Nonlinear Predictive Functional Control With Hammerstein Model and Application to Speed Control for Multicopter Rotors*

Alexander Schulzke[1] and Joachim Horn[1] and Mirco Alpen[1]

***Abstract*— This paper presents a control framework for rotor angular velocity regulation in multicopter systems operating within the PX4 autopilot environment, where direct rotor-speed commands are not natively supported. To bridge this gap, the propulsion channel is modelled as a nonlinear, input-constrained, delayed single-input single-output system using a Hammerstein structure. Based on this representation, a nonlinear predictive functional control (NPFC) scheme is developed, combining a linear predictive functional controller with numerical inversion of the static nonlinearity. A simplified linear PFC formulation is employed to reduce computational complexity, while real-time feasibility is maintained through efficient inversion methods, including a golden-section search and a look-up table approximation. Input-to-state stability of the closed-loop system is formally established using Lyapunov analysis. The proposed controller is experimentally validated on a physical multicopter platform, demonstrating stable and accurate performance in both set-point regulation and trajectory tracking tasks. Results show that the look-up table approach achieves comparable accuracy to optimisation-based inversion with significantly reduced computational effort. Furthermore, the controller exhibits robustness against model–plant mismatch and maintains satisfactory performance under system delays.**

I. INTRODUCTION

Model-based control of multicopter systems typically yields desired rotor angular velocities ω as control inputs for stabilisation and trajectory tracking [1], [2], [3]. In practice, however, widely used flight stacks such as PX4 [4] do not support direct rotor-speed commands, but instead require normalised actuator inputs. This necessitates an additional control layer that maps desired angular velocities to admissible inputs while accounting for actuator dynamics.

The resulting propulsion subsystem exhibits nonlinear, input-constrained, and delayed behaviour and is therefore in this work treated as a single-input single-output (SISO) black-box system. Classical approaches, such as PID control, typically neglect delays or compensate for them heuristically, which can limit performance. While Hammerstein models provide an effective representation of such systems by combining static input nonlinearities with linear dynamics [5], [6], their use in predictive control has primarily been explored within nonlinear model predictive control (NMPC) frameworks [7], [8], [9].

*This research made as part of the project *Digitized, legally compliant and low-emission airmobile inspection and network data collection with automated drones* (DNeD) is funded by dtec.bw - Digitalization and Technology Research Center of the Bundeswehr which we gratefully acknowledge. dtec.bw is funded by the European Union – NextGenerationEU.

[1]All Authors are with Faculty of Control Theory, Helmut Schmidt University/ University of the Federal Armed Forces Hamburg, 22043 Hamburg, Germany schulzka@hsu-hh.de

In contrast, the integration of Hammerstein models with Predictive Functional Control (PFC) [10] remains limited. A notable example is [11], which applies a Hammerstein-based nonlinear PFC scheme, but lacks rigorous stability guarantees. Moreover, predictive control approaches for rotor-speed regulation under realistic constraints—such as actuator saturation and delays—remain largely unexplored.

To address these gaps, this paper proposes a Hammerstein-based nonlinear PFC framework for rotor-speed control in PX4-based UAV systems. The approach explicitly accounts for nonlinearities, input constraints, and delays, while maintaining low computational complexity. A Lyapunov-based analysis establishes input-to-state stability of the closed-loop system. The method is validated experimentally on a multicopter platform, demonstrating accurate tracking, robustness to model mismatch, and practical feasibility for embedded implementation.

II. CONTROLLER DEVELOPMENT

A. Predictive Functional Control

A key advantage of PFC is its natural handling of system delays: delayed measurements are compared directly to the corresponding delayed model outputs, providing effective delay compensation. This property, coupled with its low computational cost, makes it a suitable choice for this study. A remaining challenge, however, is that the plant under consideration exhibits nonlinear behaviour, whereas the employed PFC formulation is inherently linear.

This section presents a reduced formulation of linear PFC, consistent with the implementation used in this study. A detailed treatment of PFC can be found in [10].

PFC is a model-based control strategy in which the disturbance or model–plant mismatch is defined as

$$d(t) = y_{\mathrm{p}}(t) - y_{\mathrm{m}}(t), \tag{1}$$

where $y_{\mathrm{p}}(t)$ denotes the measured plant output and $Y_{\mathrm{m}}(s) = \mathcal{L}\{y_{\mathrm{m}}(t)\} = G_{\mathrm{m}}(s)U(s)$ the corresponding model prediction, with the system model $G_{\mathrm{m}}(s)$ and control input $U(s)$. At steady state, the dynamics are reduced to

$$Y_{\mathrm{m,ss}} = G_{\mathrm{m,ss}} U_{\mathrm{ss}}. \tag{2}$$

Using the static gain $k_0 = G_{\mathrm{m,ss}}$ and $u_{\mathrm{ss}} = U_{\mathrm{ss}}$,

$$Y_{\mathrm{m,ss}} = k_0 u_{\mathrm{ss}}. \tag{3}$$

Thus, for the process output, with $D(s) = \mathcal{L}\{d(t)\}$,

$$Y_{\mathrm{p,ss}} = k_0 u_{\mathrm{ss}} + D_{\mathrm{ss}} = k_0 u_{\mathrm{ss}} + d_{\mathrm{ss}}. \tag{4}$$

The controller objective is to track a constant reference r, which yields

$$r \overset{!}{=} Y_{\text{p,ss}} = k_0 u_{\text{ss}} + d_{\text{ss}} \tag{5}$$

$$u_{\text{ss}} = k_0^{-1}(r - d_{\text{ss}}). \tag{6}$$

Since d_{ss} is unknown but $y_{\text{p}}(t)$ and $y_{\text{m}}(t)$ are measurable, the control law becomes

$$u(t) = k_0^{-1}(r - (y_{\text{p}}(t) - y_{\text{m}}(t))). \tag{7}$$

B. Application to the Hammerstein model

The PFC framework can be extended to systems represented by Hammerstein models in order to address nonlinear applications. A Hammerstein model consists of a static nonlinear block followed by a linear dynamic subsystem [5]. This structure offers notable advantages, as the control of the linear subsystem is well established in the literature. Consequently, only the incorporation of the static nonlinearity must be addressed, which is comparatively straightforward due to its memoryless nature. Its general representation in the Laplace space is given by

$$Y(s) = G(s)\mathcal{L}\{f(u(t))\} \tag{8}$$

where $G(s)$ describes an arbitrary linear system, and $f(u(t))$ is a static nonlinear mapping of input $u(t)$. Hammerstein models are widely used because they retain sufficient nonlinear representation while preserving linear dynamic structure, which facilitates system identification and controller design. As shown earlier, a central PFC operation is the inversion of the static mapping. This extends directly to (8), yielding

$$Y_{\text{m}}(s) = G_{\text{m}}(s)\mathcal{L}\{f_{\text{m}}(u(t))\}, \tag{9}$$

where $G_{\text{m}}(s)$ is the linear subsystem model and $f_{\text{m}}(\cdot)$ is the nonlinear static model. Because the linear portion can be reduced to pure dynamics,

$$G_{\text{m,ss}} = k_0 = 1. \tag{10}$$

Using (1)-(7) the steady-state control law becomes

$$Y_{\text{m,ss}} = G_{\text{m,ss}}\mathcal{L}\{f_{\text{m}}(u_{\text{ss}}\} = f_{\text{m}}(u_{\text{ss}}) \tag{11}$$

$$r \overset{!}{=} Y_{\text{p,ss}} = Y_{\text{m,ss}} + D_{\text{ss}} \tag{12}$$

$$= f_{\text{m}}(u_{\text{ss}}) + d_{\text{ss}} \tag{13}$$

$$u_{\text{ss}} = f_{\text{m}}^{-1}(r - d_{\text{ss}}). \tag{14}$$

This makes sense logically, as the control input at steady state is the reference inverted through system dynamics and corrected by the disturbance. Since d_{ss} is unknown but $y_{\text{p}}(t)$ and $y_{\text{m}}(t)$ are measurable, the control law becomes

$$u(t) = f_{\text{m}}^{-1}(r - (y_{\text{p}}(t) - y_{\text{m}}(t))). \tag{15}$$

Equation (15) highlights a practical difficulty: analytical inversion of the nonlinear mapping $f_{\text{m}}(\cdot)$ is generally not available. In this study, $f_{\text{m}}(u(t))$ is strictly monotonically increasing on $u \in [0.1, 1)$, allowing the inverse to be computed numerically via

$$\min_{u} J = (l - f_{\text{m}}(u))^2, \tag{16}$$

where J denotes the Cost function and l the function output to be inverted, which has a unique global minimiser u^* satisfying $f_{\text{m}}(u^*) = l$. This resolves the nonlinear inversion at the cost of additional computation, though efficient numerical solvers maintain real-time feasibility.

C. Stability Analysis

If a Hammerstein model, like (8), can describe a system, under the assumptions that

1) the linear part is stable and controllable,
2) the linear part's static gain is 1,
3) the nonlinearity is locally Lipschitz,
4) the input is constrained on $u \in (0, w]$.

The delayed system can be written in controllable canonical form as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}f(u(t - T_d)) \tag{17}$$

$$y_{\text{p}}(t) = \mathbf{C}\mathbf{x}(t) \tag{18}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \\ -\frac{a_0}{a_{\text{n}}} & -\frac{a_1}{a_{\text{n}}} & -\frac{a_2}{a_{\text{n}}} & \cdots & -\frac{a_{\text{n}}-1}{a_{\text{n}}} \end{bmatrix} \tag{19}$$

$$\mathbf{B} = \begin{bmatrix} 0 & \cdots & 0 & \frac{1}{a_{\text{n}}} \end{bmatrix}^\top, \qquad \mathbf{C} = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}, \tag{20}$$

where $a_{\text{n}}, a_{\text{n-1}}, ..., a_0$ denotes the parameters of the denominator of $G(s)$, f is a static nonlinear function and T_{d} is the input delay. At equilibrium, with $u_{\text{ss}} = f_{\text{m}}^{-1}(r - d_{\text{ss}})$, the steady-state conditions become

$$\dot{x}_{1,\text{ss}} = x_{2,\text{ss}} = 0 \tag{21}$$

$$\dot{x}_{2,\text{ss}} = x_{3,\text{ss}} = 0 \tag{22}$$

$$\cdots \tag{23}$$

$$\dot{x}_{\text{n,ss}} = -\frac{a_0}{a_{\text{n}}}x_{1,\text{ss}} - \frac{a_1}{a_{\text{n}}}x_{2,\text{ss}} + \cdots - \frac{a_{\text{n-1}}}{a_{\text{n}}}x_{\text{n,ss}} + \frac{1}{a_{\text{n}}}f(f_{\text{m}}^{-1}(r - d_{\text{ss}})) \tag{24}$$

$$= -\frac{a_0}{a_{\text{n}}}x_{1,\text{ss}} + \frac{1}{a_{\text{n}}}f(f_{\text{m}}^{-1}(r - d_{\text{ss}})) = 0 \tag{25}$$

$$x_{1,\text{ss}} = \frac{1}{a_0}f(f_{\text{m}}^{-1}(r - d_{\text{ss}})) = c, \tag{26}$$

and thus $\mathbf{x}_{\text{ss}} = \begin{bmatrix} c & 0 & \cdots & 0 \end{bmatrix}^\top$, with unknown constant c for the settled system. Defining the deviation coordinates yields

$$z_1(t) = x_1(t) - c \Rightarrow x_1(t) = z_1(t) + c \tag{27}$$

$$z_{\text{i}}(t) = x_{\text{i}}(t) \Rightarrow x_{\text{i}}(t) = z_{\text{i}}(t), \qquad \text{i} = 2, \ldots, \text{n},$$

$$\dot{z}_1(t) = \dot{x}_1(t) = x_2(t) = z_2(t) \tag{28}$$
$$\dot{z}_2(t) = \dot{x}_2(t) = x_3(t) = z_3(t) \tag{29}$$
$$\dots$$
$$\dot{z}_{\text{n-1}}(t) = \dot{x}_{\text{n-1}}(t) = x_{\text{n}}(t) = z_{\text{n}}(t) \tag{30}$$
$$\dot{z}_{\text{n}}(t) = \dot{x}_{\text{n}}(t) = -\frac{a_0}{a_{\text{n}}} x_1(t) - \frac{a_1}{a_{\text{n}}} x_2(t) + \cdots - \frac{a_{\text{n-1}}}{a_{\text{n}}} x_{\text{n}}(t) + \frac{1}{a_{\text{n}}} f(u(t - T_{\text{d}})) \tag{31}$$
$$= -\frac{a_0}{a_{\text{n}}}(z_1(t) + c) - \frac{a_1}{a_{\text{n}}} z_2(t) + \cdots - \frac{a_{\text{n-1}}}{a_{\text{n}}} z_{\text{n}}(t) + \frac{1}{a_{\text{n}}} f(u(t - T_{\text{d}})) \tag{32}$$

Because the linear part is stable, there exists for any $\mathbf{Q} = \mathbf{Q}^\top > 0$ a $\mathbf{P} = \mathbf{P}^\top > 0$ with

$$\mathbf{A}^\top \mathbf{P} + \mathbf{P}\mathbf{A} = -\mathbf{Q}. \tag{33}$$

Choosing the Lyapunov candidate as

$$V(\mathbf{z}) = \mathbf{z}^\top \mathbf{P} \mathbf{z}, \tag{34}$$

we get for the closed-loop system

$$\dot{V}(\mathbf{z}) = -\mathbf{z}^\top \mathbf{Q} \mathbf{z} + 2\mathbf{z}^\top \mathbf{P}\mathbf{B}(f(f_{\text{m}}^{-1}(r - d(t - T_{\text{d}}))) - a_0 c) \tag{35}$$
$$= -\mathbf{z}^\top \mathbf{Q} \mathbf{z} + 2\mathbf{z}^\top \mathbf{P}\mathbf{B} (f(f_{\text{m}}^{-1}(r - d(t - T_{\text{d}}))) - f(f_{\text{m}}^{-1}(r - d_{\text{ss}}))). \tag{36}$$

In equilibrium, the second part of the equation becomes 0. Therefore, we can define the distance to equilibrium function $\xi(t)$ as

$$\xi(t) = f(f_{\text{m}}^{-1}(r - d(t - T_{\text{d}}))) - f(f_{\text{m}}^{-1}(r - d_{\text{ss}})) \tag{37}$$
$$\xi_{\text{ss}} = 0, \tag{38}$$

which is bounded as the input is constrained and $f(u(t))$ is locally Lipschitz over $u(t)$. This aligns with Input-to-State Stability (ISS). The derivative of the Lyapunov function is

$$\dot{V}(\mathbf{z}) = -\mathbf{z}^\top \mathbf{Q} \mathbf{z} + 2\mathbf{z}^\top \mathbf{P}\mathbf{B}\xi(t). \tag{39}$$

Based on the Cauchy–Schwarz inequality $2\mathbf{z}^\top \mathbf{P}\mathbf{B}\xi(t) \leq 2\|\mathbf{z}\|\|\mathbf{P}\mathbf{B}\||\xi(t)|$ it becomes

$$\dot{V}(\mathbf{z}) = -\mathbf{z}^\top \mathbf{Q} \mathbf{z} + 2\|\mathbf{z}\|\|\mathbf{P}\mathbf{B}\||\xi(t)|. \tag{40}$$

Applying Young's inequality with an arbitrary constant $\epsilon > 0$

$$|\xi(t)|\|\mathbf{z}\| \leq \frac{1}{2\epsilon}\xi^2(t) + \frac{\epsilon}{2}\mathbf{z}^2. \tag{41}$$

Substituting (41) into (40) yields

$$\dot{V}(t) \leq -\mathbf{z}^\top \mathbf{Q} \mathbf{z} + \frac{1}{\epsilon}\|\mathbf{P}\mathbf{B}\|\xi^2(t) + \epsilon\|\mathbf{P}\mathbf{B}\|\mathbf{z}^2. \tag{42}$$

Choose the ISS gain with constant limit $\rho > 0$ and arbitrary function $v(t)$

$$\chi(v) := \frac{1}{\sqrt{\epsilon^2 - \rho}} v, \tag{43}$$

so that for $\mathbf{z}, \xi : \|\mathbf{z}\| \geq \chi(|\xi|)$

$$\xi(t)^2 \leq (\epsilon^2 - \rho)\mathbf{z}^2. \tag{44}$$

Fig. 1. Picture of the Hardware used for experiments

Combining (44) with (42) gives

$$\dot{V}(t) \leq -\mathbf{z}^\top \mathbf{Q} \mathbf{z} - \frac{\rho}{\epsilon}\|\mathbf{P}\mathbf{B}\|\mathbf{z}^2 + 2\epsilon\|\mathbf{P}\mathbf{B}\|\mathbf{z}^2. \tag{45}$$

Defining

$$\alpha = \lambda_{\min}(\mathbf{Q}) + \left(\frac{\rho}{\epsilon} - 2\epsilon\right)\|\mathbf{P}\mathbf{B}\|, \tag{46}$$

(45) becomes

$$\dot{V}(t) \leq -\alpha \mathbf{z}^2. \tag{47}$$

Hence, the closed-loop system is ISS for $\rho > 2\epsilon^2$.

III. Application of nonlinear PFC to angular velocity control with PX4

A. Set-Up

1) Hardware: The controller developed in the previous section is evaluated on a physical multicopter propulsion system. The experimental platform is shown in Fig. 1. The system employs a Holybro Pixhawk 6X flight controller running the PX4 autopilot firmware. The Pixhawk interfaces with the Electronic Speed Controller (ESC), a T-Motor MN3510-13, via a pulse-width modulation signal. Communication with the controller is established through the MATLAB PX4 Toolbox, which provides convenient access to the PX4 middleware but restricts the firmware version to PX4 v1.14.3.

The ESC is driven using the PX4 Actuator Write block, which accepts a normalised throttle command $u_{\text{PX4}} \in [0, 1]$, corresponding to the percentage of the maximum rotor speed. Due to hardware constraints—and because any value below $u_{\text{PX4}} = 0.1$ shuts down the rotor and is therefore treated as the discrete state "off"—the effective control domain in this work is restricted to $u_{\text{c}} \in [0.1, 1)$.

Rotor is measured using a Littelfuse 55100 miniature flange-mount Hall-effect sensor. A ThunderFly TFRPM01 module processes the Hall sensor output and generates a digital revolution count, which is transmitted to the Pixhawk via an I^{2}C interface. A custom PX4 extension interprets this count to generate an internal message providing rotor angular velocities in $\mathrm{rad/s}$.

Although each subsystem (PX4, ESC, motor, and sensor) can be analysed individually—PX4 and the ESC firmware are fully open source—deriving a complete analytical model would require significant effort. Consequently, the overall propulsion channel is treated as a nonlinear, input-constrained, delayed, SISO black-box system.

2) Model: The propulsion system is represented using a Hammerstein model of the form

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 \\ -\frac{1}{T_\mathrm{p}^2} & -\frac{2}{T_\mathrm{p}} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ \frac{1}{T_\mathrm{p}^2} \end{bmatrix} f(u_\mathrm{c}(t - T_\mathrm{d})) \quad (48)$$

$$y_\mathrm{p}(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}(t) \quad (49)$$

$$f(u_\mathrm{c}) = v_{0.5} u_\mathrm{c}^{0.5} + v_0 + v_1 u_\mathrm{c} + v_2 u_\mathrm{c}^2 + v_3 u_\mathrm{c}^3 + v_4 u_\mathrm{c}^4, \quad (50)$$

where T_p denotes the system time constant, T_d the input delay, and $v_{0.5}, v_0, \ldots, v_4$ are the coefficients of the static nonlinear mapping.

Parameter identification is performed in two stages. First, the nonlinear static mapping is simplified to $f(u_\mathrm{c}(t)) = k u_\mathrm{c}(t)$ via a coarse estimate of the static gain k. Under this simplification, the parameters T_p and T_d are identified by minimising the discrete-time squared error between the measured output $y_\mathrm{p}(k)$ and the model output $y_\mathrm{m}(k)$:

$$\min_{T_\mathrm{p}, T_\mathrm{d}} \sum (y_\mathrm{p}(k) - y_\mathrm{m}(k))^2 \quad (51)$$

Once the dynamic parameters are obtained, the full nonlinear function $f(u_\mathrm{c}(t))$ is parameterised. The coefficients $v_{0.5}, v_0, \ldots, v_4$ are determined by minimising the same squared-error criterion between measured and simulated outputs.

The resulting parameter set is provided in Table I for completeness.

TABLE I
PARAMETERS OF THE SYSTEM MODEL

Parameter	Value	Parameter	Value
T_p	0.129	v_1	969.8
T_d	0.096	v_2	1312.3
v_0	0	v_3	-3191.2
$v_{0.5}$	-125.0	v_4	1737.5

B. Inversion of the static nonlinearity

To control the physical system described in Section III-A, the control framework developed in Section II is implemented. A key requirement is the computation of the inverse nonlinearity $f^{-1}(u_\mathrm{c})$. Two approaches are considered: (i) numerical inversion via a sequential search method based on the golden-section algorithm and (ii) a discretised look-up table approximation.

1) Golden-Section Search for $f^{-1}(u_c)$: In the initial implementation, the inverse mapping is obtained using a univariate optimisation routine employing the golden-section search. This method iteratively contracts a bounded interval by evaluating the cost function $J(u_\mathrm{c})$ at two interior points whose locations depend on the golden ratio. Starting from an initial interval $[a_0,\ b_0]$ with length $d_0 = b_0 - a_0$, two candidate points x_1 and x_2 are computed at iteration k as

$$x_{1,k} = b_{k-1} - d_k, \quad (52)$$

$$x_{2,k} = a_{k-1} + d_k. \quad (53)$$

The interval is then updated according to the relative values of $J(x_{1,k})$ and $J(x_{2,k})$:

$$\text{if } J(x_{1,k}) < J(x_{2,k}) \Rightarrow a_k = a_{k-1};\ b_k = x_{2,k}, \quad (54)$$

$$\text{if } J(x_{1,k}) > J(x_{2,k}) \Rightarrow a_k = x_{1,k};\ b_k = b_{k-1}. \quad (55)$$

This process yields a contracted interval $[a_k,\ b_k]$ for the subsequent iteration. The contraction step size d_k is determined using the golden ratio:

$$d_k = \left(\frac{\sqrt{5}-1}{2}\right) d_{k-1} = \left(\frac{\sqrt{5}-1}{2}\right)^k d_0 \quad (56)$$

Iterations proceed until the termination condition $|b_k - a_k| < 0.01$ is satisfied. For all tests in this study, the initial interval was set to $[0.1,\ 1)$.

2) Look-Up-Table Approximation: Given the limited computational resources of the embedded system, an alternative look-up table-based approach is also investigated. In nonlinear applications, look-up tables typically require a large number of stored data points; however, the termination threshold used in the optimisation scheme indicates that a discretisation into 89 equidistant samples over $[0.1,\ 1)$ is sufficient to meet the desired resolution. Specifically, the set of values over $[0.1 : 0.01 : 1)$, with the final element defined as $1 - \vartheta$ (where ϑ denotes the smallest representable number in the target programming language), is used to populate the look-up table.

Employing the look-up table reduces online computational load at the expense of a minimal increase in memory usage. This trade-off is favourable for the hardware platform considered, and both inversion methods are evaluated for feasibility.

C. Experimental verification

1) Set-point tracking: To evaluate the controller's performance, a set-point tracking experiment was conducted. The PFC was required to follow a sequence of reference steps. The reference angular velocity was increased by 100 $\mathrm{rad/s}$ every 5 s, ranging from 100 $\mathrm{rad/s}$ to 600 $\mathrm{rad/s}$. The resulting closed-loop step response for the inversion with optimisation is shown in Fig. 2. It should be noted that the experiment—and therefore the plot—begins at 20 s. This offset arises because the measurement tools did not start recording at time zero, but approximately at 10 s. An additional 10 s margin was added to avoid complications.

Controller performance was assessed by computing the error between the reference and the plant measurement $y_\mathrm{p}(t)$. The square root of the squared error is shown on a logarithmic scale in Fig. 3. A logarithmic representation highlights both the large deviation at step transitions and the

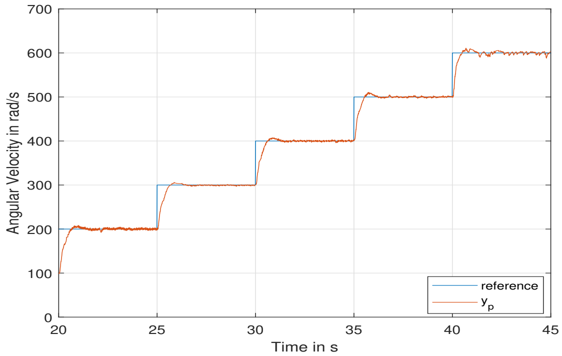

Fig. 2. Closed-loop step response

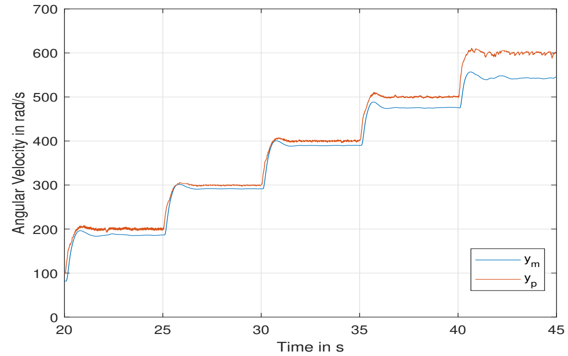

Fig. 4. Comparison of y_{p} to y_{m}

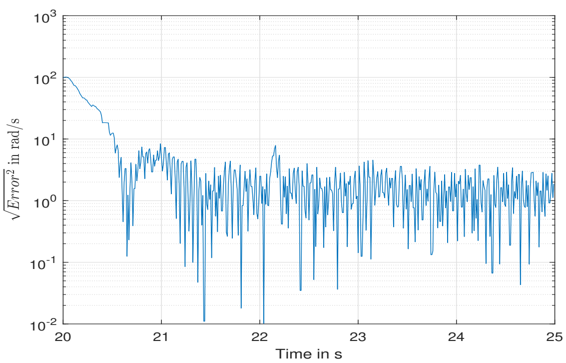

Fig. 3. Square root of squared tracking error on a logarithmic scale

small tracking errors during steady-state operation, thereby improving interpretability. Fig. 3 illustrates the square root of the squared error for the first reference step. Starting from the initial deviation, the error approaches zero within approximately 0.6s, exhibits a slight overshoot, and stabilises thereafter. This behaviour is repeated for all subsequent steps, confirming stable closed-loop performance.

For the delayed system and the inversion using the look-up table, the corresponding plots show no significant differences from Figs. 2 and 3; therefore, they are omitted for brevity.

Table II summarises the maximum error, minimum error, root-mean-squared (RMS) error, and each error's magnitude relative to the hover angular velocity (524.61 $\mathrm{rad/s}$), for all three configurations: optimisation without delay compensation, optimisation with delay compensation, and the look-up-table-based inversion for the settled system. From Table II,

TABLE II

TRACKING ERROR METRICS

Error type	Owod[1]	Owd[2]	LUT[3]
Max Absolute	11.89(2.3%)	17.37(3.3%)	10.49(2,0%)
Min Absolute	$6 \cdot 10^{-4}$(0.0%)	$8 \cdot 10^{-4}$(0.0%)	$3 \cdot 10^{-4}$(0.0%)
RMS[4]	1.94(0.4%)	3.32(0.6%)	1.64(0.3%)

[1] Optimisation without delay in $\mathrm{rad/s}$(% of hover)
[2] Optimisation with delay in $\mathrm{rad/s}$(% of hover)
[3] Look-Up Table in $\mathrm{rad/s}$(% of hover)
[4] Root Mean Squared

it can be observed that the controller remains functional even without delay compensation, although performance degrades slightly on average. The controller without delay compensation exhibits instability at higher PFC operating frequencies. Graphics demonstrating this behaviour are omitted for brevity. Comparing the optimised controller with the look-up-table-based approach shows that both achieve similar accuracy. Thus, the look-up table constitutes a viable alternative to optimisation in this application. In all three cases, the closed-loop system exhibits stable behaviour, and the resulting errors are sufficiently small for practical use.

To demonstrate the controller's inherent capability to compensate for model inaccuracies, the measured process output y_{p} is compared with the simulated model output y_{m}. The results are shown in Fig. 4. As observed in Fig. 4, the model prediction deviates from the measured response across the entire operating range, with the discrepancy increasing at higher angular velocities. Despite this mismatch, the process output converges toward the reference signal. This behaviour highlights the robustness of the controller design in compensating for plant–model discrepancies.

2) Trajectory tracking: To further evaluate the controller's performance, a trajectory-tracking experiment was conducted. The reference trajectory is defined as

$$r = 500 + 100 \cdot \sin(t - 60). \tag{57}$$

The experimental results obtained using the inversion with optimisation are presented in Fig. 5. This experiment was performed in the same run as the set-point tracking test.

Consequently, the recorded trajectory-tracking data begins at $t = 45\mathrm{s}$, immediately following the data shown in Fig. 2. Controller performance was evaluated by computing the tracking error between the reference and the plant output y_{p}, shown in Fig. 6. Following a set-point change of approximately $150\mathrm{rad/s}$, the controller exhibits a settling time comparable to the set-point tracking results. The system approaches the reference trajectory within approximately $0.7\mathrm{s}$ and enters a steady oscillatory regime after a minor overshoot. The steady-state error appears sinusoidal due to the time delay between the reference and the system response. The maximum absolute error occurs near the

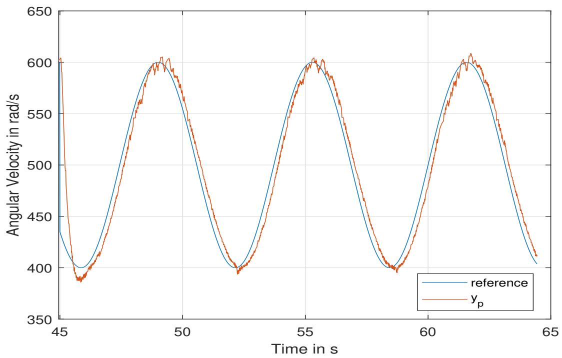

Fig. 5. Comparison of reference and measured trajectory

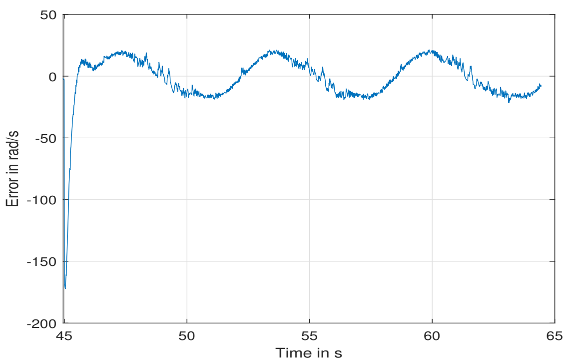

Fig. 6. Tracking error for the trajectory-following experiment

trajectory inflection points, where the reference changes most rapidly, and the system exhibits the greatest lag. The average phase shift is approximately $0.166\mathrm{s}$, of which $0.096\mathrm{s}$ is caused by the inherent system delay and the remaining $0.07\mathrm{s}$ by the closed-loop transient response. As in the set-point tracking case, the delayed system and the look-up-table-based inversion show no significant deviations from the results in Figs. 5 and 6 and are therefore omitted for brevity.

TABLE III
TRAJECTORY-TRACKING ERROR METRICS

Error type	Owod[1]	Owd[2]	LUT[3]
Max Absolute[4]	21.58(4.1%)	27.60(5.26%)	21.93(4.2%)
Min Absolute[4]	0.01(0.0%)	0.01(0.0%)	0.02(0.0%)
RMS[4,5]	13.00(2.5%)	10.54(2.0%)	12.75(2.4%)
Phase Shift/s	0.17	0.09	0.17

[1] Optimisation without delay
[2] Optimisation with delay
[3] Look-Up Table
[4] in $\mathrm{rad/s}$(% of hover)
[5] Root Mean Squared

Table III summarises the maximum, minimum, and RMS errors, together with the error magnitude relative to the hover angular velocity, for all three configurations: optimisation without delay compensation, optimisation with delay compensation, and look-up-table-based inversion. The results confirm the observations from the set-point tracking experiments: both inversion approaches achieve comparable tracking accuracy and stable behaviour. Neglecting the system delay slightly improves phase shift performance during trajectory tracking, but increases peak errors due to stronger overshoot near the trajectory extrema. Thus, although the no-delay configuration reduces phase lag, it handles direction changes less effectively.

To contextualise the results, a PID controller applied to the same experimental setup—previously used in laboratory studies—yields performance comparable to the approach proposed in this paper. Specifically, it exhibits a slightly larger overshoot of approximately $30\mathrm{rad/s}$, a similar settling time, and a phase shift of about $0.12\mathrm{s}$.

IV. CONCLUSIONS

The paper presents a control pipeline for speed control of multirotor robots. Such a pipeline relies on the use of Hammerstein models to represent the system dynamics, which is at the core of a predictive functional controller. Input-to-state stability is proven, and the effectiveness of the pipeline is shown in an experimental study, where optimisation-based inversion of the static nonlinear mapping is compared with a more embed-friendly inversion via a lookup table, together with the impact of considering or neglecting delays in the optimisation-based inversion. For future work, the control approach can be extended to the Hammerstein–Wiener model to address more complex system dynamics.

REFERENCES

[1] O. Fritsch, D. Tromba, and B. Lohmann, "Cascaded energy based trajectory tracking control of a quadrotor," *at - Automatisierungstechnik*, vol. 62, no. 6, pp. 408–422, 2014. [Online]. Available: https://doi.org/10.1515/auto-2013-1063
[2] M. Islam, M. Okasha, and E. Sulaeman, "A model predictive control (mpc) approach on unit quaternion orientation based quadrotor for trajectory tracking," *International Journal of Control, Automation and Systems*, vol. 17, no. 11, pp. 2819–2832, 2019.
[3] D. Tzoumanikas, Q. Yan, and S. Leutenegger, "Nonlinear mpc with motor failure identification and recovery for safe and aggressive multicopter flight," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8538–8544.
[4] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 6235–6240.
[5] S. A. Billings and S. Y. Fakhouri, "Non-linear system identification using the hammerstein model," *International Journal of Systems Science*, vol. 10, no. 5, pp. 567–578, 1979.
[6] E.-W. Bai and M. Fu, "A blind approach to hammerstein model identification," *IEEE Transactions on Signal Processing*, vol. 50, no. 7, pp. 1610–1619, 2002.
[7] H. H. J. Bloemen, T. J. J. van den Boom, and H. B. Verbruggen, "Model-based predictive control for hammerstein?wiener systems," *International Journal of Control*, vol. 74, no. 5, pp. 482–495, 2001.
[8] K. H. Chan and J. Bao, "Model predictive control of hammerstein systems with multivariable nonlinearities," *Industrial & Engineering Chemistry Research*, vol. 46, no. 1, pp. 168–180, 2007.
[9] K. P. Fruzzetti, A. Palazoglu, and K. A. McDonald, "Nonlinear model predictive control using hammerstein models," *Journal of Process Control*, vol. 7, no. 1, pp. 31–41, 1997.
[10] J. Richalet and D. O'Donovan, *Predictive Functional Control: Principles and Industrial Applications*. Springer, 2009.
[11] Q. Zhang, Q. Wang, and G. Li, "Nonlinear modeling and predictive functional control of hammerstein system with application to the turntable servo system," *Mechanical Systems and Signal Processing*, vol. 72-73, pp. 383–394, 2016.