actuators

MDPI

Article

Incremental Nonlinear Dynamics Inversion and Incremental Backstepping: Experimental Attitude Control of a Tail-Sitter UAV

Alexandre Athayde *, Alexandra Moutinho and José Raul Azinheira

IDMEC—Instituto de Engenharia Mecânica, Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisboa, Portugal; alexandra.moutinho@tecnico.ulisboa.pt (A.M.); jose.raul.azinheira@tecnico.ulisboa.pt (J.R.A.)

* Correspondence: alexandre.athayde@tecnico.ulisboa.pt

Abstract: Incremental control strategies such as Incremental Nonlinear Dynamics Inversion (INDI) and Incremental Backstepping (IBKS) provide undeniable advantages for controlling Uncrewed Aerial Vehicles (UAVs) due to their reduced model dependency and accurate tracking capacities, which is of particular relevance for tail-sitters as these perform complex, hard to model manoeuvres when transitioning to and from aerodynamic flight. In this research article, a quaternion-based form of IBKS is originally deduced and applied to the stabilization of a tail-sitter in vertical flight, which is then implemented in a flight controller and validated in a Hardware-in-the-Loop simulation, which is also made for the INDI controller. Experimental validation with indoor flight tests of both INDI and IBKS controllers follows, evaluating their performance in stabilizing the tail-sitter prototype in vertical flight. Lastly, the tracking results obtained from the experimental trials are analysed, allowing an objective comparison to be drawn between these controllers, evaluating their respective advantages and limitations. From the successfully conducted flight tests, it was found that both incremental solutions are suited to control a tail-sitter in vertical flight, providing accurate tracking capabilities with smooth actuation, and only requiring the actuation model. Furthermore, it was found that the IBKS is significantly more computationally demanding than the INDI, although having a global proof of stability that is of interest in aircraft control.

Keywords: incremental backstepping (IBKS); incremental nonlinear dynamics inversion (INDI); attitude control; uncrewed aerial vehicles (UAV); nonlinear control; tail-sitter; vertical take-off and landing (VTOL)

Citation: Athayde, A.; Moutinho, A.; Azinheira, J.R. Incremental Nonlinear Dynamics Inversion and Incremental Backstepping: Experimental Attitude Control of a Tail-Sitter UAV. *Actuators* **2024**, *13*, 225. https://doi.org/10.3390/act13060225

Academic Editors: Xuerui Wang and Erik-Jan Van Kampen

Received: 30 April 2024
Revised: 11 June 2024
Accepted: 14 June 2024
Published: 17 June 2024

1. Introduction

Hybrid Uncrewed Aerial Vehicles (UAVs) are a leading-edge research topic, aiming to combine the advantages of fixed-wing aircraft and multirotors—such as long endurance and high manoeuvrability—whilst avoiding their respective limitations [1]. These hybrid UAVs may be classified into multiple sub-categories, depending on the type and number of actuators, if these are fixed or rotatable,and how the transition to and from aerodynamic flight is performed, among others [2,3]. In general, tail-sitters are the simplest hybrid aircraft in terms of actuation, owing their designation to the fact that they take off and land on their tails [4]. Tail-sitters may have several different configurations, but one of the most popular is the bi-rotor design [3], having two proprotors—which act simultaneously as propellers and rotors—and two elevons—which also have the dual role of elevators and ailerons. They are suitable for high-endurance scenarios, as they can maintain control when gliding with minimal energy expenditure, thus allowing the coverage of large distances. Nevertheless, the transitions that this sort of UAV are required to perform are significantly complex, with the tail-sitter having to rely on the airspeed generated by its propellers to grant enough authority to the control surfaces to pitch the aircraft during these transition moments. As such, the automatic control of bi-rotor tail-sitters presents remarkable and

demanding challenges, as it requires that the aircraft should be stabilized not only in vertical or aerodynamic flight but also during the transition manoeuvrers between these flight stages.

Regarding UAV control, linear strategies are still common for both fixed-wing aircraft [5] and multi-rotors [6], mainly due to their simplicity of implementation and analytical proof of stability for the respective linearized models. In the case of aircraft that are capable of large altitudes or velocities, it is common to pair these linear controllers with a scheduling strategy that changes the values of relevant parameters to account for these varying conditions [7]. Notwithstanding, this process is computationally demanding, and the assumptions and approximations that result from the linearization process may negatively impact the performance of the controller [4]. To counter these limitations, nonlinear control strategies arise as strong alternatives, often including or cancelling the nonlinearities of the model of the system for the computation of the control action [3]. In aircraft control, Nonlinear Dynamics Inversion (NDI) and Backstepping (BKS) are undeniably two of the more acknowledged strategies [8], with examples of application in multi-rotors [9,10] and fixed-wings [11,12] and, more recently, hybrid UAVs [13,14]. Consisting of an inversion of the model of the system to compute the stabilizing control action, the NDI strategy results in a straightforward yet effective control law, adequate for the stabilization of the attitude of aircraft. With a similar reasoning but having a stronger theoretical foundation, the Backstepping method is also common in UAV control, with the drawbacks of requiring for the model of a given system to be expressed in lower-triangular form whilst also having more intermediate computations when compared with NDI. Nevertheless, a successful application of BKS proves the global stability of the controlled system, which is a tempting aspect to consider when dealing with aircraft due to the safety and security involved.

Although both NDI and BKS are suitable and common control strategies to stabilize the attitude of UAVs, these are still methods that rely greatly on the model of the system, which can be subject to significant uncertainty or may require expensive and demanding processes to be accurately identified [15]. Circumventing this limitation, incremental versions of these controllers—INDI and IBKS—have been gaining popularity as they require limited information of the model of the UAV to stabilize it, with a number of recent cases of applications in aircraft control [16–20]. The limited model dependency makes them ideal candidates for the challenges involved in controlling tail-sitters due to the difficulty in modelling the complex aerodynamics during transitions and the propeller/control surface interaction. Additionally, the transitions from vertical to aerodynamic flight and vice-versa that characterize tail-sitters suggest the employment of quaternion-based controllers in order to avoid singularity-related issues of traditional attitude representations [21], which adds additional complexity in implementing INDI and IBKS controllers to tail-sitters. In this context, a quaternion-based form of IBKS is absent from the current literature, and so are implementations of this control strategy to tail-sitters with experimental validation, to the best of the authors' knowledge, but some of the recent works have addressed the application of INDI to these UAVs [19,20]. Lastly, a systematic comparison between these two controllers, evaluating their respective advantages and limitations when implemented in the same UAV and exposed to the same conditions, is also absent from the literature, which is another research gap addressed in this article.

Accounting for this introduction, this paper addresses the aforementioned research gaps by exploring the application of Incremental Nonlinear Dynamics Inversion and Incremental Backstepping in the vertical flight stabilization of the X-Vert VTOL [22], a small and lightweight radio-controlled bi-rotor tail-sitter. Within this research article, three separate scientific contributions are offered: firstly, the deduction of a quaternion-based Incremental Backstepping is made, representing the theoretical contribution of this article; as a second contribution, this IBKS controller, together with the INDI solution that resulted from a previous research work [23], is implemented in the tail-sitter prototype and experimentally validated; and the third and final contribution is a systematic analysis of the INDI and IBKS when applied to the X-Vert VTOL tail-sitter, focusing on the aspects that differ, such

as global stability proof and computational requirements, as well as the tracking results obtained from the experimental flight trials. Regarding the structure of the article, the UAV model and simulator are briefly described in Section 2, and the application of these two incremental control strategies for UAV attitude control is introduced in Section 3, followed by validation in Hardware-in-the-Loop (HITL) simulations, described in Section 4. Following this validation, in Section 5 the INDI and IBKS controllers are implemented and evaluated in an experimental scenario, resorting to a Motion Capture System (MCS) to record the movement of the UAV during the flight trials, allowing a systematic comparison to be drawn between both.

2. Aircraft Model and Simulator

The simulator used in this work is based on a previous work [23], consisting of a simulated environment developed in MATLAB/Simulink 2023a. The aircraft model is largely based on the research work of [24], which modelled the X-Vert VTOL UAV as a rigid body under constant atmospheric properties, and complemented by the relevant sensor models [21]. The generic layout of the simulator is provided in Figure 1, which is briefly described afterwards.

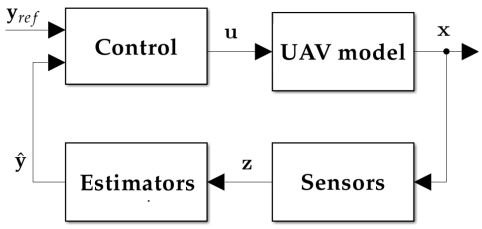

Figure 1. Layout of the simulated environment [23].

2.1. UAV Nonlinear Model

The Equations of Motion (EOM) used in the aircraft model in [21,25] are

$$\dot{\mathbf{v}}_g^B = \frac{1}{m_{uav}}\mathbf{f}^B - (\mathbf{w}_g^B \times \mathbf{v}_g^B) \tag{1}$$

$$\dot{\mathbf{w}}_g^B = \mathbf{J}_{uav}^{-1}(\mathbf{m}^B - \mathbf{w}_g^B \times \mathbf{J}_{uav}\mathbf{w}_g^B) \tag{2}$$

$$\dot{\mathbf{p}}^{NED} = \mathbf{R}_B^{NED}\,\mathbf{v}_g^B \tag{3}$$

$$\dot{\mathbf{q}}^{NED} = \frac{1}{2}\left(\mathbf{S}_{\mathbf{q}}^{\mathbf{w}}\,\mathbf{q}^{NED}\right) \tag{4}$$

which encompass the expressions for the time derivatives of the linear and angular velocities relative to the ground, expressed in the body frame of the UAV, $\mathbf{v}_g^B = [u_g, v_g, w_g]^T$ in m/s and $\mathbf{w}_g^B[p_g, q_g, r_g]^T$in rad/s, respectively, its position expressed in metres in the North-East-Down frame, $\mathbf{p}^{NED} = [N, E, D]^T$, and the quaternion-expressed orientation in relation to this same frame, $\mathbf{q}^{NED} = [q_0, q_1, q_2, q_3]^T$, which is taken as dimensionless. A more-appropriate notation for $\mathbf{q}^{NED}$ would be $\mathbf{q}_B^{NED}$ as it represents a rotation from the body to the fixed frame, but the subscript is ommited sa as not to overburden the notation. $\mathbf{R}_B^{NED}$ represents this same rotation, allowing one to express the kinematics relationship between $\mathbf{v}_g^B$ and $\mathbf{p}^{NED}$, whilst $\mathbf{S}_{\mathbf{q}}^{\mathbf{w}}$ is a skew-symmetric matrix for the rotational quaternion kinematics [21]. Lastly, the mass of the aircraft is denoted by m_{uav}, its inertia matrix by $\mathbf{J}_{uav}$, and the resultant forces and moments by $\mathbf{f}^B$ and $\mathbf{m}^B$, respectively, both expressed in the body frame.

The thirteen states in (1)–(4) are complemented by two others, $\mathbf{\Omega} = [\Omega_R, \Omega_L]^T$, which represent the angular velocities expressed in rad/s of the right and left motors, forming the state vector used in the Simulink environment, $\mathbf{x} = [\mathbf{v}_g^{B^T}, \mathbf{w}_g^{B^T}, \mathbf{p}^{NED^T}, \mathbf{q}^{NED^T}, \mathbf{\Omega}^T]^T$. Likewise, the input vector is represented by $\mathbf{u} = [\delta_a, \delta_e, \tau_r, \tau_t]^T$, denoting the aileron and

elevator deflections in radians, and the rudder and thrust inputs, taken as dimensionless. These can be gathered from the original left and right elevator deflections, δ_L and δ_R, and the left and right proprotor throttle signals, τ_L and τ_R, by means of an appropriate transform [21].

Regarding $\mathbf{f}^B$ and $\mathbf{m}^B$, these portray the influence of the aerodynamics, propulsion, ground-contact effects, and gravity force in the aircraft model used in this work. In this model, the longitudinal aerodynamics was taken from [24], which is based on flat-plate theory [26,27] to derive the lift, drag, and pitching moment curves for the X-Vert, and these are complemented by the lateral aerodynamics and effects of aerodynamic derivatives described in [21]. For the purposes of this work, the assumption of the absence of wind is present, therefore resulting in a direct correspondence between ground and air velocities, $\mathbf{v}_g^B = \mathbf{v}_a^B$ and $\mathbf{w}_g^B = \mathbf{w}_a^B$ [21,25]. The propulsion subsystem is also modelled in accordance with [24,28], with the difference that it included a Brushless Direct Current (BLDC) motor model, expressing the actuator dynamics between the throttle signals, $\tau_{R/L}$, and the corresponding angular velocity of the motors, $\Omega_{R/L}$. Finally, the ground-contact effects were modelled using a spring-damper analogy [24], accounting for five contact points, and the gravity effects consist of a simple rotation of the inertial gravity vector, $\mathbf{g}^{NED} = [0, 0, g_0]^T$, into the body-frame, where g_0 represents the acceleration of gravity, taken as constant. It should be highlighted that the utilized aircraft model has not been experimentally validated, and is thus considered only as a tool for the development and validation of control strategies, with the expectation that these will be robust to eventual mismatches between this model and the real aircraft.

As this work focuses on the stabilization of a tail-sitter UAV in vertical flight, only the states of $\mathbf{x}$ related to the attitude and vertical displacement are of relevance, forming the state vector subset:

$$\mathbf{y} = [\mathbf{w}_g^{B\,T}\ ,\ \mathbf{q}^{NED\,T}\ ,\ u_g\ ,\ D]^T. \tag{5}$$

It should be noted that the velocity, u_g, is used to stabilize D as the aircraft is pitched up during vertical flight, with its x axis perpendicular to the ground plane.

The vector in (5) represents the states that must be tracked for performing the vertical flight, arising in two instances, the first being the reference vector, $\mathbf{y}_{ref}$, and the second the corresponding measured or estimated states, $\hat{\mathbf{y}}$, as shown in Figure 1.

2.2. Sensing and Estimation

In the previous research work [23], an accelerometer and gyroscope were combined in order to estimate the attitude using a quaternion-based complementary filter [29], providing the $\mathbf{q}^{NED}$ component of $\mathbf{y}$, and $\mathbf{w}_g^B$ being obtained directly from the gyroscope. This method has a single design variable, β_{CF}, which determines the relative weight of the accelerometer in the computation of the estimated quaternion, $\hat{\mathbf{q}}^{NED}$.

Regarding altitude estimation, an analogous strategy had been employed in pairing a sonar with the accelerometer to estimate u_g and D, but it was verified that it was overly sensitive to significant attitude variations as these resulted in the lack of reception of the required response signal. This caused the sensor to reach a timeout, assuming a much larger altitude, which caused unreliability of the altitude control action, and it was found to happen more frequently when the UAV pitched backward due to the partial block of the sensor by its tail. To avoid this issue, a barometer was considered for this iteration, and a simple atmosphere model was included in the simulator, illustrating the pressure, P, decrease with the altitude above sea-level, $-D_{SL}$ [21]:

$$P = P_0 \left(\frac{T_0}{T_0 - L_0 D_{SL}} \right)^{C_{atm}} \tag{6}$$

where P_0 = 101,325 Pa and $T_0 = 288.15$ K represent the pressure and temperature at sea-level, $L_0 = -0.0065$ K/m is the rate of temperature decrease in the lower atmosphere, and $C_{atm} = -6.2649 \times 10^3$ is a dimensional constant that summarizes the relevant air

properties, available in [21]. Accounting for noise and bias, b_{bar}, present in its readings, the barometer model is given by

$$P_{bar} = P + \eta_{bar} + b_{bar} \tag{7}$$

where η_{bar} is taken as a zero-mean Gaussian noise with variance, σ_{bar}, gathered from [21], and b_{bar} is assumed to be removed during the calibration process of the sensors, and therefore is zero for simulation purposes. Together with the readings from the accelerometer and gyroscope, this results in the following output vector:

$$\mathbf{z} = [\mathbf{a}_{g,acc}^{B\ T},\ \mathbf{w}_{g,gyr}^{B\ T},\ P_{bar}]^T. \tag{8}$$

It should be noted that the components of $\mathbf{z}$ are corrupted by noise in the simulation environment, attesting for the robustness of the control solutions to noise.

For the altitude estimation, the model in (6) can be simplified and inverted, and a Low-Pass Filter (LPF) should be included to discard the effects of η_{bar}, resulting in the estimated Down position being

$$D_{bar} = LPF\left(\frac{P_{bar}}{\rho_a g_0}\right). \tag{9}$$

A second-order derivative filter [23] can be used to estimate $\dot{D}$, allowing one to obtain this derivative from a noisy sensor, although introducing some delay, which is a consequence of such filtering. In turn, $\dot{D}$ is expressed in the body frame so that it can be used to provide a barometer-based estimation of the forward velocity, u_g, denoted by $u_{g,bar}$. Using both D_{bar} and $u_{g,bar}$ as observations, a two-state Kalman filter [21] was designed in order to combine the information of the barometer and accelerometer in order to provide a more reliable estimation of u_g and D. Therefore, the resulting tracking vector becomes

$$\hat{\mathbf{y}} = [\mathbf{w}_{gyr,g}^{B\ T},\ \hat{\mathbf{q}}^{NED\ T},\ \hat{u}_g,\ \hat{D}]^T \tag{10}$$

which includes all the the necessary variables—either estimated or measured—to track in order to perform vertical flight.

It is noteworthy to clarify that both the output and the estimated tracking vectors, $\mathbf{z}$ and $\hat{\mathbf{y}}$, respectively, represent variables that are also of extreme importance for the experimental trials, as the flight controller will use the components of $\mathbf{z}$ that come from the real sensors, and compute $\hat{\mathbf{y}}$ to obtain the final control action $\mathbf{u}$.

2.3. Affine Form in Hover Conditions

Both the Nonlinear Dynamics Inversion and the Backstepping control strategies require for the equations of a given system to be expressed in affine form [30], which also holds true for its incremental variations, INDI and IBKS, explored in this work. For the attitude dynamics, the affine form of (2) becomes

$$\dot{\mathbf{w}}_g^B = \mathbf{f}_{dyn} + \mathbf{G}_{dyn}\mathbf{u}_{att} \tag{11}$$

where the subscript "*dyn*" was used to denote *dynamics*, and $\mathbf{u}_{att} = [\delta_a, \delta_e, \tau_r]^T$ represents the inputs for the attitude dynamics. For near-hover conditions, the effects of the wing aerodynamics in $\mathbf{f}_{dyn}$ can be neglected and $\mathbf{G}_{dyn}$ can be taken as constant, as stated in [23]:

$$\mathbf{f}_{dyn} = \mathbf{J}_{uav}^{-1}(-\mathbf{w}_g^B \times \mathbf{J}_{uav}\mathbf{w}_g^B) \tag{12}$$

$$\mathbf{G}_{dyn} = \mathbf{J}_{uav}^{-1}\begin{bmatrix} -2d_{AC,y}k_L & 0 & 4k_Q\frac{\Omega_0^2}{\tau_{t,0}} \\ 0 & 2\bar{c}k_m + 2d_{AC,x}k_L & 0 \\ 2d_{AC,y}k_D & 0 & -4d_{p,y}k_T\frac{\Omega_0^2}{\tau_{t,0}} \end{bmatrix} \tag{13}$$

where k_L, k_D, k_m, k_T, and k_Q respectively represent constants for the lift, drag, pitching moment, thrust, and torque that result from a linearization at hover equilibrium conditions with a motor speed, Ω_0, with throttle input, $\tau_{t,0}$, described in detail in the previous work [23]. Additionally, $d_{AC,x}$ and $d_{AC,y}$ represent the x and y coordinates of the aerodynamic centre of the right side of the wing in relation to the Centre of Gravity (COG) of the UAV, whilst $d_{p,y}$ is the analogous lateral position of the proprotor, and lastly $\bar{c}$ is the Mean Aerodynamic Chord (MAC) of the tail-sitter.

In order to apply incremental control techniques, it is useful to express (11) in its incremental form under the assumptions that the input $\mathbf{u}_{att}$ has a faster effect on the dynamics of this subsystem and a small sampling time is possible [31], resulting in

$$(\dot{\mathbf{w}}_g^B)_k = (\dot{\mathbf{w}}_g^B)_{k-1} + \mathbf{G}_{dyn}\Delta\mathbf{u}_{att} \tag{14}$$

where k stands for the current time instant and $\Delta\mathbf{u}_{att}$ is the increment of the control input. Naturally, (14) requires that the acceleration of the previous time-step, $(\dot{\mathbf{w}}_g^B)_{k-1}$, is accessible, either via direct measurement or through a suitable estimation method.

For control purposes, the cross components of G_{dyn} can be neglected, resulting in a simpler control effectiveness matrix, with the following numerical value for the X-Vert [23]:

$$\mathbf{G}_{dyn} \approx \begin{bmatrix} -25.492 & 0 & 0 \\ 0 & -95.726 & 0 \\ 0 & 0 & -274.151 \end{bmatrix}. \tag{15}$$

Moving on to the attitude kinematics, (4) can also be expressed in affine form, this time considering $\mathbf{w}_g^{NED}$ as the input of this subsystem:

$$\dot{\mathbf{q}}^{NED} = \mathbf{f}_{kin} + \mathbf{G}_{kin}\mathbf{w}_g^B \tag{16}$$

where the subscript "*kin*" symbolizes *kinematics*. Under this form, the $\mathbf{f}_{kin}$ vector takes only null values, and the respective $\mathbf{G}_{kin}$ matrix varies with $\mathbf{q}^{NED}$:

$$\mathbf{G}_{kin} = \frac{1}{2}\begin{bmatrix} q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}. \tag{17}$$

3. Incremental Control Methods for Attitude Stabilization

3.1. INDI Attitude Control

The increment of the INDI control law for the attitude dynamics can be directly deduced from the inversion of (14), taking $\dot{\mathbf{w}}_{g,d}^B$ as the desired dynamics of the system and λ^{INDI} as an Input Scaling Gain (ISG) [32]:

$$\Delta\mathbf{u}_{INDI} = \lambda^{INDI}\mathbf{G}_{dyn}^{-1}\left(\dot{\mathbf{w}}_{g,d}^B - (\dot{\mathbf{w}}_g^B)_{k-1}\right) \tag{18}$$

which must be added to the control input of the previous time-step, $(\mathbf{u}_{INDI})_{k-1}$, to compute the final value. It should be noted that the subscript *att* is omitted in (18) to not overburden the notation, as the INDI (and IBKS) is only applied to the attitude dynamics in this work. In (18), the desired dynamics are defined by

$$\dot{\mathbf{w}}_{g,d}^B = \mathbf{K}_\mathbf{w}(\mathbf{K}_\mathbf{q}\mathbf{q}_{v,err} + \mathbf{w}_g^B) \tag{19}$$

in which $\mathbf{q}_{v,err}$ is the vectorial component of the error quaternion—defined by the quaternion product between the conjungate of the estimated quaternion, $\hat{\mathbf{q}}^{NED*}$, and the respective reference quaternion to be tracked, $\mathbf{q}_{ref}^{NED}$—and $\mathbf{K}_\mathbf{w}$ and $\mathbf{K}_\mathbf{q}$ are diagonal gain matrices.

As stated before, the INDI control strategy assumes that $\dot{\mathbf{w}}_g^B$ is available. This vector can be estimated from $\mathbf{w}_{g,gyr}^B$ at each time-step using the following second-order derivative filter [23,33]:

$$SD(s) = \frac{\omega_{SD}^2 s}{s^2 + 2\zeta\omega_{SD}s + \omega_{SD}^2} \tag{20}$$

where the damping coefficient is constant, $\zeta = 2$, making the cutoff frequency, ω_{SD}, the only tunable parameter of (20).

It is also useful to include the possibility of a command filter, acting simultaneously as a saturation in ensuring that $\mathbf{u}_{INDI}$ is bound to the limits of the actuators, and as a low-pass filter, with a time-constant τ_{CF} [33], given by the expression

$$CF(s) = \frac{1}{\tau_{CF}s + 1}. \tag{21}$$

Therefore, for the INDI control strategy, ω_{SD}^{INDI}, λ^{INDI}, and τ_{CF}^{INDI} assume the role of design variables that help shape the controller response, together with the two gain matrices, $\mathbf{K_w}$ and $\mathbf{K_q}$.

3.2. IBKS Attitude Control

In this work, similarly to the INDI strategy, the IBKS method is applied to the attitude dynamics subsystem in (14), as it is the largest source of model uncertainties [34]. Therefore, the attitude kinematics are controlled using a standard Backstepping loop, which starts by defining the error variable of the kinematics subsystem, $\mathbf{z}_{kin} = -\mathbf{q}_{v,err}$ [35], with its time-derivative being

$$\dot{\mathbf{z}}_{kin} = \dot{\mathbf{q}}_{v,err} = \mathbf{G}_{kin,err}\mathbf{w}_g^B \tag{22}$$

where the additional subscript in $\mathbf{G}_{kin,err}$ is used to underline that this matrix is a function on the error quaternion, $\mathbf{q}_{err}$, meaning that $\mathbf{G}_{kin,err} = \mathbf{G}_{kin}(\mathbf{q}_{err})$.

Proposing $V_1 = \frac{1}{2}\mathbf{z}_{kin}^T\mathbf{z}_{kin}$ as the Candidate Lyapunov Function (CLF) for this subsystem, its time-derivative is given by

$$\dot{V}_1 = \mathbf{z}_{kin}^T\dot{\mathbf{z}}_{kin} = \mathbf{q}_{v,err}^T\mathbf{G}_{kin,err}\mathbf{w}_g^B. \tag{23}$$

Defining α as a stabilizing function to be tracked by $\mathbf{w}_g^B$, expressed by

$$\alpha = \mathbf{G}_{kin,err}^{-1}\left(-\mathbf{K}_1\mathbf{q}_{v,err}\right) \tag{24}$$

it can be verified that $\dot{V}_1$ becomes negative-definite for $\mathbf{w}_g^B = \alpha$ and a positive-definite diagonal matrix, $\mathbf{K}_1$, consisting of a set of gains for the controller, as described in [36]:

$$\dot{V}_1 = \mathbf{q}_{v,err}^T\mathbf{G}_{kin,err}\alpha = \mathbf{q}_{v,err}^T\left(-\mathbf{K}_1\mathbf{q}_{v,err}\right) < 0 \tag{25}$$

thus resulting in asymptotic stability.

Regarding the inverse of $\mathbf{G}_{kin,err}$ in (24), the determinant of this matrix can be analytically obtained and verified to be equal to $\frac{q_{0,err}}{8}$, guaranteeing that $\mathbf{G}_{kin,err}^{-1}$ can be computed for $q_{0,err} \neq 0$. Under stabilized flight conditions, the controller will make efforts to drive the UAV to the reference attitude described by $\mathbf{q}_{ref}^{NED}$ in an attempt to minimize $\mathbf{q}_{v,err}$, and therefore it is reasonable to assume that $q_{0,err}$ will take values close to 1, ensuring that $\mathbf{G}_{kin,err}^{-1}$ can be computed for the expected flight conditions.

Taking the second error variable as $\mathbf{z}_{dyn} = \mathbf{w}_g^B - \alpha$, its time-derivative is given by

$$\dot{\mathbf{z}}_{dyn} = \dot{\mathbf{w}}_g^B - \dot{\alpha} = \left(\dot{\mathbf{w}}_g^B\right)_{k-1} + \mathbf{G}_{dyn}\Delta\mathbf{u}_{att} - \dot{\alpha} \tag{26}$$

where the incremental form of the attitude dynamics in (14) was already accounted for [34]. Incorporating the contribution of $\mathbf{z}_{dyn}$ in V_1, an augmented CLF can be formed such that $V = V_1 + V_2 = \frac{1}{2}\mathbf{z}_{kin}^T\mathbf{z}_{kin} + \frac{1}{2}\mathbf{z}_{dyn}^T\mathbf{z}_{dyn}$, which has the following expression for its time-derivative.

$$\dot{V} = \dot{V}_1 + \dot{V}_2 = \mathbf{z}_{kin}^T\dot{\mathbf{z}}_{kin} + \mathbf{z}_{dyn}^T\dot{\mathbf{z}}_{dyn} \tag{27}$$

Accounting now for $\dot{\mathbf{z}}_{kin} = \mathbf{G}_{kin,err}\mathbf{w}_g^B$ and $\mathbf{w}_g^B = \mathbf{z}_{dyn} + \alpha$, the final CLF can be rewritten so that it includes the expression for the stabilizing function, α, in (24), becoming

$$\dot{V} = \mathbf{z}_{kin}^T\mathbf{G}_{kin,err}\big(\mathbf{z}_{dyn} + \mathbf{G}_{kin,err}^{-1}(-\mathbf{K}_1\mathbf{z}_{kin})\big) + \mathbf{z}_{dyn}^T\big((\dot{\mathbf{w}}_g^B)_{k-1} + \mathbf{G}_{dyn}\Delta\mathbf{u}_{att} - \dot{\alpha}\big) \tag{28}$$

which can be further simplified into

$$\begin{aligned}\dot{V} &= -\mathbf{K}_1\mathbf{z}_{kin}^T\mathbf{z}_{kin} + \mathbf{z}_{dyn}^T\mathbf{G}_{kin,err}\mathbf{z}_{kin} + \mathbf{z}_{dyn}^T\big((\dot{\mathbf{w}}_g^B)_{k-1} + \mathbf{G}_{dyn}\Delta\mathbf{u}_{att} - \dot{\alpha}\big)\\ &= -\mathbf{K}_1\mathbf{z}_{kin}^T\mathbf{z}_{kin} + \mathbf{z}_{dyn}^T\big((\dot{\mathbf{w}}_g^B)_{k-1} + \mathbf{G}_{dyn}\Delta\mathbf{u}_{att} - \dot{\alpha} + \mathbf{G}_{kin,err}^T\mathbf{z}_{kin}\big)\end{aligned}. \tag{29}$$

Similarly to $\dot{V}_1$, $\dot{V}$ can be forced to be negative-definite by adequately designing $\Delta\mathbf{u}_{att}$ [37], such as

$$\Delta\mathbf{u}_{IBKS} = \mathbf{G}_{dyn}^{-1}\big(-\mathbf{K}_2\mathbf{z}_{dyn} - (\dot{\mathbf{w}}_g^B)_{k-1} + \dot{\alpha} - \mathbf{G}_{kin,err}^T\mathbf{z}_1\big) \tag{30}$$

where the subscript "$IBKS$" was included in the expression for the incremental control law. It can be verified that substituting (30) into (29) yields $\dot{V} = -\mathbf{K}_1\mathbf{z}_{kin}^T\mathbf{z}_{kin} - \mathbf{K}_2\mathbf{z}_{dyn}^T\mathbf{z}_{dyn}$, thus ensuring it is always negative-definite for positive-definite gain matrices $\mathbf{K}_1$ and $\mathbf{K}_2$, and grating asymptotic stability of the controlled system.

The last step required for the implementation of (30) is to determine the time-derivative of α. For this research, a decision was made to compute $\dot{\alpha}$ analytically, although assuming a time-scale separation between the dynamics and kinematics by considering $\mathbf{G}_{kin,err}$ to be constant at each time-step, resulting in

$$\dot{\alpha} = \mathbf{G}_{kin,err}^{-1}\big(-\mathbf{K}_1\mathbf{G}_{kin,err}\mathbf{w}_g^B\big). \tag{31}$$

Similarly to the INDI controller, the incremental backstepping approach assumes that $\dot{\mathbf{w}}_g^B$ is available, and thus the second-order derivative filter in (20) was used for this purpose. It is also useful to include an IGS in (30), as well as a command-filter, in order to ensure robustness to sensor noise and smooth actuation, specially due to the introduction of this estimation method. This results in the IBKS control strategy having an analogous set of design variables to INDI: ω_{SD}^{IBKS}, λ^{IBKS}, τ_{CF}^{IBKS}, $\mathbf{K}_1$ and $\mathbf{K}_2$.

4. Hardware-in-the-Loop Validation

In order to validate the INDI and IBKS control solutions for the tail-sitter UAV, these were first tested in the described Simulink environment, followed by the implementation on a Micro-Controller Unit (MCU) in order to perform Hardware-in-the-Loop (HITL) simulations. Some considerations about this implementation are now provided.

- Altitude control:
  The incremental attitude control solutions were paired with the same altitude and vertical velocity controller, ensuring analogous testing conditions. This control strategy starts by defining a desired forward force [24],

  $$F_d = m_{uav}(2(q_0q_2 - q_1q_3))(g_0 - k_D D_{err}) + m_{uav}k_u u_{g,err} \tag{32}$$

  where D_{err} and $u_{g,err}$ are, respectively, the deviations of D and u_g regarding their reference values, and k_D and k_u are the adjustable gains for this controller. Knowing the desired forward force from (32), the propeller model can be inverted, allowing for the computation of the angular velocities of the motors, which is then used to determine the thrust input, τ_t [23,24].

- Setup description:
  As in the research work that preceded these developments [23], the setup used in HITL simulations consisted of an Arduino Nano 33 IOT [38] connected to a W5500 Ethernet module [39] via a Serial Peripheral Interface (SPI) protocol. This enables sensor data from the simulated aircraft model in MATLAB to be sent to the MCU, together with the references $\mathbf{y}_{ref}$, which proceeds to perform the necessary estimation steps and compute the control action, $\mathbf{u}$. This is then sent back to MATLAB, effectively controlling the simulated UAV. The SPI connection allows for fast bi-directional data streaming via a User Datagram Protocol (UDP) over the Ethernet, enabling the HITL simulations to be run at $T_s^{sim} = 0.005$ seconds, which was also the sampling time used to run the Arduino board and to perform the discretization of the filters described in Sections 2 and 3.
- Numerical implementation:
  It was soon verified that the IBKS implementation was more computationally-demanding than the INDI one, mainly due to the multiple matrix over vector multiplications and matrix inversions present in (24), (30) and (31). Therefore, some care was taken to optimize the code in order to minimize these costly operations. The first step in such an optimization process was to implement the different gain matrices—$\mathbf{K_w}$ and $\mathbf{K_q}$ in INDI and $\mathbf{K}_1$ and $\mathbf{K}_2$ in IBKS—as simple multiplications of the different vector components by constants, taking advantage of these matrices being diagonal and avoiding unnecessary multiplications by zeros. For a generic 3-by-3 diagonal $\mathbf{K}$ matrix with components $k_{i,i}$, $i = 1, 2, 3$ and a vector $\mathbf{v} = [v_1, v_2, v_3]^T$, this results in $\mathbf{Kv} = [k_{1,1}v_1, k_{2,2}v_2, k_{3,3}v_3]^T$. The same reasoning was applied when using $\mathbf{G}_{dyn}$, as its inverse is trivial when assumed as a diagonal matrix, as illustrated in (13).
  The only exception to this procedure comes from matrix $\mathbf{G}_{kin,err}$, which should not be simplified in order to accurately represent the attitude kinematics in (16). To avoid the computations required for its inversion at every instant, it was inverted symbolically, resorting to MATLAB, and $\mathbf{G}_{kin,err}^{-1}$ was implemented directly as a separate matrix, with the expression

$$\mathbf{G}_{kin,err}^{-1} = \frac{2}{q_{0,err}} \begin{bmatrix} (q_0^2 + q_1^2) & (q_0q_3 + q_1q_2) & (q_1q_3 - q_0q_2) \\ (q_1q_2 - q_0q_3) & 2(q_0^2 + q_2^2) & (q_0q_1 + q_2q_3) \\ (q_1q_3 + q_0q_2) & (q_2q_3 - q_0q_1) & 2(q_0^2 + q_3^2) \end{bmatrix}_{err} \tag{33}$$

  which produced faster computations. Nevertheless, the matrix and vector multiplications that involved $\mathbf{G}_{kin,err}$ or $\mathbf{G}_{kin,err}^{-1}$ were still complex, and thus a dedicated library—*BasicLinearAlgebra* [40]—was used to perform these computations.

4.1. Hardware-in-the-Loop Results

The Hardware-in-the-Loop simulations were run using the described setup, effectively allowing for the virtual UAV to be controlled by the Arduino board. The reference signals were the same as the ones in [23], starting with the take-off, followed by test rotations over the y, z, and x axis of the UAV, and ending with the landing.

The estimation and altitude control tuning parameters were kept the same in the HITL simulations for both controllers, namely $\beta_{CF}^{sim} = 0.01$, $k_D^{sim} = 10$ and $k_u^{sim} = 2$, and the filters described in Section 3 were discretized for $T_s^{sim} = 0.005$ s. Regarding the values for the parameters specific to each controller, these were:

- INDI: $\mathbf{K}_{\mathbf{w}}^{sim} = \text{diag}[10\ ,\ 5\ ,\ 10]$, $\mathbf{K}_{\mathbf{q}}^{sim} = \text{diag}[5\ ,\ 5\ ,\ 5]$, $\omega_{SD}^{INDI,sim} = 100$, $\lambda^{INDI,sim} = 0.1$, $\tau_{CF}^{INDI,sim} = 0$
- IBKS: $\mathbf{K}_1^{sim} = \text{diag}[5\ ,\ 5\ ,\ 5]$, $\mathbf{K}_2^{sim} = \text{diag}[1\ ,\ 1\ ,\ 1]$, $\omega_{SD}^{IBKS,sim} = 100$, $\lambda^{IBKS,sim} = 0.1$, $\tau_{CF}^{IBKS,sim} = 0$.

It should be noted that the HITL configuration introduces a noticeable delay when receiving the simulated sensor readings, which has a negative impact on the performance of incremental controllers mainly due to the estimation of $\dot{\mathbf{w}}_g^B$, as the second-order derivative

filter already introduces some delay. This was also the reason that motivated the absence of the low-pass filtering in the HITL simulations, evident by $\tau_{CF}^{INDI,sim} = 0$ and $\tau_{CF}^{IBKS,sim} = 0$, as it was noticed that it resulted in unwanted oscillations due to this delay. Therefore, the HITL simulations were taken as a merely qualitative validation step that preceded the experimental flight tests, and no further effort was taken to tune the above parameters. Such validation was considered satisfactory, and the results are readily presented in Figures 2 and 3, respectively, for the INDI and for the IBKS controllers, and the plots for δ_a and δ_e are shown in degrees instead of radians for easier interpretation.

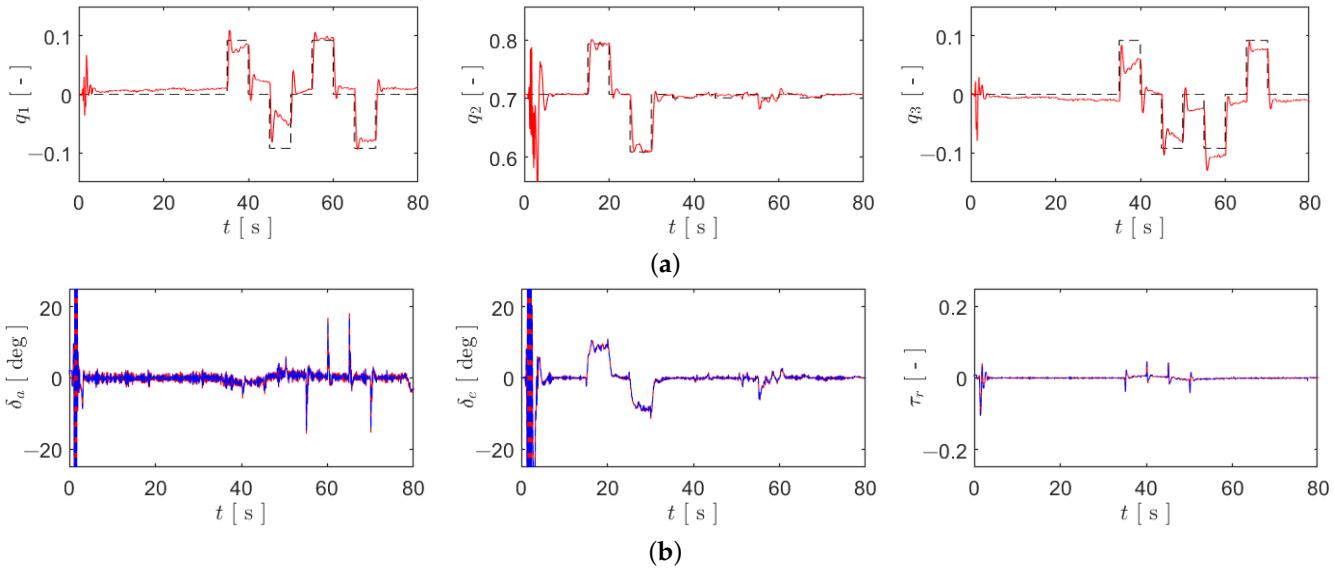

Figure 2. Plots for the HITL simulations using the INDI controller. (**a**) From left to right: tracking results for q_1, q_2, and q_3 for the INDI controller, in red, with the respective reference signal in black. (**b**) From left to right: plots of δ_a, δ_e, and τ_r for the INDI controller, in blue, compared to the respective median-filtered curves in red, applied to evaluate the oscillation.

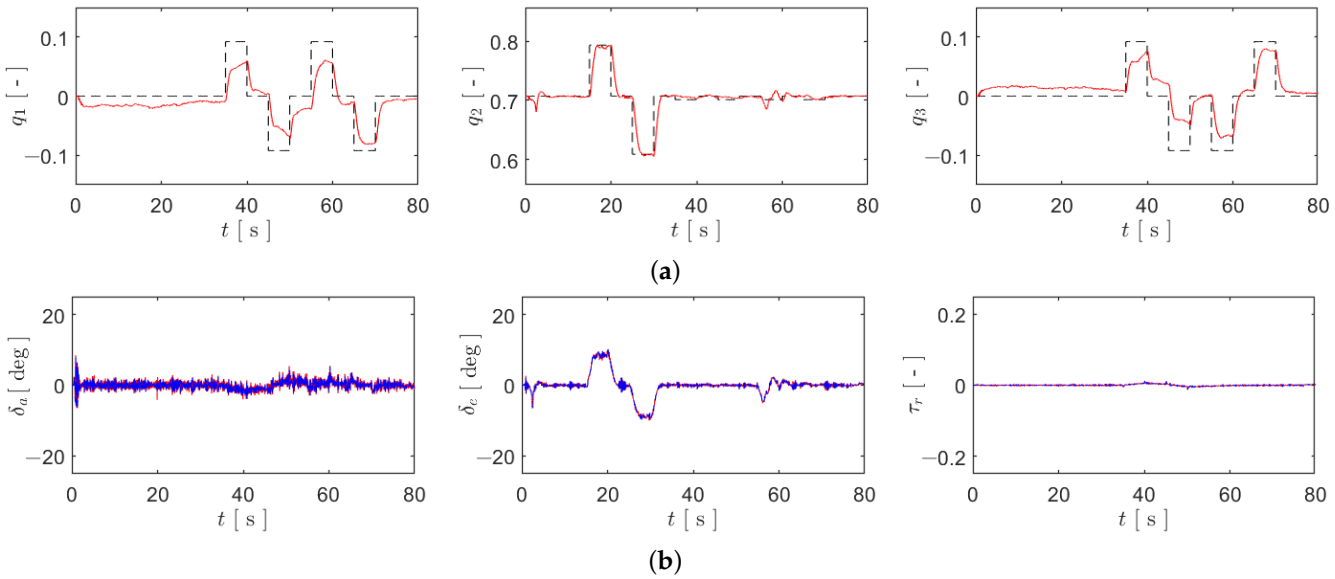

Figure 3. Plots for the HITL simulations using the IBKS controller. (**a**) From left to right: tracking results for q_1, q_2, and q_3 for the IBKS controller, in red, with the respective reference signal in black. (**b**) From left to right: plots of δ_a, δ_e, and τ_r for the IBKS controller, in blue, compared to the respective median-filtered curves in red, applied to evaluate the oscillation.

It can be verified that both incremental controllers track $\mathbf{q}_{ref}$, with a certain degree of precision specifically for q_2 but with limited accuracy for q_1 and q_3. Both control strategies present low actuator oscillation, even in the presence of significant noise in the sensors modelled in the simulator, demonstrating excellent robustness to it. Such a conclusion had already been found in [23] for the INDI control strategy, but is now extended to the IBKS. Table 1 summarizes the results for the HITL simulations, providing the Root Mean Square (RMS) values—RMS^{sim}—of each tracking error and the oscillation of each actuator measured by μ^{sim}, both metrics having been previously described in detail in [23].

The average values of both of these metrics, $\overline{RMS}_q^{sim}$ and $\overline{\mu}_{u_{att}}^{sim}$, respectively, are also shown. Inspecting this table, it can be stated that it effectively validates the implementations of these two incremental control strategies.

Table 1. Summary of the Hardware-in-the-Loop results, expressed in the respective units.

Cont.	$RMS_{q_1}^{sim}$	$RMS_{q_2}^{sim}$	$RMS_{q_3}^{sim}$	$\overline{RMS}_q^{sim}$	$\mu_{\delta_a}^{sim}$	$\mu_{\delta_e}^{sim}$	$\mu_{\tau_r}^{sim}$	$\overline{\mu}_{u_{att}}^{sim}$
INDI	0.0214	0.0125	0.0215	0.0185	0.0030	0.0009	0.0003	0.0014
IBKS	0.0299	0.0169	0.0289	0.0252	0.0033	0.0010	0.0003	0.0015

4.2. Analysis of Computational Resources

Despite the similarity in the results, as illustrated by Table 1, the INDI and IBKS differ substantially in the complexity of the required computations to calculate the control increment Δu_{att}. Assuming the estimation of $\mathbf{q}^{NED}$ and $\dot{\mathbf{w}}_g^B$ and the calculation of the error variables as common steps for both controllers, the INDI strategy can be implemented without any sort of matrix products, since $\mathbf{G}_{dyn}^{-1}$, $\mathbf{K_w}$, and $\mathbf{K_q}$ are diagonal matrices. However, this is not the case for the IBKS, which requires several products of full 3-by-3 matrices, resulting in added computational effort. This was soon noticed in the HITL simulations, were it was found that the Arduino was struggling in performing the necessary computations for the IBKS controller under the specified sample time $T_s = 0.005$ s. Although this is a clear evidence of a limitation of the chosen MCU, it was considered as an interesting aspect to analyse, as it explored a fundamental difference of two otherwise very similar control strategies. Figure 4 shows the plots for the actual cycle time, dT, during HITL simulations, with the considered time-span being 5 s after take-off and 5 s before landing, to avoid any transient behaviour.

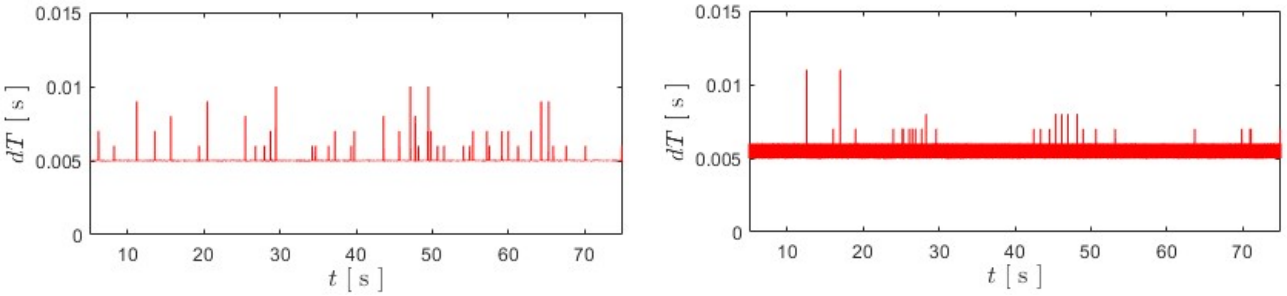

Figure 4. Plot of dT for the INDI Hardware-in-the-Loop simulation (**left**) and an analogous plot for the IBKS (**right**).

As can be seen, the Arduino board manages to perform the necessary computations for the INDI control law under the imposed sample time $T_s^{sim} = 0.005$ s, with just some occasional spikes. Contrasting with this, the dT interval in the simulation using the IBKS kept oscillating between 0.005 and 0.006 seconds, evidencing that this is a more computationally demanding control law. These results are summarized by the mean value and standard deviation of dT for the INDI and IBKS controllers, respectively represented by μ_{dT} and σ_{dT}, and shown in Table 2. Analysing this table, it can be concluded that the IBKS control law, although having the advantages of proof of global stability, is clearly

more demanding on computational resources, presenting a natural trade-off between these two aspects.

Table 2. Summary of sampling time analysis in Hardware-in-the-Loop simulations, expressed in seconds.

Cont.	μ_{dT} [s]	σ_{dT} [s]
INDI	0.005008	0.000157
IBKS	0.005334	0.000483

5. Experimental Validation

After the successful validation of the INDI and IBKS implementations in a HITL scenario, these controllers were tested to stabilize an experimental prototype, in indoor and controlled conditions.

5.1. Description of the Experimental Setup

5.1.1. Flight Controller

The Flight Controller (FC) used in this research work is an iteration of the one previously developed in [23], which was based on an Arduino Nano 33 IOT since it has both an onboard 6 Degree-of-Freedom (DOF) Inertial Measurement Unit (IMU), comprising an accelerometer and gyroscope, and Wi-Fi capabilities. However, as verified in previous research, the Wi-Fi communications were demanding in terms of processing time, and thus radio communications were used instead. A low-cost and quite popular radio transceiver was used, the nRF24L01 by Nordic Semiconductor [41], together with the *RF24* library [42] to enable its integration with the Nano 33 IOT. The second change made to the FC was the inclusion of a barometer for altitude estimation instead of a sonar, as already mentioned in Section 2. The barometer chosen was the MS5611 [43], which was connected to the Arduino via an Inter-Integrated Circuit (I2C) protocol, as specified in the utilized library [44]. The final assembly of the flight controller used in the flight tests is shown in Figure 5.

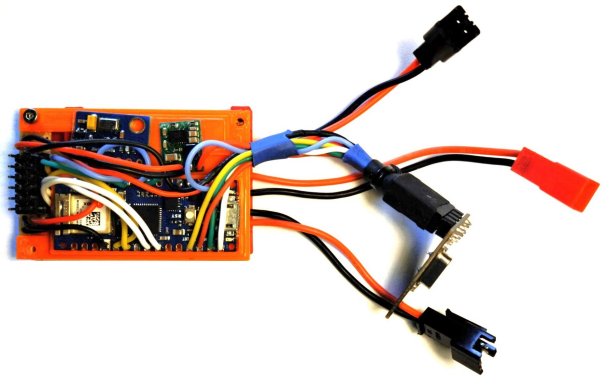

Figure 5. Flight controller assembly used for the experimental trials.

The remaining electronic components were the same as the ones in [23], in particular the motors, the Electronic Speed Controllers (ESC), the Power Distribution Board (PDB), and the battery. It should however be noted that the PDB was included as part of the FC assembly, in order to save space in the fuselage of the X-Vert. Reflective markers were also placed in the experimental prototype in order for a Motion Capture System (MCS) to be able to track its movement, as will be described next. The experimental prototype is shown in Figure 6.

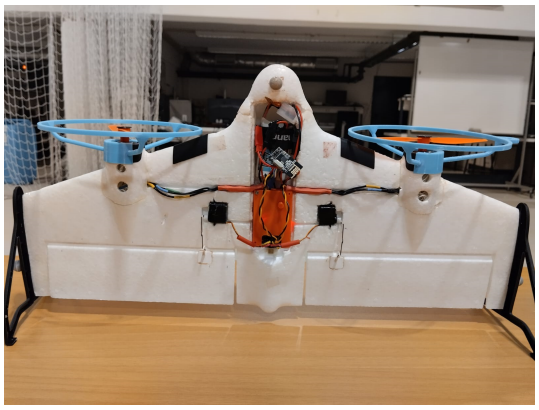

Figure 6. Experimental prototype based on the X-Vert VTOL.

5.1.2. Ground Station

To accommodate the transition to radio communications, a suitable interface that allowed MATLAB to communicate with the FC had to be designed. A Raspberry Pi Pico [45] was chosen as the MCU of this ground station, mainly due to its dual-core 133 MHz processor, compatibility with the Arduino ecosystem [46], and two SPI ports. This allowed for the Pi Pico to be simultaneously paired with a W5500 Ethernet module and a nRF24L01 with independent SPI connections, and the final ground station assembly can be seen in Figure 7. It should be highlighted that the ground station was run at 100 Hz, so that it could perform the Ethernet/radio bridging without losing data packets.

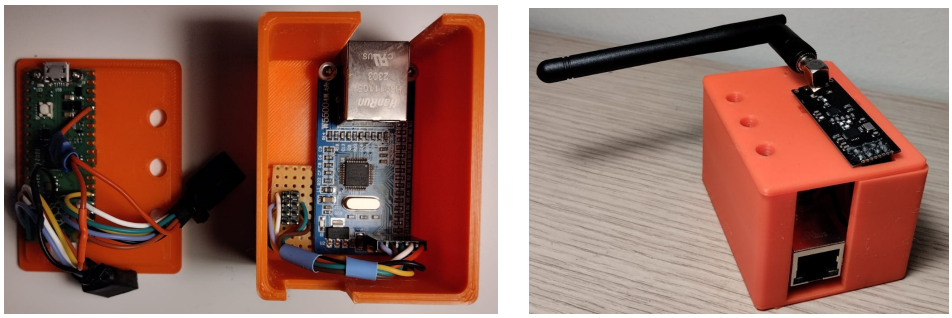

Figure 7. Disassembled ground station with the Raspberry Pi Pico and the W5500 Ethernet module (**left**) and the assembled version with the nRF24L01 (**right**).

5.1.3. Ground Truth

The experimental trials were conducted in an indoor, net-protected area covered by a Qualisys MCS. Such a system enabled the accurate and precise tracking of the movement and attitude of the X-Vert during the flight tests, providing the ground truth for evaluate of the performance of the controllers. Figure 8 shows an example of the tracking of the X-Vert by the MCS during the experimental flight trials.

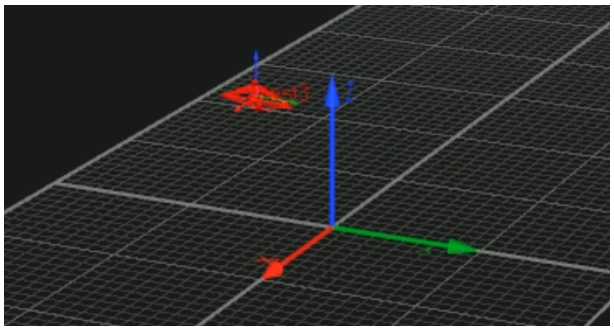

Figure 8. Reconstruction of the X-Vert using the MCS.

5.2. Experimental Results

The flight trials were conducted using the experimental prototype shown in Figure 6, controlled by the Arduino board, which computed the estimated quaternion, altitude, and the control action and returned them to MATLAB via the previously-described ground station. The MCS was then used as a validation mechanism, allowing one to track the attitude of the UAV with a separate and independent tool. A sample time of 0.01 s was chosen in order to ensure that the MCU could perform the computations required by the IBKS controller at regular intervals, and the filters were re-designed accordingly. The parameters for the estimation methods and altitude controller were kept the same for both the INDI and the IBKS,

$$\beta_{CF}^{exp} = 0.05 \quad , \quad k_D^{exp} = 2 \quad , \quad k_u^{exp} = 1 \quad , \quad T_s^{exp} = 0.01 \tag{34}$$

ensuring that these would not influence the performance of the attitude controllers. On the other hand, the relevant controller-specific parameters have the following values:

- INDI: $\mathbf{K_w}^{INDI,exp} = \text{diag}[15\ ,\ 20\ ,\ 15]$, $\mathbf{K_q}^{INDI,exp} = \text{diag}[15\ ,\ 15\ ,\ 15]$, $\lambda^{INDI,exp} = 0.8$, $\omega_{SD}^{INDI,exp} = 50$, $\tau_{CF}^{INDI,exp} = 0.01$.
- IBKS: $\mathbf{K_1}^{IBKS,exp} = \text{diag}[5\ ,\ 5\ ,\ 5]$, $\mathbf{K_2}^{IBKS,exp} = \text{diag}[15\ ,\ 15\ ,\ 15]$, $\lambda^{IBKS,exp} = 0.7$, $\omega_{SD}^{IBKS,exp} = 50$, $\tau_{CF}^{IBKS,exp} = 0.01$.

When it comes to the parameter tuning, it was soon verified that it was useful to specify a value for ω_{SD} that would be satisfactory, and then choose τ_{CF} in order to ensure a smooth, oscillation-free actuation. In turn, the gain matrices and ISG could be fine-tuned afterwards to achieve the desired control action. Nevertheless, the tuning of the IBKS was a challenge due to the influence of $\mathbf{K_1}^{IBKS,exp}$ in the several steps of the implementation, whereas the $\mathbf{K_w}^{INDI,exp}$ and $\mathbf{K_q}^{INDI,exp}$ matrices for the INDI controller were easily fine-tuned separately.

Regarding the references to be tracked, the same decision was taken to that in the previous research work [23] of exploring primarily the ability of tracking the longitudinal references in q_2, due to its importance for future tail-sitter transition manoeuvrers and the relatively limited indoor area available for the flight tests. Nevertheless, a brief lateral non-zero reference for q_3 was introduced in the trials to evaluate if the controllers could track it whilst already tracking q_2. Accounting for this, the experimental results for the INDI controller are shown in Figure 9.

As can be seen from inspecting the aforementioned figure, the INDI controller tracks the reference q_2 with precision, whilst also responding to the request in q_3. Some small overshoot moments are present, but these did not compromise the stabilization of the UAV during the flight trials. It should be noted that the discrepancy between the estimated and ground-truth values for q_1 and q_3 may be attributed to a subtle non-zero rotation before take-off, which is only perceived by the MCS as the Arduino assumes that both q_1 and q_3 are zero at the beginning of each trial. Regarding the actuation, it can be seen from the same figure that the INDI controller generates a fairly smooth control action, even in the presence of sensor noise, only having some small spikes in δ_e. Summarizing, it is possible to verify that the INDI controller manages to stabilize the aircraft in vertical flight and to track the provided references, whilst having a smooth actuation.

Moving on to the IBKS controller, the attitude tracking results for these flight tests are shown in Figure 10.

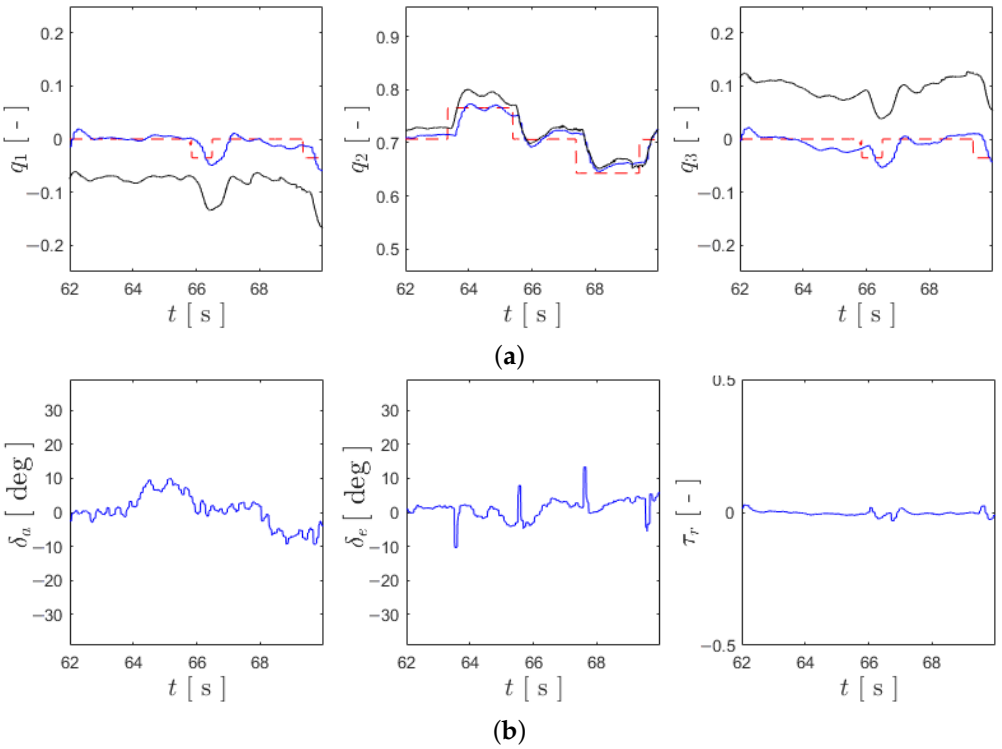

Figure 9. Plots for the attitude control of the experimental flight test using the INDI controller. (**a**) From left to right: tracking results for q_1, q_2, and q_3 for the INDI controller, with the references shown in dashed red, the ground truth in black, and the estimated quaternion in blue. (**b**) From left to right: plots of δ_a, δ_e, and τ_r for the INDI controller, in blue.

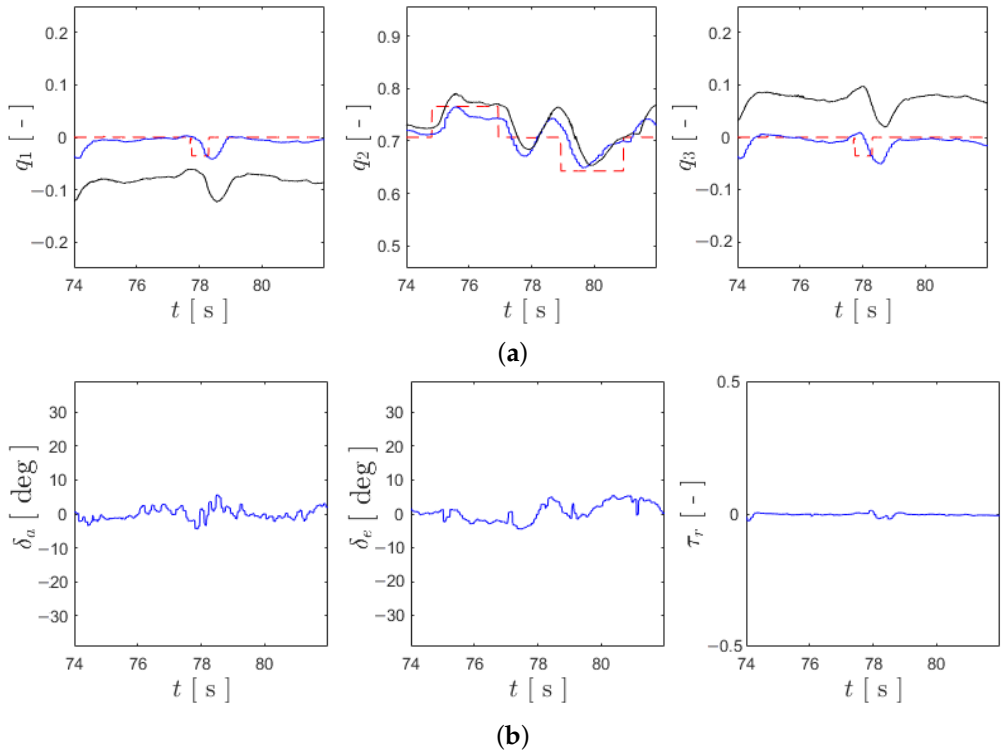

Figure 10. Plots for the attitude control of the experimental flight test using the IBKS controller. (**a**) From left to right: tracking results for q_1, q_2, and q_3 for the IBKS controller, with the references shown in dashed red, the ground truth in black, and the estimated quaternion in blue. (**b**) From left to right: plots of δ_a, δ_e, and τ_r for the IBKS controller, in blue.

Inspecting Figure 10, it is clear that the IBKS controller demonstrates a much smoother tracking of q_1 and q_3, but, when it comes to q_2 its performance is less satisfactory, showing significant overshoot, which highlights the aforementioned difficulty in tuning this controller. Regarding the actuation, the IBKS also allows for a smooth control action, even in the presence of sensor noise, with minimal oscillation and without significant spikes. Therefore, it can be concluded that this controller was also able to stabilize the prototype in vertical flight and perform the desired manoeuvrers, demonstrating that it is fully capable of controlling a tail-sitter UAV, with robustness to sensor noise.

The experimental results for the attitude control are summarized in Table 3, where both the RMS of the quaternion tracking error and actuator oscillations are provided, similarly to what was done for the results for the HITL simulations. Inspecting this table, it can be seen that it agrees with the previous considerations, demonstrating that both controllers are capable of stabilizing the X-Vert and to track the provided references, as evidenced by the relatively low values for $\overline{RMS}_q^{exp}$. Furthermore, it can be concluded that both the INDI and IBKS control strategies allow for a smooth actuation due to the small values of $\overline{\mu}_{u_{att}}^{exp}$, demonstrating excellent robustness to sensor noise.

Table 3. Summary of the experimental results, expressed in the respective units.

Cont.	$RMS_{q_1}^{exp}$	$RMS_{q_2}^{exp}$	$RMS_{q_3}^{exp}$	$\overline{RMS}_q^{exp}$	$\mu_{\delta_a}^{exp}$	$\mu_{\delta_e}^{exp}$	$\mu_{\tau_r}^{exp}$	$\overline{\mu}_{u_{att}}^{exp}$
INDI	0.0796	0.0283	0.1024	0.0701	0.0045	0.0081	0.0011	0.0046
IBKS	0.0822	0.0382	0.0769	0.0658	0.0042	0.0038	0.0007	0.0029

The last aspect of the experimental flights to analyse is the altitude control which, although not the focus of this research work, should also be the topic of some considerations. These results are shown in Figure 11, which comprises the altitude and thrust input plots for both the INDI and IBKS controllers.

As can be seen from the inspection of the aforementioned figure, none of the controllers tracks the reference altitude value with precision, and the experimental flight for the INDI controller has significant oscillation. Such oscillation may be attributed to the low-pass filtering used in obtaining the estimated altitude, but it was considered necessary in order to reduce the noise of the sensor measurements to acceptable values to be used in the altitude controller. The difficulty in tracking D with more precision could be compensated with higher values for k_D and k_u, but it was found that this would cause additional oscillation, and therefore these results for altitude control were considered satisfactory as they enabled successful take-off, altitude hold, and landing with a smooth control action τ_t. Since it was verified that a sonar was not an ideal solution for altitude estimation by itself in the previous research work [23], it is now evident that the barometer should be paired with a similar distance measurement sensor in order to accurately track the altitude, specially during take-off and landing manoeuvrers. The application of incremental solutions to the altitude control should also be evaluated, as it is expected for them to be robust to the aforementioned sensor noise, whilst allowing for precise tracking.

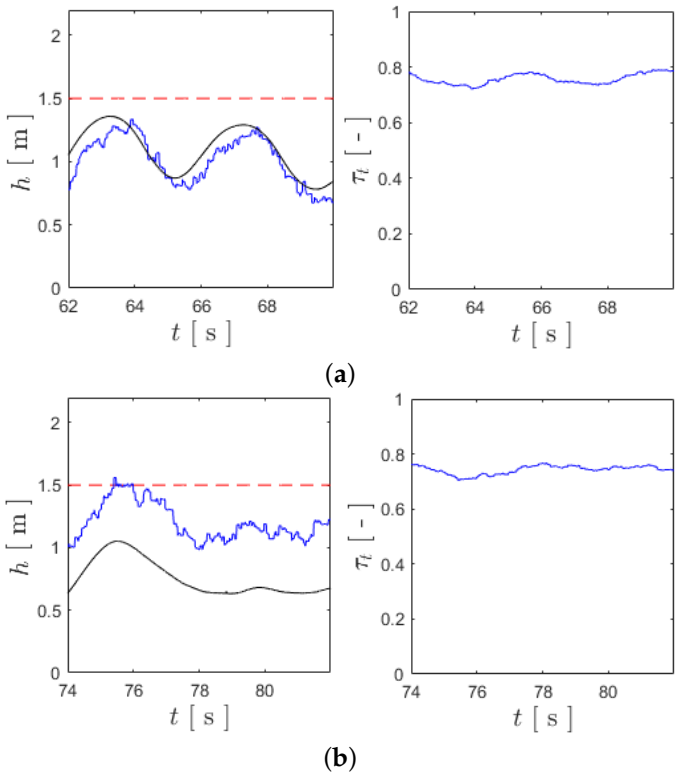

Figure 11. Plots for the experimental flight test using the IBKS controller. (**a**) Tracking results for $h = -D$ for the experimental flight with the INDI controller, with the reference in dashed red, the estimated value in blue, and the ground truth in black (**left**), and the corresponding throttle input, τ_t, in blue (**right**). (**b**) Tracking results for $h = -D$ for the experimental flight with the IBKS controller, with the reference in dashed red, the estimated value in blue, and the ground truth in black (**left**), and the corresponding throttle input, τ_t, in blue (**right**).

6. Conclusions

The research involved in this work demonstrated that both Incremental Nonlinear Dynamics Inversion and Incremental Backstepping are control strategies fully capable of controlling a tail-sitter UAV in vertical flight. Once these controllers were implemented in simulation and validated with Hardware-in-the-Loop tools, the experimental flight trials took place, conducted in an indoor environment, performing simple manoeuvrers in vertical flight and demonstrating the tracking capabilities and excellent robustness to sensor noise.

Summarizing the comparison between these two control strategies, it can be stated that the INDI provides a simple and straightforward control law, easy to implement and with low computational requirements, but it lacks a global asymptotic stability proof. The IBKS contrasts in these aspects, providing a global stability proof, but demanding several steps in obtaining the control law—all of which increase the computational demands of the MCU—and harder tuning of the controller parameters.

As future work, it is suggested to utilize a more capable MCU, preferably one capable of performing floating-point math for faster calculations, as opposed to the integer-based computations that the Arduino Nano 33 IOT employs. Furthermore, it is recommended to evaluate the application of both INDI and IBKS for aerodynamic flight, and naturally also during transitions, so that their performance can be assessed for a full tail-sitter flight. This is expected to demand for additional sensors to be included in an experimental prototype and requires further investigation. Lastly, altitude estimation and control should also be a topic of additional research, in order to ensure a robust and precise altitude tracking, whether it is by reintroducing a range sensor like a sonar, by applying incremental solutions for altitude control, or a combination of both.

Author Contributions: Conceptualization, A.A., A.M. and J.R.A.; methodology, A.A., A.M. and J.R.A.; software, A.A. and J.R.A.; validation, A.M. and J.R.A.; formal analysis, A.A.; investigation, A.A.; resources, A.M.; data curation, A.A.; writing—original draft preparation, A.A.; writing—review and editing, A.M. and J.R.A.; visualization, A.A.; supervision, A.M. and J.R.A.; project administration, A.M.; funding acquisition, A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Funds by FCT—Fundação para a Ciência e Tecnologia, I.P. under project LAETA Base Funding (DOI: 10.54499/UIDB/50022/2020) and under project Eye in the Sky (DOI: 10.54499/PCIF/SSI/0103/2018).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AC	Aerodynamic Center
AOA	Angle-of-Attack
BKS	Backstepping
BLDC	Brushless Direct Current
CF	Complementary/Command Filter
CG	Center-of-Gravity
DOF	Degree-of-Freedom
ESC	Electronic Speed Controller
FC	Flight Controller
HITL	Hardware-in-the-Loop
I2C	Inter-Integrated Circuit
IMU	Inertial Measurement Unit
IBKS	Incremental Backstepping
INDI	Incremental Nonlinear Dynamics Inversion
ISG	Input Scaling Gain
LPF	Low-Pass Filter
LQR	Linear Quadratic Regulators
MAC	Mean Aerodynamic Chord
MCS	Motion Capture System
MCU	Micro-Controller Unit
MDPI	Multidisciplinary Digital Publishing Institute
NDI	Nonlinear Dynamics Inversion
NED	North-East-Down
PCB	Prrinted Circuit Board
PID	Proportional Integral-Derivative
PWM	Pulse WidthModulation
QTM	Qualysis Track Manager
RMS	Root-Mean-Square
SD	Second-(order) Derivative
SPI	Serial Peripheral Interface
UAV	Uncrewed Aerial Vehicles
UDP	User Datagram Protocol
VTOL	Vertical Takeoff and Landing

References

1. Zhou, Y.; Zhao, H.; Liu, Y. An evaluative review of the VTOL technologies for unmanned and manned aerial vehicles. *Comput. Commun.* **2020**, *149*, 356–369. [CrossRef]
2. Rakesh, D.; Kumar, N.A.; Sivaguru, M.; Keerthivaasan, K.V.R.; Janaki, B.R.; Raffik, R. Role of UAVs in innovating agriculture with future applications: A review. In Proceedings of the 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, 8–9 October 2021.
3. Ducard, G.J.; Allenspach, M. Review of designs and flight control techniques of hybrid and convertible VTOL UAVs. *Aerosp. Sci. Technol.* **2021**, *118*, 107035. [CrossRef]
4. Saeed, A.S.; Younes, A.B.; Cai, C.; Cai, G. A survey of hybrid unmanned aerial vehicles. *Prog. Aerosp. Sci.* **2018**, *98*, 91–105. [CrossRef]
5. Yu, Z.; Zhang, Y.; Jiang, B. PID-type fault-tolerant prescribed performance control of fixed-wing UAV. *J. Syst. Eng. Electron.* **2021**, *32*, 1053–1061.
6. Kazemi, M.H.; Tarighi, R. PID-based attitude control of quadrotor using robust pole assignment and LPV modeling. *Int. J. Dyn. Control* **2024**. [CrossRef]
7. Tunik, A.A.; Nadsadna, O.I. Robust digital gain-scheduling control of the UAV altitude. In Proceedings of the 2017 IEEE 4th International Conference Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD), Kiev, Ukraine, 17–19 October 2017.
8. Wang, J.; Holzapfel, F.; Peter, F. Comparison of nonlinear dynamic inversion and backstepping controls with application to a quadrotor. In Proceedings of the EuroGNC, Delft, The Netherlands, 10–12 April 2013; pp. 1245–1263.
9. Azinheira, J.R.; Moutinho, A. Hover Control of an UAV with Backstepping Design Including Input Saturations. *IEEE Trans. Control. Syst. Technol.* **2014**, *16*, 517–526. [CrossRef]
10. Jang, K.; Bang, H.; Kim, Y. Mitigating Time-Delay in Nonlinear Dynamics Inversion for Multirotor Unmanned Aerial Vehicles. *J. Guid. Control. Dyn.* **2024**, *47*, 573–588. [CrossRef]
11. Xin, H.; Chen, Q.; Xian, Y.; Wang, P.; Wang, Y.; Yao, X.; Hou, Z. Longitudinal Decoupling Control of Altitude and Velocity for a Fixed-Wing Aircraft. *IEEE Trans. Intell. Veh.* 2024, *early access*. [CrossRef]
12. Khanna, A.; Mukherjee, B.K. UAV Performing Level Turn Maneuver Under CG Offset: Backstepping Control Scheme. In Proceedings of the 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), Gautam Buddha Nagar, India, 1–3 December 2023.
13. Lombaerts, T.; Kaneshige, J.; Schuet, S.; Hardy, G.; Aponso, B.L.; Shish, K.H. Nonlinear dynamic inversion based attitude control for a hovering quad tiltrotor eVTOL vehicle. In Proceedings of the AIAA Scitech 2019 Forum, San Diego, CA, USA, 7–11 January 2019.
14. Amiri, N.; Ramirez-Serrano, A.; Davies, R.J. Integral backstepping control of an unconventional dual-fan unmanned aerial vehicle. *J. Intell. Robot. Syst.* **2013**, *69*, 147–159. [CrossRef]
15. Cordeiro, R.A.; Azinheira, J.R.; Moutinho, A. Robustness of incremental backstepping flight controllers: The boeing 747 case study. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *57*, 3492–3505. [CrossRef]
16. Safwat, E.; Weiguo, Z.; Kassem, M.; Mohsen, A. Robust Nonlinear Flight Controller For Small Unmanned Aircraft Vehicle based on Incremental BackStepping. In Proceedings of the AIAA Scitech 2020 Forum, Orlando, FL, USA, 6–10 January 2020.
17. Wang, Y.C.; Chen, W.S.; Zhang, S.X.; Zhu, J.W.; Cao, L.J. Command-filtered incremental backstepping controller for small unmanned aerial vehicles. *J. Guid. Control. Dyn.* **2018**, *41*, 954–967. [CrossRef]
18. Tal, E.; Karaman, S. Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness. *IEEE Trans. Control. Syst. Technol.* **2020**, *29*, 1203–1218. [CrossRef]
19. Smeur, E.J.; Bronz, M.; de Croon, G.C. Incremental control and guidance of hybrid aircraft applied to a tailsitter unmanned air vehicle. *J. Guid. Control. Dyn.* **2020**, *43*, 274–287. [CrossRef]
20. Tal, E.A.; Karaman, S. Global trajectory-tracking control for a tailsitter flying wing in agile uncoordinated flight. In Proceedings of the AIAA Aviation 2021 Forum, Virtual Event, 2–6 August 2021.
21. Beard, R.W.; McLain, T.W. *Small Unmanned Aircraft: Theory and Practice*, 1st ed.; Princeton University Press: Princeton, NJ, USA, 2018.
22. E-Flite XVERT VTOL Webpage. Available online: https://www.horizonhobby.de/en_DE/product/x-vert-vtol-bnf-basic-504mm/EFL1850.html (accessed on 16 April 2024).
23. Athayde, A.; Moutinho, A.; Azinheira, J.R. Experimental Nonlinear and Incremental Control Stabilization of a Tail-Sitter UAV with Hardware-in-the-Loop Validation. *Robotics* **2024**, *13*, 51. [CrossRef]
24. Chiappinelli, R.; Nahon, M. Modeling and Control of a Tailsitter UAV. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12–15 June 2018.
25. Stevens, B.L.; Lewis, F.L. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*, 3rd ed.; John Wiley & Sons: Hoboken, NJ, USA, 2016.
26. Khan, W.; Nahon, M. Real-time modeling of agile fixed-wing UAV aerodynamics. In Proceedings of the 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, CO, USA, 9–12 June 2015.
27. Khan, W.; Nahon, M. Modeling dynamics of agile fixed-wing UAVs for real-time applications. In Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA, 7–10 June 2016.
28. Khan, W.; Nahon, M. Toward an accurate physics-based UAV thruster model. *IEEE ASME Trans. Mechatron.* **2013**, *18*, 1269–1279. [CrossRef]

29. Madgwick, S.O.; Harrison, A.J.; Vaidyanathan, R. Estimation of IMU and MARG orientation using a gradient descent algorithm. In Proceedings of the 2011 IEEE International Conference on Rehabilitation Robotics, Zurich, Switzerland, 29 June–1 July 2011.
30. Horn, J.F. Non-linear dynamic inversion control design for rotorcraft. *Aerospace* **2019**, *6*, 38. [CrossRef]
31. Coelho, R.; Moutinho, A.; Azinheira, J.R. Quadrotor Attitude Control using Incremental Nonlinear Dynamics Inversion. In Proceedings of the International Conference on Informatics in Control, Automation and Robotics, Madrid, Spain, 26–28 July 2017.
32. Azinheira, J.R.; Moutinho, A.; Carvalho, J.R. Lateral control of airship with uncertain dynamics using incremental nonlinear dynamics inversion. *IFAC Pap. Online* **2015**, *48*, 69–74. [CrossRef]
33. Cordeiro, R.A.; Marton, A.S.; Azinheira, J.R.; Carvalho, J.R.; Moutinho, A. Increased robustness to delay in incremental controllers using input scaling gain. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *58*, 1199–1210. [CrossRef]
34. Acquatella, P.; van Kampen, E.; Chu, Q.P. Incremental backstepping for robust nonlinear flight control. In Proceedings of the EuroGNC 2013, Delft, The Netherlands, 10–12 April 2013.
35. Oland, E.; Kristiansen, R. Quaternion-based backstepping control of a fixed-wing unmanned aerial vehicle. In Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA, 2–9 March 2013.
36. Khalil, H.K. *Nonlinear Systems*, 3rd ed.; Prentice Hall: Englewood Cliffs, NJ, USA, 2020.
37. Van Ekeren, W.; Looye, G.; Kuchar, R.; Chu, Q.; Van Kampen, E.J. Design, implementation and flight-test of incremental backstepping flight control laws. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Kissimmee, FL, USA, 8–12 January 2018.
38. Arduino Nano 33 IOT Product Page. Available online: https://docs.arduino.cc/hardware/nano-33-iot/ (accessed on 16 April 2024).
39. Wiznet W5500 Product Page. Available online: https://www.wiznet.io/product-item/w5500/ (accessed on 16 April 2024).
40. Basic Linear Algebra Library Page, by Tom Stewart. Available online: https://github.com/tomstewart89/BasicLinearAlgebra (accessed on 16 April 2024).
41. Nordic Semiconductor nRF24 Series Product Page. Available online: https://www.nordicsemi.com/Products/nRF24-series (accessed on 16 April 2024).
42. NRF24L01 Library Page, by TMRh20. Available online: https://nrf24.github.io/RF24/ (accessed on 16 April 2024).
43. MS5611 Barometer Product Page. Available online: https://www.okystar.com/product-item/gy-63-ms5611-atmospheric-pressure-sensor-module-height-sensor-module-iicspi-communication-oky3232/29/ (accessed on 16 April 2024).
44. MS5611 Pressure and Temperature Sensor Library Page, by Rob Tillaart. Available online: https://www.arduino.cc/reference/en/libraries/ms5611/ (accessed on 16 April 2024).
45. Raspberry Pi Pico Product Page. Available online: https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html (accessed on 16 April 2024).
46. Raspberry Pi Pico Arduino Core Page, by Earle F. Philhower, III. Available online: https://arduino-pico.readthedocs.io/en/latest/ (accessed on 16 April 2024).