Online Nonlinear Identification of Unmanned Aerial Vehicle Dynamics with Multi-Forgetting-Factor Recursive Least Squares.

Lucas Sodré
Laboratório de Controle e Sistemas
Universidade Federal do Pará (UFPA)
Belém, Brasil
lucas.sodre@itec.ufpa.br

Antonio Silveira
Laboratório de Controle e Sistemas
Universidade Federal do Pará (UFPA)
Belém, Brasil
asilveira@ufpa.br

Rufin Azonsivo
Laboratório de Controle e Sistemas
Universidade Federal do Pará (UFPA)
Belém, Brasil
rufin.azonsivo@itec.ufpa.br

Matheus Morais da Silva
Laboratório de Controle e Sistemas
Universidade Federal do Pará (UFPA)
Belém, Brasil
matheus.morais@itec.ufpa.br

***Abstract*—Unmanned aerial vehicles are increasingly offering several benefits to society. The series of functional characteristics for these actions justify this importance. However, such systems present challenges, as they are multi-input and multi-output systems, time-varying and non-linear. Given this, this work proposes the application of an extended Recursive Least Squares algorithm with Multiple Forgetting Factors to identify, in real-time, the external and internal dynamic parameters of the Parrot AR. Drone 2.0 drone, considering its non-linear nature. Using the Python 3 PS-Drone library for data collection and modeling based on discretized differential equations, the study incorporates dynamic couplings between the axes and parametric variations caused by operational conditions. The proposed algorithm allows a differentiated adaptation for each parameter, improving the estimation accuracy and the robustness of the model against noise and sudden changes. Controlled experiments demonstrated that the technique obtains excellent adherence to the real behavior of the Quadcopter, obtaining a Normalized Root Mean Squared Error index greater than 0.8 in all system states. The parametric convergence of the proposed algorithm was compared to other algorithms based on Recursive Least Squares, making it evident that it is more sensitive to the detection of operational failures, reinforcing its potential for applications in adaptive control and real-time diagnostics.**

***Index Terms*—Autonomous Vehicles and Drones; Diagnosis, Prognosis and System Identification.**

I. Introduction

Quadcopters have gained relevance in several applications, including environmental monitoring, military operations, and autonomous deliveries. The efficient performance of these tasks requires precision in automatic flight and reliable detection of operational failures, factors that critically depend on the accurate estimation of the dynamic parameters that govern angles, velocities, and positions during the mission [1]. However, these embedded systems present significant challenges due to their inherent complexity, as they are non-linear and have multiple inputs and outputs (MIMO), becoming a challenge in control, automation, and system identification.

Given these characteristics, several works [2] [3] [4] [5] make proposals to limit unwanted dynamics and to obtain more simplified models and from there, apply the most diverse control techniques. One approach tries to linearize the system surrounding the hovering maneuver and, thus, work with slight variations of the states [6], applying techniques already consolidated in aerospace systems control.

One approach to overcome these limitations is to use feedback linearization, a technique that compensates for system nonlinearities through feedback, effectively converting it into a linear system. This methodology has been successfully applied in studies such as [7], where the linearized model was combined with predictive control (MPC), demonstrating satisfactory results in a simulated environment. However, dynamic parameters such as inertia, mass, and engine rotation speed vary significantly between drones — even within the same model — and are sensitive to external (e.g., wind, altitude) and internal (high engine temperatures) operating conditions, which can alter the internal dynamic relationships of the system during flight.

It is essential to employ adaptive identification algorithms that can track dynamic changes in real-time to deal with the parametric variations that naturally occur during drone operation. The research presented in [8] demonstrates the effectiveness of extended recursive least squares (RLS) for state estimation, offering not only a smoothed computation of the parameters but also an explicit modeling of the perturbations affecting the system. Complementing this approach, the work of [9] introduces a more sophisticated version of the RLS algorithm, incorporating multiple forgetting factors that allow for individualized weighting for each model parameter. This innovation provides greater flexibility to the identification process, allowing the incorporation of prior knowledge of the designer and finer control over the adaptation of the algorithm to system variations. Together, these techniques

The authors from the Federal University of Pará thank the National Council for Scientific and Technological Development (CNPq), and the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES), for the financial support.

represent significant advances in developing robust control systems for unmanned aerial vehicles and operating phase sensing, especially in dynamic operational scenarios where parameters vary considerably during flight.

Recursive techniques are essential for detecting operational faults in UAVs. The work of [10] presents a state-of-the-art survey on the topic, highlighting that faults such as loss of motor efficiency, motor lock-up or total failure, unexpected thrust variations, and failures in servomechanisms can be effectively identified using these approaches. The application of such techniques enables the rapid detection and isolation of anomalies, allowing the control system to respond appropriately with safe maneuvers to prevent collisions or loss of the aircraft.

The main innovation of this work lies in the ability to capture the nonlinear dynamics of the angular behavior, quantifying its influence throughout the system's operation and minimizing the impact of unmodeled effects. This approach is associated with the use of RLS with multiple forgetting factors, which allows parameter weighting to be configured based on the designer's prior knowledge. Compared to the conventional implementation that uses a single forgetting factor, the proposed methodology demonstrates superior performance in parameter estimation, highlighting its greater flexibility and accuracy.

II. Methodology

This section is structured into three fundamental components. Initially, the architecture of the communication library responsible for the interface between the input signals and the quadrotor system is described, addressing its functional modules and transmission protocols.

Next, the discrete mathematical model representing the drone's dynamics is detailed, with emphasis on the formulation of the differential equations governing its physical behavior. This modeling serves as the basis for developing the parametric estimation algorithm.

The final stage introduces the estimation methodology based on instrumental variables, presenting its theoretical foundations and practical implementation for parameter identification. Particular attention is given to the criteria for selecting instruments and the statistical properties of the estimator.

A. Library

Created by [11], the PS-Drone library is a complete Application Programming Interface written in Python for Parrot's AR.Drone 2.0. The average sending time for the sensor information packet and flight commands is $T_s = 1/15$ seconds and could be faster. However, in the default configuration, the data is filtered by the quadcopter's own *firmware*. Therefore, the time for sending and receiving data is the sampling time of the quadcopter's discrete system. We can access most commands and data from this library, including sensor data (NavData) and flight commands.

The variables were used to build the model are the sensors (outputs): altitude (h) in meters, pitch (ϕ), roll (θ) and yaw (ψ) angles in radians, forward speed (v_x) and lateral speed (v_y) in meters per second, and yaw angular speed ($\dot{\psi}$) in radians per second. The inputs are and the commands are: forward thrust u_{vx}, lateral thrust u_{vy}, and altitude thrust u_h. It is important to note that these thrusts are dimensionless and limited to $[-1.0; 1.0]$ and that the nomenclature thrust is due to the library only accessing high-level input commands from the quadcopter. The firmware handles low-level commands since sending electric current commands to the motors with a delay of 1/15s of communication would be counterproductive. Therefore, the impulse command is an input that executes maneuvers already standardized by the *firmware*, allowing the user to control only the spatial movement of the drone (slower dynamics), and the firmware is responsible for controlling the mechatronic dynamics of the drone (faster dynamics).

The library stands out for having fundamental commands during the experiment, such as takeoff, landing, and hovering, which can be activated via the command prompt. Thus, the researcher can interrupt or start the experiment with practicality and safety.

B. Quadricopter Model

The four-engine aircraft has 6 degrees of freedom, which describes its movements in three-dimensional space. Of these, 3 degrees of freedom correspond to the angular movements around the three axes of the body coordinate system, known as roll angle, pitch angle, and yaw angle. The other 3 degrees of freedom refer to the linear movements along the three axes of the inertial coordinate system (X, Y, Z), which are forward/backward (X), left/right (Y), and up/down (Z). This coordinate system and the degrees of freedom are represented in figure 1.

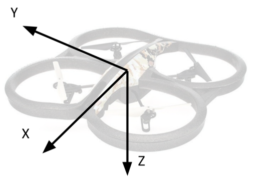

Fig. 1. NED Quadcopter.

The rotational movements of the quadcopter are responsible not only for rotation around the body axes but also for movement along the frontal and lateral axes. These movements are described by the Euler angles, which represent the orientations of the drone in three-dimensional space. The Equations (1), (2), and (3) describe respectively the rotational movements around the longitudinal, lateral, and vertical axes.

$$\ddot{\phi} = \frac{\dot{\theta}\dot{\psi}(I_{yy} - I_{zz})}{I_{xx}} + \frac{U_2}{I_{xx}} \tag{1}$$

$$\ddot{\theta} = \frac{\dot{\phi}\dot{\psi}(I_{zz} - I_{xx})}{I_{yy}} + \frac{U_3}{I_{yy}} \tag{2}$$

$$\ddot{\psi} = \frac{\dot{\phi}\dot{\theta}(I_{xx} - I_{yy})}{I_{zz}} + \frac{U_4}{I_{zz}} \tag{3}$$

The dynamic equations of the quadcopter describe its rotational motion around the three axes of the body coordinate system (x, y, z). These equations involve physical variables and parameters that represent the characteristics of the drone and its interactions with the environment. The variable ϕ (*roll*) represents the rotation angle around the x axis, θ (*pitch*) is the rotation angle around the y axis, and ψ (*yaw*) is the rotation angle around the z axis. The derivatives $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$ correspond to the angular velocities, while $\ddot{\phi}$, $\ddot{\theta}$ and $\ddot{\psi}$ represent the angular accelerations. The terms I_{xx}, I_{yy} and I_{zz} are the moments of inertia of the four-engine aircraft about the x, y and z axes, respectively. The control inputs U_2, U_3 and U_4 are the moments applied to control the *roll*, *pitch* and *yaw* movements.

The thrust command library generates the control moments, which establishes a linearized and decoupled relationship between the control inputs and the attitude angles, as implemented in the work of [12] for AR. Drone 2.0. In this work, we expand this linear modeling by incorporating the nonlinear dynamics of the system, resulting in 4

$$\begin{cases} \ddot{\theta} = K_\theta \omega_{\theta,\max}^2 u_\theta - 2\zeta_\theta \omega_\theta \dot{\theta} - \omega_\theta^2 \theta + \dfrac{\dot{\phi}\dot{\psi}(I_{zz} - I_{xx})}{I_{yy}}, \\ \ddot{\phi} = K_\phi \omega_{\phi,\max}^2 u_\phi - 2\zeta_\phi \omega_\phi \dot{\phi} - \omega_\phi^2 \phi + \dfrac{\dot{\theta}\dot{\psi}(I_{yy} - I_{zz})}{I_{xx}}, \\ \ddot{z} = \left(\dfrac{K_z \dot{z}_{\max}}{\tau_z}\right) u_z - \left(\dfrac{1}{\tau_z}\right) \dot{z}, \\ \ddot{\psi} = \left(\dfrac{K_\psi \dot{\psi}_{\max}}{\tau_\psi}\right) u_\psi - \left(\dfrac{1}{\tau_\psi}\right) \dot{\psi} + \dfrac{\dot{\phi}\dot{\theta}(I_{xx} - I_{yy})}{I_{zz}}. \end{cases} \tag{4}$$

where K_θ, K_ϕ, K_z and K_ψ are interpreted as the process gains, ω_θ and ω_ϕ are the natural frequencies, ζ_θ and ζ_ϕ are the damping coefficients and τ_z and τ_ψ are the time constants. Furthermore, ϕ_{max}, θ_{max}, $\dot{z}_{max}$ and $\dot{\psi}_{max}$ represent limiting parameters for the roll and pitch orientations, vertical speed and yaw, respectively.

The library's digital interface requires rewriting the equations in discrete time, with simplifications to make implementing the algorithms viable. The system input u, defined by the *thrust* command ($u[k-1]$), represents the thrust that generates angular torques and vertical accelerations (up/down), incorporating the delay of a sampling step (T_s) inherent to digital processing, as follows:

$$\ddot{x}(t) = \ddot{x}(k) \approx \frac{\dot{x}(k) - \dot{x}(k-1)}{T_s} \tag{5}$$

$$\dot{x}(k) \approx \frac{x(k) - x(k-1)}{T_s} \tag{6}$$

where x are the system states, t represents continuous time, k is the discrete-time sampling rate, and T_s is the sampling period (interval between discrete samples).

In the dynamic design of the quadcopter, after the temporal discretization of the state variables, an additional simplification is implemented by considering identical moments of inertia in the longitudinal and transverse axes. This assumption is physically justifiable by the radial symmetry in the mass distribution of the structure, as demonstrated by the relation presented in following relation:

$$\begin{aligned} I_x &\approx I_y = I, \\ A_c &= -I + I_z, \\ -A_c &= I - I_z. \end{aligned} \tag{7}$$

These considerations and the discretization of the system result in the following rotational equations:

$$\begin{aligned} \dot{\psi}[k] = \left(\frac{1}{T_s} + \frac{1}{\tau_\psi}\right) \Bigg(& B_\psi u_\psi[k-1] + \frac{\dot{\psi}[k-1]}{T_s} \\ & + \cancelto{0}{\frac{I - I}{I_y}} \dot{\phi}[k]\dot{\theta}[k] \Bigg) \end{aligned} \tag{8}$$

$$\begin{aligned} \dot{\theta}[k] = \frac{1}{\left(\frac{1}{T_s} + 2\zeta_\theta \omega_\theta\right)} \Bigg(& \frac{A_c \dot{\psi}[k]\dot{\phi}[k]}{I} + B_\theta u_\theta[k-1] \\ & + \frac{\dot{\theta}[k-1]}{T_s} - \omega_\theta^2 \theta[k] \Bigg) \end{aligned} \tag{9}$$

$$\begin{aligned} \dot{\phi}[k] = \frac{1}{\left(\frac{1}{T_s} + 2\zeta_\phi \omega_\phi\right)} \Bigg(& - \frac{A_c \dot{\psi}[k]\dot{\theta}[k]}{I} + B_\phi u_\phi[k-1] \\ & + \frac{\dot{\phi}[k-1]}{T_s} + \omega_\phi^2 \phi[k] \Bigg) \end{aligned} \tag{10}$$

where B are the scalars that multiply the system inputs.

With this set of equations, it is necessary to rewrite them based on the quadcopter angles and the angular couplings since only the yaw angle has a direct reading of angular velocity. Thus, the discretized models of the dynamics of each angle, taking into account the cross-couplings mediated by $\dot{\psi}$, are presented in the equations (11), (12), (13) and (14).

$$\begin{aligned} \theta[k] = \frac{1}{1 + \Gamma[k]} \Big(& a_{1_\theta} \theta[k-1] + a_{2_\theta} \theta[k-2] + b_\theta u_\theta[k-1] \\ & + \dot{\psi} \big(a_{1_{\theta\phi}} \phi[k-1] + a_{2_{\theta\phi}} \phi[k-2] + b_{\theta\phi} u_\phi[k-1] \big) \Big) \end{aligned} \tag{11}$$

$$\begin{aligned} \phi[k] = \frac{1}{1 + \Gamma[k]} \Big(& a_{1_\phi} \phi[k-1] + a_{2_\phi} \phi[k-2] + b_\phi u_\phi[k-1] \\ & + \dot{\psi} \big(a_{1_{\phi\theta}} \theta[k-1] + a_{2_{\phi\theta}} \theta[k-2] + b_{\phi\theta} u_\theta[k-1] \big) \Big) \end{aligned} \tag{12}$$

$$\dot{\psi}[k] = a_{1_{\dot{\psi}}} \, \dot{\psi}[k-1] + b_{\dot{\psi}} \, u_\psi[k-1] \tag{13}$$

$$\psi[k] = a_{1_\psi} \, \psi[k-1] + a_{2_\psi} \, \psi[k-2] + b_\psi \, u_\psi[k-1] \tag{14}$$

The factor $\Gamma[k]$ is represented in the (15). Γ quantifies the dynamic coupling effect between the pitch and roll axes as a function of the yaw rate $\dot{\psi}[k]$, making the system slower as the yaw rate norm increases.

$$\Gamma[k] = \frac{(I - I_{zz})^2 \, \dot{\psi}[k]^2}{I^2 \left(\frac{1}{T_s} + 2\zeta_\theta \omega_\theta\right) \left(\frac{1}{T_s} + 2\zeta_\phi \omega_\phi\right)} \tag{15}$$

The nonlinear discretized MIMO model of the drone's rotational dynamics results in a bilinear system [13], which underlies the simplifications adopted in previously cited works. In experiments performed with slight variations in the yaw angle, it is observed that the angular coupling between the states can be neglected. In this work, the values of $\Gamma[k]$ are sufficiently small for its influence on the dynamics to be considered negligible. This simplification is physically justified by the inequality $(I - I_{zz})^2 \ll I^2 \left(\frac{1}{T_s} + 2\zeta_\theta\omega_\theta\right)\left(\frac{1}{T_s} + 2\zeta_\phi\omega_\phi\right)$, indicating that only significant variations in the yaw angle would be able to alter the dynamic behavior of the system significantly.

From the dynamic equations of the angles, the translational motion model of the drone is developed. The displacements in the frontal and lateral axes result from the torques generated by the motors. Considering that the low-level control ensures the generation of these torques without compromising the altitude of the vehicle, the dynamic equations that describe these accelerations fall into Equations (16) and (17).

$$a_y = g\tan\phi - \frac{D}{m}v_y \tag{16}$$

$$a_x = -g\tan\theta - \frac{D}{m}v_x \tag{17}$$

where the acceleration is a, g is the gravitational acceleration, D the friction coefficient, m the mass of the drone and v the velocity. For variations of up to 20 degrees, the approximation $\tan\theta \approx \theta$ (valid by the Taylor series expansion, neglecting nonlinear terms) simplifies the to $a_x \approx g\theta - \frac{D}{m}v_x$, revealing a direct proportionality between the acceleration and the angle, and thus balancing precision and simplicity. Rewriting the system in terms of velocity, we have Equations (18) and (19), which describe the discrete evolution of $v_y[k]$ and $v_x[k]$, respectively.

$$v_y[k] = a_{vy_\theta}\,\theta[k] + a_{vy}\,v_y[k-1] \tag{18}$$

$$v_x[k] = a_{vx_\phi}\,\phi[k] + a_{vx}\,v_x[k-1] \tag{19}$$

Finally, the altitude grid is discretized, generating the Equation(20) model.

$$h[k] = a_{1_h}\,h[k-1] + a_{2_h}\,h[k-2] + b_h\,u_h[k-1] \tag{20}$$

Once the models are in hand, parametric estimation methods can be established, considering the temporal parametric change during the quadcopter flight.

C. Recursive Least Squares

The application of real-time identification algorithms has several purposes, such as supervision, tracking of varying parameters for adaptive control, filtering, prediction, signal processing, detection and diagnosis [14]. In this context, the Recursive Least Squares (RLS) Method provides reliable bases for these applications, consolidating itself as one of the industry's most widely used techniques for estimating parameters in real-time.

The parametric estimates made sample by sample of the model are made employing a gain vector L, whose values are calculated according (21), where P is the parametric covariance matrix, Φ the values of the functions that control the system at that moment and λ is the forgetting factor.

$$L(k) = \frac{P(k-1)\Phi(k)}{\lambda + \Phi^T(k)P(k-1)\Phi(k)} \tag{21}$$

The estimated parameters are updated according to (22).

$$\beta_{est}(k) = \beta_{est}(k-1) + L(k)e(k) \tag{22}$$

where β_{est} are the estimated parameters, β are the real parameters and $e(k)$ is the estimation error.

The gains of the matrix L are responsible for updating the covariance matrix P at each iteration of the loop according to (23) where m is the number of parameters in the model. This operation progressively reduces their values until convergence to stable minima.

$$P(k) = \frac{\left[I_{m\times m} - L(k)\Phi^T(k)\right]P(k-1)}{\lambda} \tag{23}$$

Finally, the quadratic error in each interaction is minimized according to (24).

$$J(k) = \left(y(k) - \Phi(k)\beta_{est}(k-1)\right)^2 \tag{24}$$

In the specialized literature, empirical studies and consolidated theoretical analyses in the identification of systems [15] establish that the forgetting factor λ should preferably be in the interval $\lambda \in [0.99, 1.00)$ to simultaneously guarantee numerical stability, maintaining convergence properties and avoiding parametric divergence; asymptotic convergence, ensuring that $\beta_{est}(k) \to \beta(k)$ when $k \to \infty$; and robustness to noise, preserving the ability to filter high-frequency disturbances without compromising parameter tracking.

D. Extend Recursive Least Squares

The standard RLS algorithm exclusively calculates the system parameters; however, following the approach proposed by [2]., it becomes essential to incorporate the estimation of external disturbances and their dynamic propagation in the aerial vehicle. To this end, an additional regressor portion is added to the observation vector present in Equations (25) and (26), composed of the estimation errors of the outputs generated by the Extended Recursive Least Squares (ERLS) algorithm. In this work, the RLS is extended to the frontal velocity (v_x), lateral velocity (v_y), yaw angle (ψ), and altitude (h) meshes, allowing the dynamics that disturb the system to be included, in the algorithm.

$$e(k) = y(k) - y_{\text{est}}(k) \tag{25}$$

$$\Phi_{\text{ext}}(k) = \begin{bmatrix}\Phi(k)^T & e(k-1)\end{bmatrix}^T \tag{26}$$

E. Recursive Least Squares with Multiple Forgetting Factors

In the application of ERLS, the parameters will be weighted equally. However, in the work An Extended U-D Algorithm with Multiple Forgetting Factors for RLS Estimation of Model Parameters, by G. Hardier [16], another approach to RLS is presented. This approach introduces Multiple Forgetting Factors (MFFs), allowing each model parameter to have an update rate appropriate to its temporal change. This approach is particularly relevant in aerospace applications, such as airplanes and unmanned aerial vehicles (UAVs), where different parameters can vary at different scales. Using a single factor can compromise the accuracy or stability of the algorithm.

The formulation allows tracking rapid variations in some parameters while preserving robustness against noise in other, more stable ones. The article demonstrates, through simulations with realistic academic benchmarks, that using MFFs improves the estimation performance compared to traditional versions of RLS with a single factor. This technique is applicable in embedded systems for diagnostics and fault tolerance (FDD/FTC), offering better parameter tracking in degraded operating conditions, such as actuator failures, ice formation or load asymmetries.

For multiple factors $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m)$, the update of P_k is described in (27).

$$P_{k+1} = \Lambda^{-1/2} \left[P_k - \frac{P_k \Phi_k \Phi_k^\top P_k}{1 + \Phi_k^\top P_k \Phi_k} \right] \Lambda^{-1/2} \tag{27}$$

where $\Lambda^{-1/2} = \text{diag}(1/\sqrt{\lambda_1}, \ldots, 1/\sqrt{\lambda_m})$.

The vector $\beta_{est}[k]$ update is maintained. Thus, if all parameters are in the exact weighting, all values of λ_i are equal, the algorithm falls back to traditional RLS.In this work, we apply the MFF RLS in its extended version (MFF-ERLS), modeling both the internal dynamics of the quadrotor and its external dynamics.

The MFF-ERLS method allows the designer to assign different update priorities to each parameter, thereby incorporating prior knowledge of the plant's behavior into the parameter identification process. The following pseudocodes, which represent the three estimation approaches (with $\lambda = 1$, $\lambda < 1$, and multiple forgetting factors), share the same structural framework and differ only in the way parameter weights are configured.

Algorithm 1 ERLS with Configurable Forgetting Weights

```
Input: Regressor vector $\Phi_k$, output $y[k]$
Parameters:
    - $\lambda = 1$: no forgetting
    - $\lambda < 1$: single forgetting factor (uniform forgetting for all parameters)
    - $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_m)$: multiple forgetting factors (individual weight per parameter)
Initialization: $\beta_{\text{est}}[0]$, $P[0]$
for $k = 1$ to $N$ do
    if using multiple forgetting factors (MFF) then
        $W \leftarrow \Lambda^{-1/2}$        // Individual parameter weighting matrix
    else
        $W \leftarrow \sqrt{\frac{1}{\lambda}} \cdot I_m$        // Uniform weight applied to all parameters
    end if
    $K[k] \leftarrow \frac{P[k-1]\Phi_k}{1 + \Phi_k^T P[k-1]\Phi_k}$
    $\beta_{\text{est}}[k] \leftarrow \beta_{\text{est}}[k-1] + K[k]\left(y[k] - \Phi_k^T \beta_{\text{est}}[k-1]\right)$
    $P[k] \leftarrow W\left(P[k-1] - K[k]\Phi_k^T P[k-1]\right) W$
end for
```

III. Experiments

The experimental testing environment for identifying the AR Drone 2.0 model must be large enough to make collisions more difficult to occur, with good lighting (since the drone's translational speeds are estimated via pixel flow using a camera located on the bottom of the drone, pointing at the ground) and isolated from external interference (indoor) such as wind, for example.

In the experiment, inputs with limited amplitude in the range of $[-0{,}2,\ 0{,}2]$ were applied in order to excite the dynamics of the quadcopter in a controlled manner. This limitation was imposed to avoid inducing high speeds or unstable behaviors that could lead to loss of control of the UAV. Figure 2 shows the sequence of inputs used.

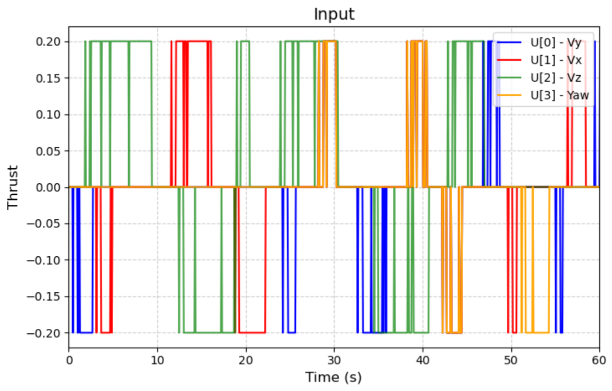

Fig. 2. Inputs.

The dynamic system of the quadcopter, subject to input commands, has its states monitored in real-time, allowing the construction of the vector $\Phi(k)$ used in the parametric identification methods used in this work. Three configurations were implemented: Classical ERLS with forgetting factor $\lambda = 1$, uniformly distributing the weight of the observations; Adaptive ERLS ($0 < \lambda < 1$), prioritizing recent data through exponential decay; and an approach with multiple forgetting factors, where time-varying parameters ($\lambda_i = \lambda < 1$) coexist with static parameters ($\lambda_i = 1$). This strategy allows the simultaneous estimation of the transient dynamics and the stationary characteristics of the system.

All algorithms were initialized with $P_0 = 10^4 \mathbf{I}_m$ (initial covariance matrix) because the high magnitude accelerates the initial convergence, and the initial parameters were null, ensuring identical conditions for fair comparison between the methods.

Model validation was performed using the Normalized Root Mean Squared Error (NRMSE) as the performance metric. This normalized measure, based on the mean squared error, offers the advantage of being independent of the data scale and experiment length. Its interpretation is intuitive: values close to 1 indicate that the estimated model behaves the same way as the real system while decreasing values reflect a loss of precision. Normalization by the amplitude of the real data allows absolute comparisons between different experimental scenarios, making it particularly suitable for evaluating dynamic systems. The Equation (28) demonstrates the calculation of the index.

$$J_{\text{NRMSE}} = 1 - \frac{\sum_{k=1}^{N} (y[k] - y_{est}[k])^2}{\sum_{k=1}^{N} (y[k] - \bar{y}[k])^2} \tag{28}$$

IV. Results

After the experimental flight, the forgetting factor of 0.999 was determined by trial and error, within the recommended range. With this value established, the MFF-ERLS was applied with selective adjustment in the forgetting factors related to the parameters of the quadcopter inputs. This adaptation is justified by the temporal dynamics of the actuators since, during the flight, degradation in the performance of the motors is observed due to the heating, progressive reduction of rotation, and energy limitations of the battery, such factors directly impacting the execution of the pre-programmed maneuvers. The value of the forgetting factor in these parameters was 0.9999. In contrast, the parameters related to the physical states of the system maintained a forgetting factor equal to 1 since the fundamental quantities, such as air density, gravity, inertia, and mass, remain constant in the established experimental conditions.

The tested algorithms have their parameters consolidated over the experiment time, decreasing the amplitude of the covariance matrix values. Thus converging to values of minor variation. Figure 3 shows the trace of the covariance matrices throughout the experiment, with the algorithms converging after 40 seconds of the experiment.

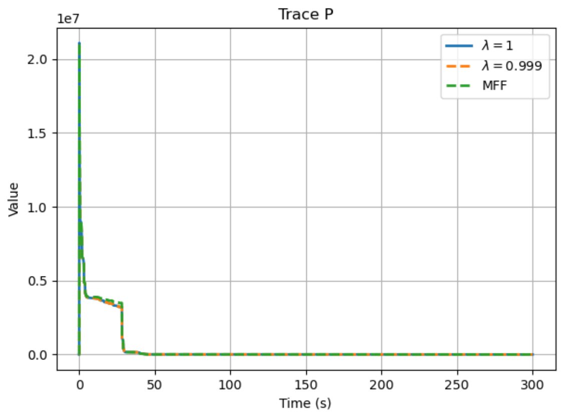

Fig. 3. The trace of the covariance matrix P over time.

With the actual output and the outputs estimated by the models collected, the NRMSE values calculated for each state of the air system from the algorithm's convergence point (from 40 s until the end of the experiment), as shown in Table 1.

TABLE I
NRMSE

States	ERLS ($\lambda = 1$)	ERLS ($\lambda = 0.999$)	MFF-ERLS
θ	0.9039	0.904	0.904
ϕ	0.9003	0.9004	0.9003
v_x	0.898	0.8983	0.8981
v_y	0.8919	0.8925	0.8923
ψ	0.9924	0.9934	0.9937
$\dot{\psi}$	0.818	0.8183	0.8181
h	0.9841	0.9843	0.9843

The comparative analysis confirms that the estimated models accurately reproduce (NRMSE ≥ 0.8) the real dynamics of the UAV in all states, according to standards established in the literature [14]. This correspondence qualifies the models for applications in model-based control, particularly in projects that require dynamic accuracy and predictability of system response. Figure 4 and Figure 5 demonstrate how the models and the real output behave similarly.

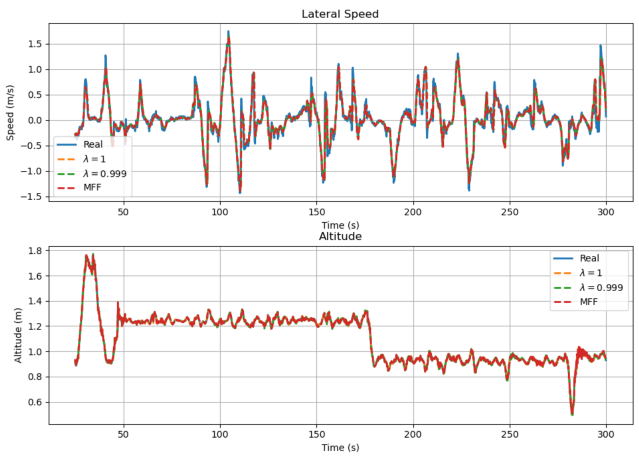

Fig. 4. Lateral Speed and Altitude.

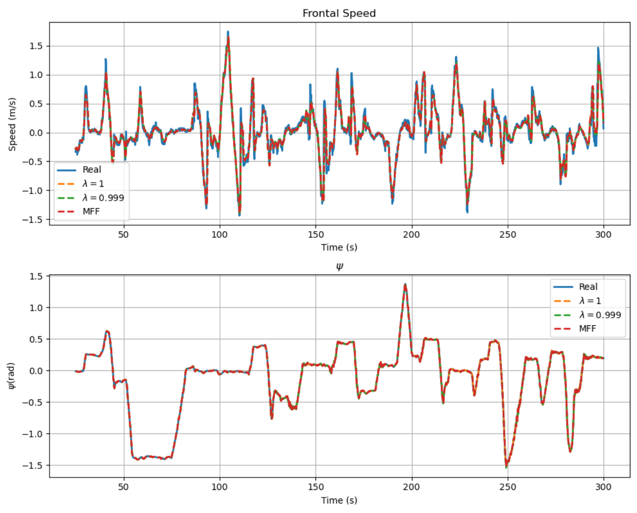

Fig. 5. Frontal Speed and ψ Angle.

A significant altitude loss was detected between $270s$ and $300s$, resulting from insufficient battery charge. This operational fault led to the premature termination of the flight, highlighting the importance of real-time energy diagnostics to ensure UAV reliability and mission success. This scenario reinforces the relevance of using recursive estimation techniques, which evaluate data sample by sample, enabling faster fault detection and real-time adaptation during flight.

Although the state estimation performance of the quadcopter appears similar across the different configurations, this consistency does not extend to the estimation of the system parameters. The particular characteristics and internal mechanisms of each identification algorithm significantly affect how the parameters evolve over time during the experiments. As shown in Figure 6, the MFF-ERLS algorithm demonstrates a more aggressive behavior in the estimation of parameters b_h and b_ψ, rapidly adapting to changes and exhibiting greater sensitivity. On the other hand, when estimating the remaining parameters, the algorithm adopts a more conservative stance, resulting in smoother and more gradual adjustments, as illustrated in Figure 7. This contrast highlights the influence of the algorithm's design on the parameter dynamics and its potential to selectively respond to variations in the system.

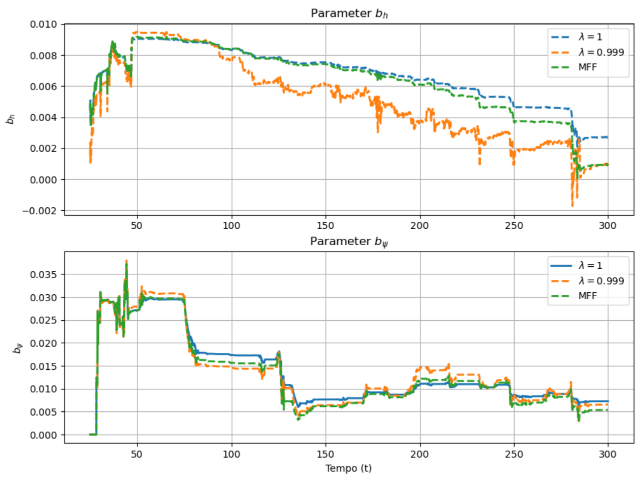

Fig. 6. Parameters b_h and b_ψ.

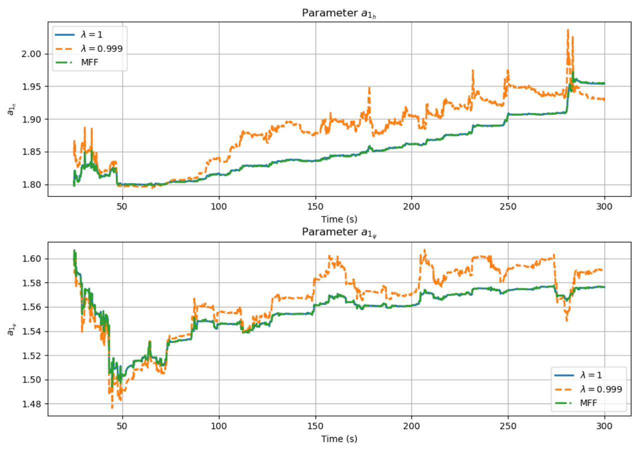

Fig. 7. Parameters a_{h_1} and a_{ψ_1}.

The parameters estimated by the ERLS algorithm with a forgetting factor equal to 1 converge more conservatively, reflecting greater inertia in the face of new observations. In contrast, the ERLS with a forgetting factor $\lambda = 0.999$ presents a more aggressive convergence, attributing greater weight to recent observations and distributing the estimation error evenly among all parameters.

The MFF-ERLS algorithm, in turn, adopts a hybrid behavior: its response is similar to that of the ERLS with $\lambda = 1$ in the parameters a_{1_h} and a_{1_ψ} (associated with the internal dynamics of the states), presenting a more conservative convergence; on the other hand, it shows a more responsive behavior in the parameters b_h and b_ψ (related directly to the system inputs).

This difference in behavior is due to the estimation error update mechanism in the MFF-ERLS, which penalizes the parameters associated with the inputs more heavily when there is a greater discrepancy between the model and the measurement, assigning greater responsibility for the prediction error. As a result, the parameters b_h and b_ψ undergo more intense adjustments, while the state coefficients a_{1_h} and a_{1_ψ} are adjusted more cautiously.

This behavior allows algorithms to incorporate prior knowledge about the priority in which parameters should be adjusted. This is especially useful in complex systems, such as the quadcopter, which involve many parameters resulting from linear and nonlinear relationships. This avoids unnecessary updating of parameters not directly related to the measurement error, reducing the variability in the estimates throughout the experiment. Consequently, there is less variation in the parameters estimated by MFF-ERLS than in the other algorithms, as shown in Table 2. Where e_{vy}, e_{vx}, e_ψ, and e_h are the parameters associated with the regressors of the lateral velocity error, forward velocity error, yaw angle error, and altitude error, respectively.

TABLE II
VARIANCE OF PARAMETER ESTIMATION

Parameter	**ERLS ($\lambda = 1$)**	**ERLS ($\lambda = 0.999$)**	**MFF-ERLS**
a_{1_θ}	0.000390	0.000796	0.000389
a_{2_θ}	0.000331	0.000680	0.000329
b_θ	2.45×10^{-6}	3.52×10^{-6}	2.50×10^{-6}
$b_{\theta\phi}$	0.006460	0.006248	0.006439
$a_{1_{\theta\phi}}$	0.061977	0.076781	0.061800
$a_{2_{\theta\phi}}$	0.073598	0.093901	0.073425
a_{1_ϕ}	0.000258	0.000664	0.000257
a_{2_ϕ}	0.000325	0.000747	0.000325
b_ϕ	2.39×10^{-6}	5.33×10^{-6}	2.61×10^{-6}
$b_{\phi\theta}$	0.009248	0.010057	0.009318
$a_{1_{\phi\theta}}$	0.071790	0.101851	0.072603
$a_{2_{\phi\theta}}$	0.133698	0.161202	0.134966
a_{vy_θ}	0.002310	0.011939	0.002310
a_{vy}	8.87×10^{-6}	3.03×10^{-5}	8.87×10^{-6}
e_{vy}	0.001029	0.002522	0.001029
a_{vx_ϕ}	0.001177	0.003437	0.001177
a_{vx}	7.87×10^{-6}	1.10×10^{-5}	7.87×10^{-6}
e_{vx}	0.002700	0.005778	0.002700
a_{1_ψ}	0.010118	0.009660	0.009960
a_{2_ψ}	0.010088	0.009651	0.009934
b_ψ	5.90×10^{-5}	6.34×10^{-5}	6.49×10^{-5}
e_ψ	0.005027	0.005764	0.005210
$a_{1_{\dot\psi}}$	0.011329	0.014505	0.011255
$b_{\dot\psi}$	0.018704	0.024780	0.019677
a_{1_h}	0.001869	0.002408	0.001902
a_{2_h}	0.001869	0.002408	0.001901
b_ψ	3.06×10^{-6}	6.81×10^{-6}	4.95×10^{-6}
e_h	0.000514	0.000938	0.000530
Total	**0.4143**	**0.5376**	**0.4128**

Note that the ERLS algorithm with a forgetting factor equal to 0.999 makes more aggressive adjustments to the parameters to improve the estimate, responding quickly to variations in the data. In contrast, ERLS with a factor equal to 1 presents a slower response to new information, but the uniform weighting makes any change in the data affect all parameters equally. Finally, MFF-ERLS adapts in a more balanced way to parametric changes, as it uses different forgetting factors, allowing a more selective adjustment consistent with the designer's prior knowledge, resulting in less parametric variation.

This aspect is fundamental for developing model-based adaptive control strategies, such as Linear Quadratic Regulator, Linear Quadratic Gaussian and Model Predictive Control, making them more robust and intelligent. In addition, the online identification of the parameters contributes to the early detection of system failures. An example of this can be observed in the interval between $270s$ and $300s$ when a low battery charge causes a loss of altitude. This operational failure is more evident in the estimation performed by the MFF-ERLS algorithm, demonstrating its greater sensitivity to dynamic changes and real-time diagnostic capacity.

V. Conclusion

This work employs the Extended MFF-ERLS algorithm to identify the nonlinear dynamics of the Parrot AR.Drone 2.0 quadcopter. The algorithm was tested in real-world experiments, and its performance and behavior were compared with those of various Extended RLS configurations. Although the NRMSE index showed similar performance across configurations, the Extended MFF-ERLS approach offered a significant advantage: it allowed for the integration of prior knowledge about the system's behavior into the algorithm design. This enabled more effective detection of both operational and parametric changes in the system.

Such results are possible because, while the internal state dynamics of the quadcopter remain relatively stable, the relationship between the inputs and the states varies due to factors such as motor heating and battery discharge. The MFF-ERLS algorithm accounts for this by enabling different sensitivity weights to be assigned to each parameter, enhancing its adaptability to the system's time-varying characteristics.

For future work, it is recommended to explore more advanced strategies for adaptively tuning the weights of each parameter, such as decision-making models. These approaches would allow the configuration to range from robust adaptive control to the detection and prevention of operational failures. Additionally, employing the nonlinear drone model within the Unscented Kalman Filter framework would enable joint estimation of both states and parameters, providing a more comprehensive and flexible adaptation to the system's dynamic behavior.

References

[1] D. W. Casbeer, D. B. Kingston, R. W. Beard, and T. W. M. and, "Cooperative forest fire surveillance using a team of small unmanned air vehicles," *International Journal of Systems Science*, vol. 37, no. 6, pp. 351–360, 2006.

[2] A. Silveira, A. Silva, A. Coelho, J. Real, and O. Silva, "Design and real-time implementation of a wireless autopilot using multivariable predictive generalized minimum variance control in the state-space," *Aerospace Science and Technology*, vol. 105, p. 106053, 2020.

[3] L. V. Santana, A. S. Brandão, and M. Sarcinelli-Filho, "Navigation and cooperative control using the AR.Drone quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 84, pp. 327–350, 2016.

[4] V. P. Tran, F. Santoso, and M. A. Garratt, "Adaptive trajectory tracking for quadrotor systems in unknown wind environments using particle swarm optimization-based strictly negative imaginary controllers," *IEEE transactions on aerospace and electronic systems*, vol. 57, no. 3, pp. 1742–1752, 2021.

[5] A. J. Abougarair, H. Almgallesh, and N. A. A. Shashoa, "Dynamics and optimal control of quadcopter," in *2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*. IEEE, 2024, pp. 136–141.

[6] A. S. Elkhatem and S. N. Engin, "Robust LQR and LQR-PI control strategies based on adaptive weighting matrix selection for a UAV position and attitude tracking control," *Alexandria Engineering Journal*, vol. 61, no. 8, pp. 6275–6292, 2022.

[7] Z. Cai, S. Zhang, and X. Jing, "Model predictive controller for quadcopter trajectory tracking based on feedback linearization," *IEEE Access*, vol. 9, pp. 162 909–162 918, 2021.

[8] D. Benjumea, A. Alcántara, A. Ramos, A. Torres-Gonzalez, P. Sánchez-Cuevas, J. Capitan, G. Heredia, and A. Ollero, "Localization system for lightweight unmanned aerial vehicles in inspection tasks," *Sensors*, vol. 21, no. 17, p. 5937, 2021.

[9] W. T. Chor, C. P. Tan, A. Bakibillah, Z. Pu, and J. Y. Loo, "Robust vehicle mass estimation using recursive least m-squares algorithm for intelligent vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 165–177, 2023.

[10] R. Puchalski and W. Giernacki, "Uav fault detection methods, state-of-the-art," *Drones*, vol. 6, no. 11, p. 330, 2022.

[11] J. Graff, "The ps-drone-api: Programming a parrot AR.Drone 2.0 with python-the easy way," 2014.

[12] R. S. Leonello, M. Z. Seixas, H. X. de Araújo, and B. Ordonez, "Ar quadrotor trajectory control. Drone 2.0 via LQR and H_∞ approaches. controle de trajetória do quadrotor AR.Drone 2.0 via abordagens lqr e h_infinito," in *Simpósio Brasileiro de Automação Inteligente-SBAI*, vol. 1, no. 1, 2021.

[13] F. Vicario, "Okid as a general approach to linear and bilinear system identification," Ph.D. dissertation, COLUMBIA UNIVERSITY, 2014.

[14] A. A. R. Coelho and L. dos Santos Coelho, *Identificação de sistemas dinâmicos lineares*, 2015.

[15] Karl Johan Åström and Björn Wittenmark, *Adaptive Control*, 2nd ed. Dover Publications, 2008.

[16] G. Hardier, "An extended U-D algorithm with multiple forgetting factors for RLS estimation of model parameters," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 200–207, 2015.