β, given the values of the linearizing signals u and τ_r^b entering the system in 3.1 with 2.5.

Since 3.2 holds, from the definition of the angles α and β illustrated in 2.1, under the assumption that $u_3 \neq 0$ [1], the following is straightforward:

$$\alpha = \arctan\left(\frac{u_1}{u_3}\right),$$
$$\beta = \arctan\left(\frac{u_2}{u_3}\right),$$

where u_1, u_2 and u_3 are the components of the vector u.

Knowing α and β, it is possible to combine the third component of 2.2 and 3.2 to find the sum of the norms of the forces F_u^b and F_l^b, given that $\cos\gamma \neq 0$[2]:

$$||F_u^b|| + ||F_l^b|| = \frac{u_3}{\cos\gamma},$$

while their difference is given by exploiting 2.4:

$$||F_u^b|| - ||F_l^b|| = 50\,\tau_{r,3}^b,$$

where $\tau_{r,3}^b$ is the third component of the vector τ_r^b. Using 2.1, we obtain the following system of equations:

$$\begin{cases} (K_l - K_d)\,(\omega_u^2 + \omega_l^2) = \frac{u_3}{\cos\gamma} \\ (K_l - K_d)\,(\omega_u^2 - \omega_l^2) = 50\,\tau_{r,3}^b \end{cases},$$

whose solutions are:

$$\omega_u = \sqrt{\frac{\frac{u_3}{\cos\gamma} + 50\,\tau_{r,3}^b}{2\,(K_l - K_d)}},$$
$$\omega_l = \sqrt{\frac{\frac{u3}{\cos\gamma} - 50\,\tau_{r,3}^b}{2\,(K_l - K_d)}}$$

and only exist if $\tau_{r,3}^b$ satisfies:

$$|\tau_{r,3}^b| \leq \frac{u_3}{50\,\cos\gamma}.$$

3.2 Simulation results

The implemented control law was tested on a trajectory tracking task. To complete the task, the helicopter is required to perform, cyclically with period $T = 100\,\text{s}$, an helical trajectory for 70 s and then to come back to the origin in the remaining 30 s, while keeping the yaw angle to zero, i.e.

$$P^d = \begin{cases} \begin{bmatrix} r \cdot cos(\frac{t}{d}) \\ r \cdot sin(\frac{t}{d}) \\ 1 + \frac{t}{k} \end{bmatrix} & \text{if } t \;\%\; 50 < 30 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} & \text{else} \end{cases} \tag{3.3}$$

$$W^d = 0 \tag{3.4}$$

with $r = 5\,\text{m}$ radius of the helix, $d = 5$ a scaling factor regulating the speed of the (x, y) plane and $k = 10$ a scaling factor regulating the rate of climb. The symbol % stands for remainder after division. Moreover, the initial position of the robot belongs to the helical trajectory, while the initial attitude is null.

For the sake of realism, the control inputs to the system cannot range over an infinite span, but it is appropriate to constrain them to vary over limited intervals. More specifically, we impose the following constraints:

$$\omega_u \in [0, 2800]\,rpm \tag{3.5}$$
$$\omega_l \in [-2800, 0]\,rpm \tag{3.6}$$
$$\alpha \in [-10, 10]^\circ \tag{3.7}$$
$$\beta \in [-10, 10]^\circ \tag{3.8}$$

Figure 3.1 shows the comparison between the reference and the actual trajectory traversed by the robot. The tracking performance is outstanding as long as the error is small, but failure occurs in the instant that the error is too large, because of the explosion of the states governing the forced zero dynamics. We deduce that the system is extremely unrobust, so the control action is not sufficiently reliable.

[1] Always satisfied during normal conditions, since there will always be a part of u_3 counteracting gravity.
[2] Consequence of $u_3 \neq 0$[1] using 3.2 and 2.2.

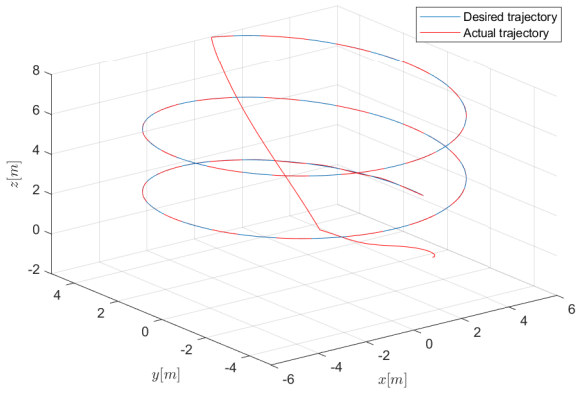

Figure 3.1: Trajectory tracking.

This phenomenon is also visible in Figure 3.2. Despite the pitch having, up to a certain point, an acceptable evolution, the roll exhibits noticeable fluctuations that induce Ingenuity to assume an unrealistic behavior. As an additional consequence, the small angles assumption of the previous signals is violated, leading to a significant error in the yaw angle, whose reference is not followed. Moreover, as in the previous graph, the abrupt change in the reference causes the bursting of the examined states.

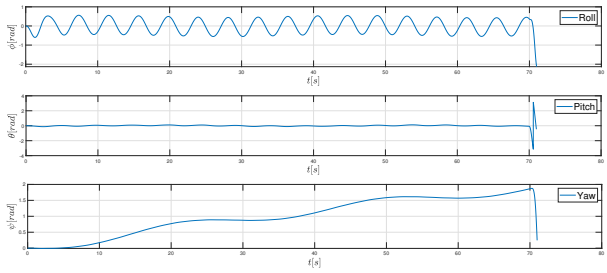

Figure 3.2: Evolution of the roll, pitch and yaw angles respectively.

The previous results can only be obtained if the inputs are not constrained. In fact, Figures 3.3 and 3.4 show that the value of the angle β breaks the constraint 3.8. If saturations were imposed on the input signals, the system would fail to track the trajectory and diverge even in the initial phase.

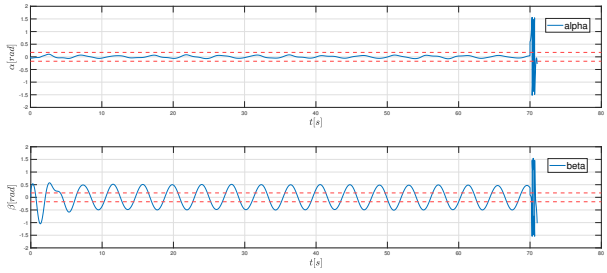

Figure 3.3: Angles α and β of the swashplate.

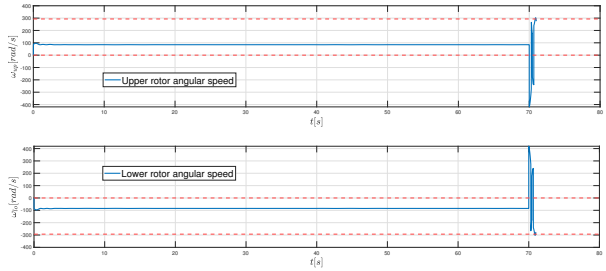

Figure 3.4: Angular velocities of the upper and lower rotor respectively.

4 Control via backstepping

In this section we will present the backstepping control technique applied to the helicopter model. The main idea is to act on the whole system to ultimately obtain global asymptotic stability and therefore track a desired trajectory. Let us consider once again the state-space representation in 3.1. In order to find an analytical solution to the control problem, we will assume that the horizontal components of the control input u (defined in 3.2) are negligible. This is justified by the fact that, in nominal operational conditions, the main contribution of the thrust generated by the rotors is the vertical one with respect to the body frame. Therefore u can be written as:

$$u = \begin{bmatrix} 0 \\ 0 \\ u_3 \end{bmatrix}. \tag{4.1}$$

The objective is to design a control law (u, τ^b_{tot}), depending only on the measurable states, such that

the tracking error

$$\varepsilon(t) \coloneqq (P(t) - P^d(t), W_3(t) - W_3^d(t))$$

is asymptotically stable for all initial conditions.

We start the procedure with the definition of the position error in the body reference frame:

$$\delta_1 \coloneqq R^T(P^d - P).$$

Its time derivative is given by:

$$\begin{aligned}\frac{d}{dt}\delta_1 &= -\Omega^b \times R^T(P^d - P) + R^T\left(\frac{d}{dt}P^d - \frac{d}{dt}P\right)\\ &= -\Omega^b \times \delta_1 + R^T\left(\frac{d}{dt}P^d - V^b\right).\end{aligned}$$

From this, we identify the velocity error in the body reference frame:

$$\delta_2 \coloneqq R^T\left(\frac{d}{dt}P^d - V^b\right).$$

After differentiation with respect to time, we obtain:

$$\frac{d}{dt}\delta_2 = -\Omega^b \times \delta_2 + R^T\frac{d^2}{dt^2}P^d - \frac{1}{m}(R^T F_g^i - u).$$

Define as V_1 the first Lyapunov function, regarding the pseudo-energy of the position and velocity errors, as:

$$V_1 \coloneqq \frac{1}{2}(\delta_1 + \lambda\delta_2)^T(\delta_1 + \lambda\delta_2) + \frac{1}{2}\delta_1^T\delta_1,$$

where $\lambda > 0$ is a slack variable to be tuned. The time derivative of V_1 is:

$$\frac{d}{dt}V_1 = (\delta_1 + \lambda\delta_2)^T\left(\delta_2 + \lambda R^T\frac{d^2}{dt^2}P^d - \frac{\lambda}{m}u\right) + \delta_1^T\delta_2.$$

We are looking for a virtual control input $(\frac{\lambda}{m}u)^*$ making $\frac{d}{dt}V_1 \leq -V_1$ barring an error that has to be recovered in the next iteration of the backstepping procedure. The aforementioned virtual control is:

$$\left(\frac{\lambda}{m}u\right)^* = \delta_1 + \lambda\delta_2 + 2\delta_2 + \lambda R^T\frac{d^2}{dt^2}P^d.$$

This choice results in:

$$\begin{aligned}\frac{d}{dt}V_1 = -(\delta_1 + \lambda\delta_2)^T(\delta_1 + \lambda\delta_2) - \lambda\delta_2^T\delta_2 +\\ (\delta_1 + \lambda\delta_2)^T\delta_3,\end{aligned}$$

where:

$$\delta_3 \coloneqq \left(\frac{\lambda}{m}u\right)^* - \frac{\lambda}{m}u.$$

We proceed further by computing the time derivative of δ_3 as:

$$\begin{aligned}\frac{d}{dt}\delta_3 = -\Omega^b \times \delta_3 - \frac{\lambda}{m}\Omega^b \times u + \frac{\lambda+2}{\lambda}\delta_3 - \frac{\lambda+2}{\lambda}\delta_1\\ -\frac{(\lambda+2)^2 - \lambda}{\lambda}\delta_2 + \lambda R^T\frac{d^3}{dt^3}P^d - \frac{\lambda}{m}\frac{d}{dt}u.\end{aligned}$$

Let us consider at this stage the error in the yaw component:

$$\varepsilon_2 = W_3^d - W_3.$$

The second Lyapunov function is:

$$V_2 = \frac{1}{2}\delta_3^T\delta_3 + \frac{1}{2}\varepsilon_2^2,$$

with its time derivative being:

$$\begin{aligned}\frac{d}{dt}V_2 = \delta_3^T\Bigg(-\Omega^b \times \delta_3 - \frac{\lambda}{m}\Omega^b \times u - \frac{\lambda+2}{\lambda}\delta_1 +\\ \frac{\lambda+2}{\lambda}\delta_3 - \frac{(\lambda+2)^2 - \lambda}{\lambda}\delta_2 + \lambda R^T\frac{d^3}{dt^3}P^d - \frac{\lambda}{m}\frac{d}{dt}u\Bigg)\\ +\varepsilon_2\left(\frac{d}{dt}W_3^d - \frac{d}{dt}W_3\right).\end{aligned}$$

The third component[3] of the first bracket in the previous equation can be directly shaped by imposing the value of the derivative of the lift u:

$$\begin{aligned}\frac{d}{dt}u_3 = \frac{m}{\lambda}\begin{bmatrix}0 & 0 & 1\end{bmatrix}\Bigg(\frac{\lambda+2}{\lambda}\delta_3 - \frac{\lambda+2}{\lambda}\delta_1\\ -\frac{\lambda^2+3\lambda+4}{\lambda}\delta_2 + \lambda R^T\frac{d^3}{dt^3}P^d + \delta_1 + \lambda\delta_2\Bigg).\end{aligned}$$

The vector $\begin{bmatrix}0 & 0 & 1\end{bmatrix}$ has been used to isolate the third component of the equation. The residual positional error is taken into account with the goal of imposing the decrease rate of $\frac{d}{dt}(V_1 + V_2)$ along

[3]Notice that $\delta_3^T(-\Omega^b \times \delta_3) = 0$ and $\Omega^b \times u = \begin{bmatrix}\star & \star & 0\end{bmatrix}^T$.

the system trajectories with the choice of the virtual control:

$$\left(\frac{\lambda}{m}\Omega^b \times u\right)^* \coloneqq \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}\left(\frac{\lambda+2}{\lambda}\delta_3 - \frac{\lambda+2}{\lambda}\delta_1 - \frac{\lambda^2+3\lambda+4}{\lambda}\delta_2 + \lambda R^T \frac{d^3}{dt^3}P^d + \delta_1 + \lambda\delta_2 + k_1\delta_3\right).$$

that lives in the orthogonal plane to u_3, where $k_1 > 0$ is slack variable to be tuned in order to determine the rate of convergence of the error δ_3.
Through the last two choices, we have only controlled the part of $\frac{d}{dt}V_2$ regarding the position error. Now we can focus on the yaw term by introducing the virtual control:

$$\left(\frac{d}{dt}W_3\right)^* \coloneqq \frac{d}{dt}W_3^d + k_2\varepsilon_2,$$

The derivative of the second Lyapunov function becomes:

$$\frac{d}{dt}V_2 = \delta_3^T\delta_4 - k_1\delta_3^T\delta_3 - \delta_3^T(\delta_1 + \lambda\delta_2) - k_2\varepsilon_2^2 + \varepsilon_2\varepsilon_3,$$

where:

$$\delta_4 \coloneqq \left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \frac{\lambda}{m}\Omega^b \times u,$$
$$\varepsilon_3 \coloneqq \frac{d}{dt}W_3^* - \frac{d}{dt}W_3.$$

Finally, the last iteration starts with the computation of the time derivative of the latter:

$$\frac{d}{dt}\delta_4 = \frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \frac{\lambda}{m}\left(\frac{d}{dt}\Omega^b\right) \times u - \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right),$$
$$\frac{d}{dt}\varepsilon_3 = \frac{d^2}{dt^2}W_3^d + k_2\left(\frac{d}{dt}W_3^d - \frac{d}{dt}W_3\right) - \frac{d^2}{dt^2}W_3.$$

At this point, we can define the third Lyapunov function and proceed as usual.

$$V_3 = \frac{1}{2}\delta_4^T\delta_4 + \frac{1}{2}\varepsilon_3^2.$$

Which, after differentiation, results in:

$$\frac{d}{dt}V_3 = \delta_4^T\left(\frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \frac{\lambda}{m}\left(\frac{d}{dt}\Omega^b\right) \times u - \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right)\right) + \varepsilon_3\left(\frac{d^2}{dt^2}W_3^d + k_2\left(\frac{d}{dt}W_3^d - \frac{d}{dt}W_3\right) - \frac{d^2}{dt^2}W_3\right).$$

Once again, our goal is to "stabilize" the function $\frac{d}{dt}V_3$. This can be achieved through the following choices:

$$\frac{d^2}{dt^2}W_3 = \frac{d^2}{dt^2}W_3^d + k_2\left(\frac{d}{dt}W_3^d - \frac{d}{dt}W_3\right) + k_4\varepsilon_3 + \varepsilon_2,$$

$$\frac{\lambda}{m}\left(\frac{d}{dt}\Omega^b\right) \times u = \frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right) + \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}\delta_3 + k_3\delta_4.$$

The very last equation results into the following expressions for the derivatives of the angular velocities in the body frame:

$$\frac{d}{dt}\Omega_1^b = -\frac{m}{\lambda u_3}\begin{bmatrix}0 & 1 & 0\end{bmatrix}\left[\frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right) + \delta_3 + k_3\delta_4\right],$$
$$\frac{d}{dt}\Omega_2^b = \frac{m}{\lambda u_3}\begin{bmatrix}1 & 0 & 0\end{bmatrix}\left[\frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right) + \delta_3 + k_3\delta_4\right].$$

According to the above choices, the derivative of the third Lyapunov function becomes:

$$\frac{d}{dt}V_3 = -k_4\varepsilon_3^2 - \varepsilon_2\varepsilon_3 - \delta_4^T\delta_3 - k_3\delta_4^T\delta_4.$$

To summarize, consider the sum of the three aforementioned Lyapunov functions as the overall Lyapunov function V:

$$V = V_1 + V_2 + V_3 = \frac{1}{2}(\delta_1 + \lambda\delta_2)^T(\delta_1 + \lambda\delta_2) + \frac{1}{2}\delta_1^T\delta_1 + \frac{1}{2}\delta_3^T\delta_3 + \frac{1}{2}\epsilon_2^2 + \frac{1}{2}\delta_4^T\delta_4 + \frac{1}{2}\epsilon_3^2,$$

with its time derivative being:

$$\frac{d}{dt}V = -(\delta_1 + \lambda\delta_2)^T(\delta_1 + \lambda\delta_2) - \lambda\delta_2^T\delta_2 - k_1\delta_3^T\delta_3 \\ -k_2\varepsilon_2^2 - k_4\varepsilon_3^2 - k_3\delta_4^T\delta_4,$$

meaning that V is monotonically decreasing (and thus the control objective is achieved) if and only if we pick the following:

$$\frac{d}{dt}u_3 = \frac{m}{\lambda}\begin{bmatrix}0 & 0 & 1\end{bmatrix}\left(\frac{\lambda+2}{\lambda}\delta_3 - \frac{\lambda+2}{\lambda}\delta_1 \right. \\ \left. -\frac{\lambda^2+3\lambda+4}{\lambda}\delta_2 + \lambda R^T\frac{d^3}{dt^3}P^d + \delta_1 + \lambda\delta_2\right),$$

$$\frac{d}{dt}\Omega_1^b = -\frac{m}{\lambda u_3}\begin{bmatrix}0 & 1 & 0\end{bmatrix}\left[\frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \right. \\ \left. \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right) + \delta_3 + k_3\delta_4\right],$$

$$\frac{d}{dt}\Omega_2^b = \frac{m}{\lambda u_3}\begin{bmatrix}1 & 0 & 0\end{bmatrix}\left[\frac{d}{dt}\left(\frac{\lambda}{m}\Omega^b \times u\right)^* - \right. \\ \left. \frac{\lambda}{m}\Omega^b \times \left(\frac{d}{dt}u\right) + \delta_3 + k_3\delta_4\right],$$

$$\frac{d}{dt}\Omega_3^b = \frac{c_2}{c_1}\left(\frac{d^2}{dt^2}W_3^d + k_2\left(\frac{d}{dt}W_3^d - \frac{d}{dt}W_3\right) + k_4\varepsilon_3 \right. \\ \left. +\varepsilon_2 + \begin{bmatrix}0 & 0 & 1\end{bmatrix}M^{-1}\frac{d}{dt}M\,M^{-1}\Omega^b - \frac{s_1}{c_2}\frac{d}{dt}\Omega_2^b\right).$$

with the latter resulting from the differentiation of the yaw angle W_3, and M being:

$$M = \begin{bmatrix}-s_2 & 0 & 1\\ c_2s_1 & c_1 & 0\\ c_2c_1 & -s_1 & 0\end{bmatrix}.$$

4.1 Actuation

This section is the analogue of Section 3.1 for the backstepping controller.

The backstepping procedure yields the values of the time derivative of the lift $\frac{d}{dt}u$ and the desired angular velocity dynamics $\frac{d}{dt}\Omega^b$, which must be converted into the actual inputs of the robot ω_u, ω_l, α, and β.

For simplicity, let us use the following notation:

$$\tau := \tau_{tot}^b.$$

Assuming that α and β are small:

$$T_{(\alpha,\beta)} \approx \begin{bmatrix}\alpha\\ \beta\\ 1\end{bmatrix},$$

by 2.1, 2.2, 2.3, 2.4 and 2.5, we obtain a system of four equations in four unknowns:

$$\begin{cases}\begin{bmatrix}\tau_1\\ \tau_2\\ \tau_3\end{bmatrix} = \begin{bmatrix}-\beta\,(K_l - K_d)\,(d_{cm,u}\,\omega_u^2 + d_{cm,l}\,\omega_l^2)\\ \alpha\,(K_l - K_d)\,(d_{cm,u}\,\omega_u^2 + d_{cm,l}\,\omega_l^2)\\ 0.02\,(K_l - K_d)\,(\omega_u^2 - \omega_l^2)\end{bmatrix},\\ u_3 = (K_l - K_d)(\omega_u^2 + \omega_l^2)\end{cases}$$

whose (real) solutions exist only if the following condition is satisfied:

$$|\tau_3| \le \frac{u_3}{50}.$$

In that case, the solutions are:

$$w_u = \sqrt{\frac{u_3 + 50\,\tau_3}{2(K_l - K_d)}}$$

$$w_l = \sqrt{\frac{u_3 - 50\,\tau_3}{2(K_l - K_d)}}$$

$$\alpha = \frac{2\tau_2}{(d_{cm,l} + d_{cm,u})\,u_3 + 50\,(d_{cm,u} - d_{cm,l})\,\tau_3}$$

$$\beta = \frac{-2\tau_1}{(d_{cm,l} + d_{cm,u})\,u_3 + 50\,(d_{cm,u} - d_{cm,l})\,\tau_3},$$

where τ_1, τ_2 and τ_3 are the three components of τ, whose value comes from 3.1:

$$\tau = I\frac{d}{dt}\Omega^b + \Omega^b \times (I\Omega^b).$$

4.2 Simulation results

The backstepping controller was also tested on the a trajectory tracking task akin to the one seen in Section 3.2. The reference trajectory is expressed in 3.3 and 3.4, with the same parameters $r = 5\,\text{m}$, $d = 5$ and $k = 10$.

With the current controller, which is far more robust than of the previous one, we can also afford to start from an initial condition corresponding to a nonzero initial tracking error, i.e., the origin.

The comparison between the reference and the actual trajectory of Ingenuity is shown in Figure 4.1.

Notice that the robot is able to follow the desired path with a good approximation, while recovering both the initial position error and the final one (generated by the abrupt change in the reference).

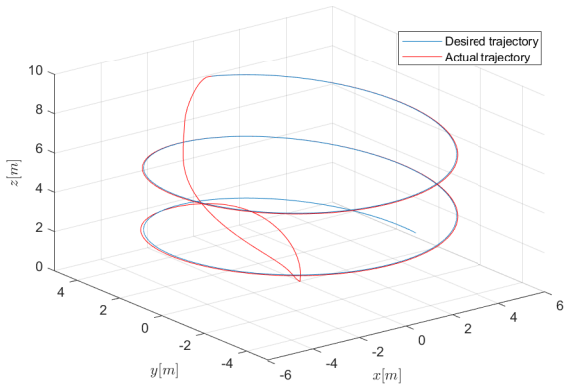

Figure 4.1: Trajectory tracking.

As theoretically expected, the controller exploits the roll and pitch dynamics to impart the desired behavior to the remainder of the system (see Figure 4.2).

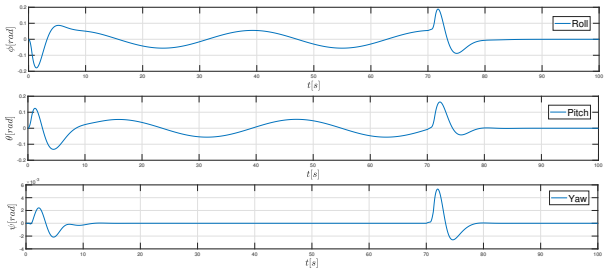

Figure 4.2: Evolution of the roll, pitch and yaw angles respectively.

Finally, the necessary control inputs, adequately saturated according to the constraints 3.5 - 3.8 so as to reflect the actual physical limitations of the actuators, are depicted in Figures 4.3 and 4.4.
We can therefore state that the controller is not only robust to external disturbances, but also to these types of constraints.

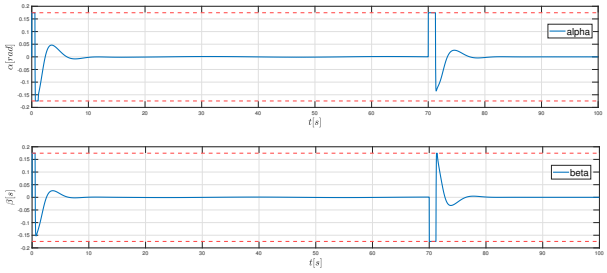

Figure 4.3: Angles α and β of the swashplate.

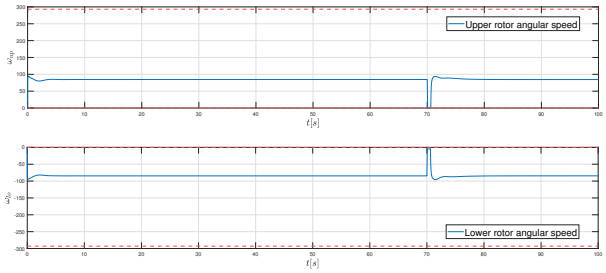

Figure 4.4: Angular velocities of the upper and lower rotor respectively.

5 Conclusions

The Mars helicopter *Ingenuity* is a complex system that requires careful modeling and control design to ensure its stability and performance. In this paper, we have presented a detailed mathematical model of the helicopter dynamics during flight through the Martian atmosphere, and then used it to design an input-output feedback linearization controller and a backstepping controller to stabilize it around a desired trajectory. As a result, the following conclusions can be drawn:

- Ingenuity is an underactuated system with four control inputs and six degrees of freedom, whose dynamics are highly nonlinear and depend on the altitude, the rotor speed, and the blade pitch angles. The model we have developed captures these dependencies and provides a good approximation of the real system behavior.

- The input-output feedback linearization controller is able to stabilize the positional and yaw dynamics of Ingenuity around a desired trajectory, but the hidden nonlinearities respond to the control inputs in an undesired and unrealistic way. This is due to the fact that the controller does not take into account the coupling between the different states of the system.
- The backstepping controller, on the other hand, is able to stabilize the full dynamics of Ingenuity because it accounts for the cascade-like inner structure of the system and exploits it by recursively designing virtual control laws and stabilizing error dynamics at deeper levels within the system, gradually steering it towards the goal.

References

[1] M. Vendittelli. *Ingenuity dynamic model*. Slides. *Control problems in robotics - Modeling and control of multi-rotor UAVs*, "La Sapienza" University, Rome. 2023.

[2] F. Moretti. *Modello dinamico di Ingenuity*. Research project. *SSAS*, "La Sapienza" University, Rome. 2022.

[3] Minh Duc Hua. "Contributions to the automatic control of aerial vehicles". Theses. Université Nice Sophia Antipolis, Dec. 2009. URL: https://theses.hal.science/tel-00460801.

[4] A. Dzul, T. Hamel, and R. Lozano. "Modeling and nonlinear control for a coaxial helicopter". In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 6. 2002, 6 pp. vol.6-. DOI: 10.1109/ICSMC.2002.1175550.

[5] Shannah Withrow-Maser et al. "Mars Science Helicopter: Conceptual Design of the Next Generation of Mars Rotorcraft". In: 2020. URL: https://api.semanticscholar.org/CorpusID:232168057.

[6] Håvard Fjær Grip et al. "Flight Control System for NASA's Mars Helicopter". In: *AIAA Scitech 2019 Forum* (2019). URL: https://api.semanticscholar.org/CorpusID:86511266.