Real-Time Implementation of Hammerstein Models for Temperature Estimation in Permanent-Magnet Synchronous Motors

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 July 2026, accepted 13 August 2026, date of publication 18 August 2026, date of current version 24 August 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3724791

RESEARCH ARTICLE

Real-Time Implementation of Hammerstein Models for Temperature Estimation in Permanent-Magnet Synchronous Motors

ERICK AXEL MARTINEZ-RÍOS [1,2], IRVING S. AGUILAR-ZAMORATE [1], SAULIUS PAKŠTYS [3], RENATO GALLUZZI [1], (Senior Member, IEEE), LUIS M. CASTELLANOS MOLINA [3], NICOLA AMATI [3], AND ROGELIO BUSTAMANTE-BELLO [1]

[1] School of Engineering and Sciences, Tecnologico de Monterrey, Mexico City 14380, Mexico
[2] Advanced Modeling Sciences, PepsiCo Research and Development, Mexico City 02300, Mexico
[3] Center for Automotive Research and Sustainable Mobility, Politecnico di Torino, 10129 Turin, Italy

Corresponding author: Renato Galluzzi (renato.galluzzi@tec.mx)

This work was supported in part by the EM-TECH Research Project, part of the 2Zero partnership funded by the European Commission under Grant 101096083. The work of Erick Axel Martinez-Ríos and Irving S. Aguilar-Zamorate was supported by Secretaría de Ciencia, Humanidades, Tecnología e Innovación (SECIHTI), under Grant 1010770 and Grant 1105077. The work of Saulius Pakštys was supported by the PNRR-NGEU Project under Grant MUR–DM 352/2022.

ABSTRACT Accurate real-time temperature estimation of permanent-magnet synchronous motors is essential to prevent overheating and ensure reliable operation. To this end, electric and thermal models are often used due to their low computational burden. However, parameter estimation for these models is challenging and error-prone. Though less sensitive to errors, data-driven techniques demand numerous parameters and are difficult to implement in real time. Additionally, considerations for translating models from double to single precision are often overlooked. This paper presents a numerical stability analysis and a real-time implementation of Hammerstein models initialized with a lumped-parameter thermal network to estimate the temperature in the windings and magnets of an out-runner permanent-magnet synchronous motor. The numerical stability of the Hammerstein model was evaluated to assess the feasibility of converting its coefficients from double to single precision by computing the condition number and eigenvalues of the state-transition matrix in the Hammerstein model's linear block. A balanced state-space realization was used to reduce the condition number and avoid truncation errors when converting parameters to single precision. The model was implemented via a processor-in-the-loop simulation on an inverter-grade 32-bit microcontroller using MATLAB-Simulink, where the execution time and CPU utilization were computed. Results show that balancing the Hammerstein linear block improves the condition number of the state-transition matrix below 1.75, with an execution time under 200 μs and CPU utilization below 0.04%.

INDEX TERMS Hammerstein model, hardware implementation, numerical stability, processor-in-the-loop.

I. INTRODUCTION

Thermal monitoring of permanent-magnet synchronous motors (PMSMs) is crucial to prevent component degradation. Faults generated due to thermal stress in PMSMs can be classified as electrical and magnetic [1]. The former are associated with the melting of the winding insulation, which could produce short circuits. Conversely, magnetic faults are related to the irreversible demagnetization of permanent magnets (PMs), which reduces the torque capability of the machine [2]. Preventing thermal stress issues is crucial for high-performance applications such as vehicle traction, where the machine is pushed to its limits and durability is paramount. However, the thermal monitoring of PMSMs is

The associate editor coordinating the review of this manuscript and approving it for publication was Amin Mahmoudi.

not a trivial task, and it has led to the development of different methods for estimating the temperature of components, such as finite element analysis and computational fluid dynamics tools, electric and thermal models, or data-driven techniques [3].

From the available PMSM thermal monitoring methods, real-time temperature estimation methods are crucial since they allow a proactive intervention—adjusting the load, reducing power, or increasing the cooling—to prevent overheating and irreversible damage to the machine [4]. Although direct measurements using contact and contactless temperature sensors can serve this purpose well, using high-precision sensors is costly. Sensor inclusion usually requires performing structural changes in the machine. Installation and measurement difficulties arise when addressing rotating components [5], [6]. Thus, estimation techniques are necessary.

Finite-element tools are not feasible for real-time thermal monitoring of PMSMs due to their computational complexity [7], [8]. Electric and thermal models are preferred for real-time temperature estimation due to their lower computational burden [9]. Electric models try to estimate the temperature inside the machine by estimating temperature-sensitive parameters such as copper resistivity or PM flux linkage. These methods often depend on flux observers relying on phase currents and voltage measurements [10]. Nevertheless, electric models are highly sensitive to parameter estimation errors, as small deviations in the electrical signals can produce a significant variation in the temperature estimates [11], [12]. Large errors are present, especially at low speed, where the signal-to-noise ratio of voltage measurements is low.

In this respect, lumped-parameter thermal networks (LPTN) have been extensively proposed to estimate temperature in real time [13], [14], [15]. In contrast to flux observers, LPTN models can be applied across the full torque and speed range of the machine [3]. The state-of-the-art has explored the development of LPTN models in real time from the second to the fifth order [16]. Nonetheless, these models are populated with parameters dependent on the machine geometry, the properties of its cooling system and the rotor speed, among other features. Model accuracy comes at the cost of correctly identifying these parameters using sufficient experimental data obtained from a diverse set of operational conditions [17], [18]. While standard low-order linear parameter-varying (PV) LPTN models are computationally fast to execute once established [3], [19], their true deployment challenge lies in their susceptibility to numerical ill-conditioning and truncation errors when translated to single-precision embedded environments. Furthermore, real-time temperature observers are integrated alongside the motor controller on a shared embedded processor. Although the thermal dynamics of PMSMs are slow and can be reliably evaluated with very long integration steps (on the order of 1 s), an execution-time target below 1 ms was adopted to provide nominal scheduling margin when the thermal observer shares the processor with the motor control tasks. Conflict-free integration must nevertheless be verified through worst-case execution-time, interrupt-latency, and jitter analyses under concurrent motor control execution.

Kalman filtering provides a complementary framework for PMSM thermal estimation by recursively combining process model predictions with temperature-sensitive electrical information while accounting for process and measurement uncertainties. Representative applications use high-frequency signal injection [20], speed-harmonic information [21], or a flux-based PM-temperature observer with a reduced-order LPTN [22]. In contrast, the present work does not add a recursive measurement-update stage but evaluates a nonlinear Hammerstein-LPTN process model and its single-precision implementation. A Kalman architecture could therefore use the proposed model together with an electrical temperature measurement in future work.

The challenges stemming from parameter estimation in PMSMs have motivated data-driven PMSM temperature estimators, including support vector regression and regression trees, as well as deep architectures based on recurrent and convolutional neural networks [23], [24], [25]. Recent neural thermal estimators combine data-driven nonlinearities with LPTN structure. Thermal neural networks retain an LPTN-like state-space formulation [26], whereas LPTN-informed long short-term memory networks (LSTM) compensate for unmodeled thermal dynamics [1]. These hybrid approaches can improve prediction accuracy, while preserving some physical interpretability. However, they require representative training and validation data, while large or purely black-box models may increase embedded resource demands and complicate functional-safety assessment [27].

Very few works have evaluated the execution time and numerical equivalence of data-driven methods on embedded hardware. Kirchgässner et al. used recurrent neural networks and temporal convolutional networks for stator and rotor temperature estimation in PMSMs [23]. Their trained models were embedded in a Raspberry Pi 3 Model B+. However, the Raspberry platform does not have a real-time operating system, making it susceptible to latency. In [28], a hybrid model consisting of a second-order LPTN and a multilayer neural network was used to estimate the temperature in the stator and rotor of a PMSM, and the resultant model was implemented in a 32-bit digital signal processor. This combination retains part of the physical essence of the system through the LPTN, while also exploiting the adaptability of a data-driven approach. Nevertheless, details about the implementation and architecture of the neural network were not reported and the proposed LPTN model did not consider the effect of the varying thermal resistance due to air convection surrounding the PM. Other hybrid data-driven approaches have been proposed that combine LPTN with neural networks, such as the Hammerstein model [29].

However, that study did not examine the numerical conditioning of the state-space realization, or quantify the effects of single-precision coefficient conversion on the model poles and dynamic response.

Another aspect that has not been addressed in the related research is how to deal with the sensitivity that LPTN dynamic models could have due to closely spaced poles, which make them susceptible to numerical instability when performing truncation or round-off errors [30], [31], [32]. In a state-space model representation, pole sensitivity can produce an ill-conditioned state-transition matrix, impacting eigenvalue computations and, consequently, model dynamics [33]. Improperly scaled models and conversion between representations—from state space to transfer function and vice versa—could introduce numerical errors in the model, particularly for multiple-input multiple-output systems expressed in companion, canonical, and observable forms [34]. These aspects have implications for the computational resources required to execute the estimated temperature model on a restricted hardware platform, because, in most cases, the model is trained on a personal computer with a 64-bit processor and double-precision data. In contrast, when translated to embedded hardware, model parameters are truncated to 32-bit floating-point numbers in the best case. This operation introduces truncation errors in the estimated model coefficients, consequently hindering the estimated model dynamics [35].

Although Hammerstein-LPTN models have been proposed for PMSM thermal estimation, most studies have focused on estimation accuracy [29], [36]. Less attention has been paid to the numerical robustness of these models when implemented on embedded hardware. In particular, the effects of model conditioning, coefficient truncation, and finite-precision arithmetic on model stability and dynamic behavior remain largely unexplored. Addressing these aspects is essential to ensure the reliable deployment of thermal estimators on resource-constrained motor-drive controllers.

This work presents primary considerations for implementing ill-conditioned Hammerstein models estimated via MATLAB-Simulink on a 32-bit embedded hardware platform typical for motor inverters. The microcontroller unit is used to estimate the winding and PM hotspot temperatures of an out-runner PMSM in real time. To this end, a Hammerstein model was initialized via a linear time-invariant (LTI) LPTN of the fourth order. This method has been experimentally proven as adequate for temperature estimation in the target application [29]. The Hammerstein model is analyzed in terms of model stability via the computation of its eigenvalues. Its numerical stability is assessed by calculating the condition number of the state-transition matrix. These indicators allow presenting the main issues and considerations when embedding the proposed model into a 32-bit embedded platform in real time. It should be noted that the objective of this work is not to benchmark different thermal estimation algorithms, but rather to investigate the practical deployment of a Hammerstein-based

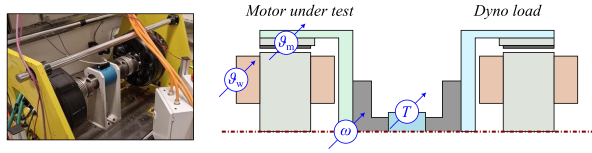


FIGURE 1. Testbed back-to-back setup picture (left) and diagram (right) with motor under test (torque controlled) and dyno load (speed controlled). The testbed is instrumented to measure mechanical torque (T), rotor angular speed (ω), winding temperature (ϑ_w) and rotor-surface temperature. The permanent-magnet (PM) temperature reference (ϑ_m) is derived from the rotor-surface measurement as described in Section II-A.

LPTN model on resource-constrained automotive embedded hardware. A detailed comparison between conventional LPTN, Hammerstein, and nonlinear autoregressive models with exogenous inputs, including estimation accuracy, has already been presented in [29]. Therefore, the present study focuses instead on implementation aspects that have received considerably less attention.

The main contributions of this work are listed as follows:

- Finite-precision implementation analysis and resource characterization of the previously validated Hammerstein estimator initialized from a fourth-order LPTN, using processor-in-the-loop (PIL) execution on a 32-bit motor-control microcontroller for estimating winding and permanent-magnet hotspot temperatures in an out-runner PMSM.
- An analysis of the numerical stability of the proposed model through the evaluation of its eigenvalues and the condition number of the state-transition matrix.
- A set of practical considerations for the deployment of ill-conditioned thermal estimation models on resource-constrained motor-drive controllers.

The remainder of this manuscript is structured as follows. Section II presents the dataset and the theoretical background of the methods employed. Section III shows the results and discussions. Finally, Section IV presents the conclusions and potential future work.

II. MATERIALS AND METHODS

A. DATASET

The experimental data for estimating and validating the Hammerstein model were collected from a 200-kW liquid-jacket-cooled out-runner PMSM utilized in vehicle powertrain applications. The machine was driven by applying two standardized driving cycles in sequence—Federal Test Procedure-75 (FTP-75) and Worldwide Harmonized Light Vehicles Test Cycle, Class 3 (WLTC3)—both preceded by preheating periods. Each preheating period lasts for 1800 s, while the FTP-75 and WLTC3 correspond to 1874 s and 1800 s, respectively; thus, the full test lasts 7274 s.

To this end, the back-to-back motor configuration in Fig. 1 was devised, where the dyno load was speed-controlled and the machine under test was torque-controlled. Figure 2 illustrates the profile used to drive the motor. The complete 7274-s sequence was conducted once at each coolant inlet

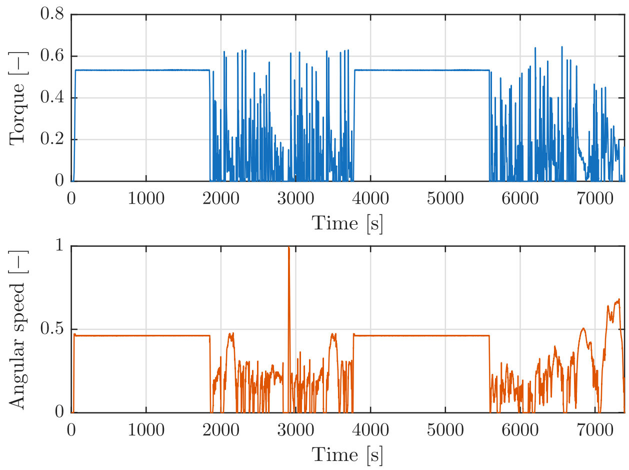


FIGURE 2. Torque (top) and angular speed (bottom) profiles that drive the motor during testing. Axes have been normalized to protect proprietary data.

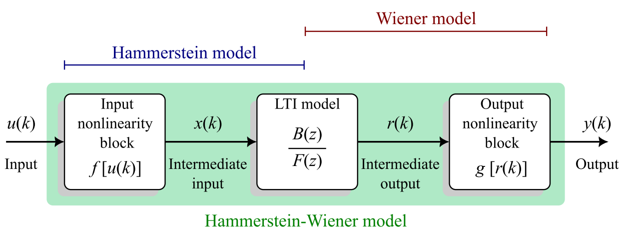


FIGURE 3. General architecture of a Hammerstein–Wiener model. Adapted from [29].

temperature of 20, 40, and 60 °C. Thus, three complete tests were conducted, and no independent replicate runs were available at any coolant-temperature condition.

During each test, the winding temperature, rotor-surface temperature, coolant inlet temperature, ambient temperature, motor torque and rotational speed were recorded. All six signals were sampled at 5 Hz. PM temperature was estimated by applying a constant offset of 5 °C to the rotor surface temperature. The datasets were partitioned by complete coolant-temperature condition rather than by randomly assigning individual time samples. The complete datasets obtained at 20 and 60 °C were used jointly for model identification, whereas the complete dataset obtained at 40 °C was reserved for validation. Consequently, validation evaluates interpolation to an unseen intermediate coolant-temperature condition under the same driving-cycle sequence. The PM-reference correction and further details of the experimental setup are provided in previous work [29]. Exact motor geometry, winding arrangement, and internal cooling-jacket geometry cannot be disclosed because of third-party confidentiality restrictions.

B. HAMMERSTEIN MODELS

Hammerstein models are a particular case of the so-called Hammerstein-Wiener representations used in nonlinear system identification. The core idea of Hammerstein-Wiener models is that a system model can be divided into blocks, in particular, an input and output nonlinearity block and an LTI block represented by the transfer function $B(z)/F(z)$. The general structure of the Hammerstein-Wiener model is illustrated in Fig. 3. From this representation, it can be noticed that using the input nonlinearity before the linear model constitutes a Hammerstein representation [37], [38], [39].

Mathematically, Hammerstein models can be expressed as

$$x(k) = f[u(k)], \qquad y(k) = s(k),$$
$$r(k) = \frac{B(z)}{F(z)}x(k), \quad s(k) = r(k) + v(k) \tag{1}$$

where $u(k)$ are the system inputs, $y(k)$ are the system outputs, $x(k)$ is the intermediate input, $r(k)$ is the intermediate output, $s(k)$ is the system intermediate variable, and $v(k)$ represents stochastic white noise with zero mean. The LTI block of the system can be represented in polynomial form by considering $B(z) = b_1z^{-1} + b_2z^{-2} + \cdots + b_qz^{-q}$ and $F(z) = 1 + a_1z^{-1} + a_2z^{-2} + \cdots + a_pz^{-p}$ in terms of the unit delay operator.

The above structure allows the incorporation of the PMSM's linear thermal dynamics through the linear block of the Hammerstein models. On the other hand, the input nonlinearity can be modeled through neural networks, dead zones, polynomial functions, or saturations. For this work, the LTI block of the Hammerstein model was generated using the PMSM's LPTN, and the input nonlinearity was built with sigmoid neural networks. This work used the System Identification Toolbox in MATLAB R2023b to estimate the Hammerstein model.

C. LINEAR TIME-INVARIANT LUMPED-PARAMETER THERMAL NETWORK

The LTI LPTN model of the PMSM was based on the fourth-order representation proposed by Wallscheid et al. [40]. However, their LPTN is parameter-varying (PV), since the convection thermal resistances surrounding the PM depend on the motor speed, and the conduction thermal resistance between the coolant and stator yoke depends on the coolant temperature. In contrast, the Hammerstein approach requires an LTI model and not a PV model. To generate an LTI representation of the LPTN and to initialize the linear block of the Hammerstein model, the methodology presented in [29] was used to perform the real-time implementation of the Hammerstein model. Varying thermal resistances of the LPTN can be assumed to be fixed, but the speed-dependent effect in thermal dynamics of the motor must be neglected. This affects the temperature estimation accuracy of the tool. To generate an LTI model while considering the effect of the varying thermal resistance, Martinez-Ríos et al. proposed to model the varying thermal resistance of the LPTN as a temperature source [29]. The above is illustrated in the thermal circuit diagram of Fig. 4. The LPTN includes four main domains: the stator yoke (y), stator winding (w), stator tooth (t), and rotor PM (m). There are two additional external elements: the coolant (c) and the surrounding air (a). Temperatures (ϑ), heat sources (P), thermal capacitances (C) and resistances (R) are the variables and parameters of interest.

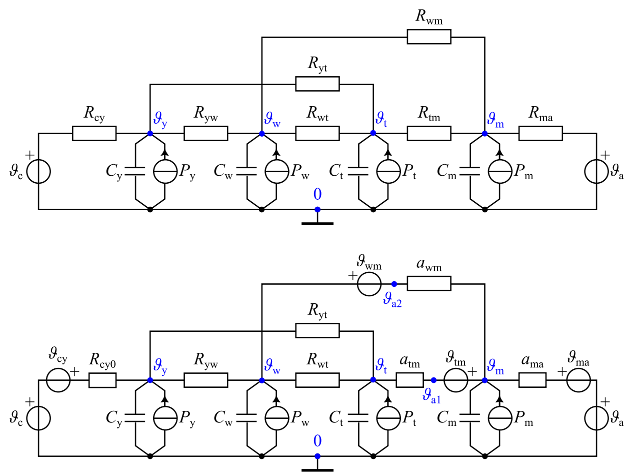


FIGURE 4. Fourth-order lumped-parameter thermal network of the PMSM: original model from [40] (top), modified model (bottom) where temperature sources in series with fixed thermal resistances replace the varying resistance components of the PV LPTN. All 22 identified parameter values and their units are listed in Table 1.

To construct a time-invariant representation of the LPTN, the energy balance of each node of the thermal circuit is considered according to

$$C_i \frac{d\vartheta_i(t)}{dt} = \sum_{j \neq i} \frac{\vartheta_j(t) - \vartheta_i(t)}{R_{ij}} + P_i(t) \tag{2}$$

where i denotes the node under consideration and j denotes the remaining nodes connected to it in the thermal network. The leftmost term represents the rate of change of thermal energy stored in the ith node, where C_i is the thermal capacitance and $\vartheta_i(t)$ is the temperature of the node at time t. The summation term describes the net conductive heat exchange between the ith node and all other nodes j in the network, where R_{ij} is the thermal resistance between nodes i and j, and $\vartheta_j(t)$ is the temperature of the jth node. Finally, $P_i(t)$ denotes the power loss applied to the ith node at time t. This allows the generation of a state-space representation of the form given in (19)-(25), as shown at the bottom of the next page.

This definition ensures that the state-transition matrix A has full rank independent of the values of the thermal capacitances or thermal resistances obtained from the identification process, which avoids the generation of null eigenvalues in the continuous domain and the generation of a marginally stable LPTN representation.

The model inputs are the power losses in the stator and PM, the ambient and coolant temperatures, and the added auxiliary temperature sources. The outputs are the temperatures in the stator yoke, tooth, winding, and PM. The power losses in each of the nodes of the thermal circuit were estimated from look-up tables computed by employing finite-element models in the torque-speed plane. The winding losses were separated into alternate current ($P_{\mathrm{w,ac}}$) and direct current ($P_{\mathrm{w,dc}}$) components according to

$$P_{\mathrm{w,dc}}(\vartheta_{\mathrm{w}}) = P_{\mathrm{w0,dc}}(1 + \alpha_{\mathrm{Cu}}(\vartheta_{\mathrm{w}} - \vartheta_{\mathrm{ref}})), \tag{3}$$

$$P_{\mathrm{w,ac}}(\vartheta_{\mathrm{w}}) = \frac{P_{\mathrm{w0,ac}}}{1 + \alpha_{\mathrm{Cu}}(\vartheta_{\mathrm{w}} - \vartheta_{\mathrm{ref}})}, \tag{4}$$

$$P_{\mathrm{w}} = P_{\mathrm{w,dc}}(\vartheta_{\mathrm{w}}) + P_{\mathrm{w,ac}}(\vartheta_{\mathrm{w}}) \tag{5}$$

where $\alpha_{\mathrm{Cu}} = 0.39\,\%/\mathrm{K}$. Then, stator losses P_{s} were divided into yoke and tooth components by considering their dependence on normalized current, $I/I_{\max}$, and normalized speed, $\omega/\omega_{\max}$ [40]:

$$P_{\mathrm{y}} = k_{\mathrm{s}}(I, \omega)\, P_{\mathrm{s}}, \tag{6}$$

$$P_{\mathrm{t}} = (1 - k_{\mathrm{s}}(I, \omega))\, P_{\mathrm{s}}, \tag{7}$$

$$k_{\mathrm{s}}(I, \omega) = k_0 + k_1 \frac{I}{I_{\max}} + k_2 \frac{\omega}{\omega_{\max}} + k_3 \frac{I}{I_{\max}} \frac{\omega}{\omega_{\max}}. \tag{8}$$

The temperature dependence of the iron losses was not modeled. However, hysteresis and eddy-current losses can vary with temperature, flux density and frequency [41]. Since the stator yoke and tooth temperatures were not measured, the dependence could not be identified or validated independently. Thus, its omission is a modeling limitation that may contribute to temperature-estimation error, particularly at working points where iron losses are significant.

Temperature nodes ϑ_{a1} and ϑ_{a2}, ϑ_{wm}, ϑ_{tm}, ϑ_{ma}, and ϑ_{cy}, are all computed using the following expressions. Further details on the derivation of these expressions are provided in [29].

$$\vartheta_{\mathrm{a1}} = \frac{\vartheta_{\mathrm{m}} + \frac{R_{\mathrm{tm1}}\vartheta_{\mathrm{t}}}{a_{\mathrm{tm}}}}{1 + \frac{R_{\mathrm{tm1}}}{a_{\mathrm{tm}}}}, \tag{9}$$

$$\vartheta_{\mathrm{a2}} = \frac{\vartheta_{\mathrm{w}} + \frac{R_{\mathrm{wm1}}\vartheta_{\mathrm{m}}}{a_{\mathrm{wm}}}}{1 + \frac{R_{\mathrm{wm1}}}{a_{\mathrm{wm}}}}, \tag{10}$$

$$\vartheta_{\mathrm{wm}} = \left(\frac{R_{\mathrm{wm1}}}{a_{\mathrm{wm}}}\right)\left(\frac{\vartheta_{\mathrm{w}} - \vartheta_{\mathrm{m}}}{1 + \frac{R_{\mathrm{wm1}}}{a_{\mathrm{wm}}}}\right), \tag{11}$$

$$\vartheta_{\mathrm{tm}} = \left(\frac{R_{\mathrm{tm1}}}{a_{\mathrm{tm}}}\right)\left(\frac{\vartheta_{\mathrm{m}} - \vartheta_{\mathrm{t}}}{1 + \frac{R_{\mathrm{tm1}}}{a_{\mathrm{tm}}}}\right), \tag{12}$$

$$\vartheta_{\mathrm{ma}} = \left(\frac{R_{\mathrm{ma1}}}{a_{\mathrm{ma}}}\right)\left(\frac{\vartheta_{\mathrm{m}} - \vartheta_{\mathrm{a}}}{1 + \frac{R_{\mathrm{ma1}}}{a_{\mathrm{ma}}}}\right), \tag{13}$$

$$\vartheta_{\mathrm{cy}} = \left(\frac{R_{\mathrm{cy01}}}{R_{\mathrm{cy0}}}\right)\left(\frac{\vartheta_{\mathrm{y}} - \vartheta_{\mathrm{c}}}{1 + \frac{R_{\mathrm{cy01}}}{R_{\mathrm{cy0}}}}\right). \tag{14}$$

The varying convection thermal resistances R_{wm}, R_{tm}, R_{ma} are computed according to

$$R_{i,j}(\omega) = R_{i,j,0}\, e^{-\frac{\omega}{\omega_{\max}} \frac{1}{b_{i,j}}} + a_{i,j} \tag{15}$$

where ω is the motor speed, and $R_{i,j,0}$, $b_{i,j}$, and $a_{i,j}$ are constant parameters. The exponential term is used to compute the values of R_{wm1}, R_{tm1}, and R_{ma1}. The parameter $a_{i,j}$ is used to compute the values of the resistances a_{wm}, a_{tm}, and a_{ma}, as illustrated in Fig. 4. In addition, the varying thermal conduction resistance between the coolant and stator yoke is computed according to the following expression:

$$R_{\mathrm{cy}} = R_{\mathrm{cy0}}\left[1 + \alpha_{\mathrm{cy}}\left(\vartheta_{\mathrm{c}} - \vartheta_{\mathrm{c0}}\right)\right] \tag{16}$$

where $\vartheta_{\mathrm{c0}} = 40°\mathrm{C}$ is the reference coolant temperature. The term $R_{\mathrm{cy01}} = R_{\mathrm{cy0}}\alpha_{\mathrm{cy}}(\vartheta_{\mathrm{c}} - \vartheta_{\mathrm{c0}})$ is the temperature-dependent contribution to R_{cy} such that $R_{\mathrm{cy}} = R_{\mathrm{cy0}} + R_{\mathrm{cy01}}$. Finally, the state matrices of the LPTN model presented in (20)–(25) consider as inputs the temperatures ϑ_{cv} and ϑ_{av}, which can be computed through

$$\vartheta_{\mathrm{cv}} = \vartheta_{\mathrm{c}} + \vartheta_{\mathrm{cy}}, \tag{17}$$
$$\vartheta_{\mathrm{av}} = \vartheta_{\mathrm{a}} + \vartheta_{\mathrm{ma}} \tag{18}$$

The proposed LTI representation of the LPTN presents constant parameters according to (20)–(25). The added temperature sources account for the effect of varying the thermal resistance, and their values are discretized according to the period at which the motor speed and coolant temperature are sampled. Thus, discretization must be performed once, possibly offline. In contrast, the LPTN proposed by Wallscheid et al. [40] required an online discretization of the LPTN model each time a change in the thermal resistance values occurs according to the motor's speed or coolant temperature.

D. IDENTIFICATION OF HAMMERSTEIN MODEL

Particle swarm optimization was used to estimate the 22 parameters of the fourth-order LPTN model, following the methodology presented in [29] and [40]. Despite the proposed LPTN having four additional inputs and being expressed in an LTI form, the model's fitting process does not differ from that presented in the related research. A swarm size of 220 and 100 iterations were selected to estimate the parameters of the LPTN. The maximum likelihood function between measured and estimated outputs—temperature values of the winding and PM—was set as the cost function to minimize. The LPTN model was identified by employing the profiles with the coolant at temperatures of 20 °C and 60 °C, and the model was validated with the profiles with the coolant inlet temperature at 40 °C. The identified LPTN parameters are listed in Table 1. The reader is referred to [29] for further details regarding the estimation process of the LPTN.

The continuous-time eigenvalues of the LTI LPTN model can be computed based on the identified parameters by substituting their values presented in Table 1 in (19). The eigenvalues λ of matrix $\mathbf{A}$ are computed as follows:

$$\det(\lambda\mathbf{I} - \mathbf{A}) = 0 \tag{26}$$

which yields stable, real eigenvalues in continuous time:

$$\begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\\ \lambda_4\end{bmatrix} = \begin{bmatrix}-0.1320\\ -0.0254\\ -0.0120\\ -0.0045\end{bmatrix} \tag{27}$$

Following the estimation of the LPTN parameters, the model was expressed in the state space in (19) and discretized through the zero-order hold and the Tustin methods. The zero-order hold method provides exact time-domain discretization by assuming that the inputs are constant between samples. On the other hand, the Tustin method provides the best frequency-domain matching between the continuous and discrete models [42]. The purpose of testing different discretization methods is to assess their influence on the numerical stability of the linear block of the Hammerstein model following its parametrization. Each discretization

$$\begin{cases}\dot{x} = Ax + Bu\\ y = Cx + Du\end{cases} \tag{19}$$

$$A = \begin{bmatrix} -\frac{1}{C_{\mathrm{y}}}\left(\frac{1}{R_{\mathrm{yw}}} + \frac{1}{R_{\mathrm{yt}}} + \frac{1}{R_{\mathrm{cy0}}}\right) & \frac{1}{C_{\mathrm{y}}R_{\mathrm{yw}}} & \frac{1}{C_{\mathrm{y}}R_{\mathrm{yt}}} & 0\\ \frac{1}{C_{\mathrm{w}}R_{\mathrm{yw}}} & -\frac{1}{C_{\mathrm{w}}}\left(\frac{1}{R_{\mathrm{yw}}} + \frac{1}{R_{\mathrm{wt}}}\right) & \frac{1}{C_{\mathrm{w}}R_{\mathrm{wt}}} & \frac{1}{C_{\mathrm{w}}a_{\mathrm{wm}}}\\ \frac{1}{C_{\mathrm{t}}R_{\mathrm{yt}}} & \frac{1}{C_{\mathrm{t}}R_{\mathrm{wt}}} & -\frac{1}{C_{\mathrm{t}}}\left(\frac{1}{R_{\mathrm{yt}}} + \frac{1}{R_{\mathrm{wt}}} + \frac{1}{a_{\mathrm{tm}}}\right) & 0\\ 0 & 0 & \frac{1}{C_{\mathrm{m}}a_{\mathrm{tm}}} & -\frac{1}{C_{\mathrm{m}}}\left(\frac{1}{a_{\mathrm{wm}}} + \frac{1}{a_{\mathrm{ma}}}\right) \end{bmatrix} \tag{20}$$

$$B = \begin{bmatrix} \frac{1}{C_{\mathrm{y}}} & 0 & 0 & 0 & \frac{1}{C_{\mathrm{y}}R_{\mathrm{cy0}}} & 0 & 0 & 0\\ 0 & \frac{1}{C_{\mathrm{w}}} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{C_{\mathrm{w}}a_{\mathrm{wm}}}\\ 0 & 0 & \frac{1}{C_{\mathrm{t}}} & 0 & 0 & 0 & \frac{1}{C_{\mathrm{t}}a_{\mathrm{tm}}} & 0\\ 0 & 0 & 0 & \frac{1}{C_{\mathrm{m}}} & 0 & \frac{1}{C_{\mathrm{m}}a_{\mathrm{ma}}} & -\frac{1}{C_{\mathrm{m}}a_{\mathrm{tm}}} & \frac{1}{C_{\mathrm{m}}a_{\mathrm{wm}}} \end{bmatrix} \tag{21}$$

$$x = \left[\vartheta_{\mathrm{y}}\ \vartheta_{\mathrm{w}}\ \vartheta_{\mathrm{t}}\ \vartheta_{\mathrm{m}}\right]^{\top} \tag{22}$$
$$u = \left[P_{\mathrm{y}}\ P_{\mathrm{w}}\ P_{\mathrm{t}}\ P_{\mathrm{m}}\ \vartheta_{\mathrm{cv}}\ \vartheta_{\mathrm{av}}\ \vartheta_{\mathrm{a1}}\ \vartheta_{\mathrm{a2}}\right]^{\top} \tag{23}$$
$$C = I_{4\times 4} \tag{24}$$
$$D = 0_{4\times 8} \tag{25}$$

TABLE 1. Parameter values estimated for the LPTN model. Data taken from [29].

Parameter	Value	Unit	Parameter	Value	Unit
C_{m}	6652	J/K	R_{cy0}	0.004	K/W
C_{t}	3043	J/K	R_{tm0}	0.484	K/W
C_{w}	5296	J/K	R_{wm0}	0.665	K/W
C_{y}	3340	J/K	R_{wt}	0.436	K/W
a_{ma}	0.036	K/W	R_{yt}	0.015	K/W
b_{ma}	-0.176	–	R_{yw}	0.009	K/W
R_{ma0}	0.314	K/W	a_{tm}	0.094	K/W
a_{wm}	0.492	K/W	b_{tm}	0.200	–
b_{wm}	0.968	–	k_0	0.775	–
k_1	-0.080	–	k_2	-0.008	–
k_3	-0.115	–	α_{cy}	-0.009	1/K

technique was applied by considering 0.5, 1, 2, and 4 seconds as sampling periods. Similar to the discretization methods, different sampling periods were tested to assess their influence on the numerical stability of the Hammerstein model.

The thermal time constants can be computed from the stable continuous-time real poles p_i of the linear LPTN model as $\tau_i = -p_i^{-1}$. The resulting thermal time constants are 7.57, 39.42, 83.18, and 223.83 s. Relative to the fastest mode of 7.57 s, sampling periods of 0.5, 1, 2, and 4 s correspond to 15.15, 7.57, 3.79, and 1.89 sampling intervals per time constant, respectively. Thus, the 0.5- and 1-s periods provide the finest temporal resolution, the 2-s period provides intermediate resolution, and the 4-s period resolves the fastest mode only coarsely. Moreover, because a threshold temperature crossing immediately after an estimator may remain undetected until the next update, the sampling-induced detection delay can approach the sampling period. However, since no specific maximum detection time was set, the results provide a relative sampling-time tradeoff but not compliance with a thermal-protection requirement.

The nonlinear input functions employed in the Hammerstein model are represented by single-neuron sigmoid networks. The sigmoid network is defined as

$$S(X) = \sum_{i=1}^{n} s_i f\left(X^\top Q b_i + c_i\right) \tag{28}$$

where $S(X)$ denotes the output of the sigmoid network, $X \in \mathbb{R}^m$ is the input vector, and n is the number of sigmoid units (neurons). The coefficients $s_i \in \mathbb{R}$ are output weights. The matrix $Q \in \mathbb{R}^{m\times q}$ is a projection matrix with $m \geq q$, while $b_i \in \mathbb{R}^q$ are dilation vectors and $c_i \in \mathbb{R}$ are translation scalars. The activation function $f(\cdot)$ is a sigmoid:

$$f(z) = \frac{1}{1+e^{-z}}, \tag{29}$$

where

$$z = X^\top Q b_i + c_i. \tag{30}$$

The one-neuron sigmoid networks used as the input nonlinearity were applied only to the inputs associated with the estimated power losses of the LPTN model, whereas the temperature source inputs were fed directly to the linear thermal network. This design was selected to introduce the minimum nonlinear complexity required to capture the nonlinear relationship between the estimated power losses, obtained from finite element simulations and lookup tables, and the thermal dynamics of the motor. Since the underlying fourth-order LPTN already provides an accurate description of the linear thermal behavior, incorporating a more complex nonlinear mapping would unnecessarily increase the number of parameters and the risk of overfitting without a corresponding improvement in model fidelity. The use of a single-neuron sigmoid network preserves the sparsity of the overall Hammerstein-LPTN model while providing a minimal, smooth, and bounded nonlinear mapping for the power-loss inputs. Since alternative nonlinear blocks were not evaluated, this architecture is not claimed to be optimal. During the estimation process, the poles of the discrete LPTN model were kept fixed, and the inputs related to the power losses were normalized before being processed by the sigmoid nonlinearities. Figure 5 illustrates the architecture of the LTI LPTN model with the incorporation of the Hammerstein model.

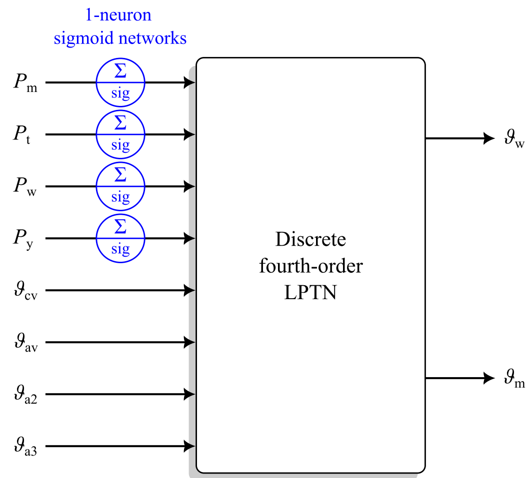


FIGURE 5. Structure of the Hammerstein model incorporating the discrete LTI LPTN. This figure was adapted from [29].

The fitting process consisted of 100 iterations by employing the Levenberg-Marquardt algorithm. Like the LPTN model, the Hammerstein model was estimated using the profiles with the coolant inlet temperatures at 20 °C and 60 °C and validated with the 40 °C profile. The identified Hammerstein-based LPTN model is valid specifically within the thermal conditions defined by the driving cycles and coolant temperatures. Consequently, the model is intended to operate within these boundaries, and the results presented in this work are limited to this operational envelope. Additionally, the 40 °C validation dataset was employed to evaluate interpolation at an intermediate coolant-temperature condition not included in the identification data, under the same driving-cycle sequence. It is essential to mention that the MATLAB R2023b implementation transforms the LTI model used to initialize the Hammerstein model to a polynomial form for fitting the Hammerstein model as described in (1). Nevertheless, the above could lead to a numerically unstable model, making the model unfit for truncation and round-off errors because polynomial form models are inherently ill-conditioned representations of LTI systems [43], [44], [45]. The conversion between model types can also introduce numerical errors in the model, especially for multiple-input multiple-output models, as in the case of the Hammerstein model presented in this work [34]. Hence, ensuring the model's numerical stability is crucial to maintaining adequate performance when converting its parameters from double to single precision.

The following section provides an overview of the metrics and methods used to analyze and improve the numerical stability of the Hammerstein model, particularly its linear block, and embed it into a 32-bit microcontroller.

E. NUMERICAL STABILITY ANALYSIS

Numerical stability refers to the extent to which an algorithm maintains accuracy when subject to rounding errors or small perturbations in the input data [46]. The numerical stability of the Hammerstein model was evaluated by analyzing the eigenvalues and condition number of the discrete state-transition matrix of the LTI block of the Hammerstein model after being fitted. To achieve this, the polynomial form of the linear block in the Hammerstein model was transformed into state-space form. The eigenvalues of the model were computed to determine the model's stability and analyze how the poles are affected by truncation errors when transforming the model parameters from double to single precision to embed the model in a 32-bit microcontroller. Based on the calculated condition number, the model was balanced to improve the condition of the state-transition matrix and minimize truncation errors when converting from double to single precision.

1) CONDITION NUMBER

The condition number of the state-transition matrix was computed to assess the numerical stability of the discrete Hammerstein model. The condition number of a matrix measures its sensitivity to input perturbations and round-off errors [47]. For a nonsingular matrix A, it is defined as

$$\kappa(A) = \|A\| \cdot \|A^{-1}\| \tag{31}$$

i.e., the scalar product between the norm of matrix A and the norm of its inverse.

A condition number close to one indicates a well-conditioned matrix, while a higher condition number signifies an ill-conditioned matrix [48]. The computation of the condition number is crucial to determine if the obtained discrete state-space representation will maintain its accuracy when the values of the state-space matrix are truncated to a 32-bit representation (i.e., single precision) to implement the model on an embedded processor. In particular, determining whether the discrete state-transition matrix is well-conditioned is important for characterizing the numerical conditioning of the implemented realization. However, its condition number alone neither guarantees unchanged eigenvalues after single-precision coefficient conversion nor implies that an ill-conditioned matrix will necessarily exhibit drastic eigenvalue shifts; these effects are therefore assessed directly through the double- and single-precision pole comparisons in Tables 2 and 4.

2) BALANCED STATE-SPACE REALIZATION

A balanced state-space realization, also known as a balancing transformation, is a specific form of state-space representation for LTI systems. In this form, the controllability and observability Gramians are equal and diagonal. This allows for balancing the input-to-state and state-to-output energy transfer. It is worth mentioning that balanced realization can only be applied to stable and minimal LTI systems. In particular, the system must be asymptotically stable, controllable, and observable such that the controllability and observability Gramians exist and are positive definite [49], [50].

To compute a balanced state representation, consider a generic state-space representation according to (19), where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is the input vector, $y \in \mathbb{R}^p$ is the output vector, $A \in \mathbb{R}^{n\times n}$ is the state-transition matrix, $B \in \mathbb{R}^{n\times m}$ is the input matrix, $C \in \mathbb{R}^{p\times n}$ is the output matrix, and $D \in \mathbb{R}^{p\times m}$ is the feedthrough matrix. The pair (A, B) is assumed controllable and the pair (A, C) is assumed observable. Furthermore, it is assumed that the controllability Gramian W_c and observability Gramian W_o satisfy the algebraic Lyapunov equations [50]:

$$AW_c + W_cA^\top + BB^\top = 0, \tag{32}$$

$$A^\top W_o + W_oA + C^\top C = 0. \tag{33}$$

To obtain a balanced realization, a transformation $T \in \mathbb{R}^{n\times n}$ is sought such that the controllability and observability Gramians are equal and diagonal:

$$\hat{W}_c = \hat{W}_o = \Sigma \tag{34}$$

where Σ is a diagonal matrix containing the Hankel singular values, $\hat{W}_o = T^*W_oT$, and $\hat{W}_c = T^{-1}W_c(T^{-1})^*$. Multiplying the transformed controllability and observability Gramians yields

$$\hat{W}_c\hat{W}_o = T^{-1}W_cW_oT = \Sigma^2 \tag{35}$$

The product of (35) with T leads to

$$W_cW_oT = T\Sigma^2 \tag{36}$$

Consequently, the balancing transformation T can be computed by the eigendecomposition of the product of the controllability and observability Gramians [51]. This expression remains valid irrespective of any scaling applied to the eigenvectors. Thus, an adequate scaling should be selected to balance the Gramians correctly. The above implies that there are many transformations T that satisfy $\hat{W}_c\hat{W}_o = \Sigma^2$, but where the individual Gramians are different. The transformation T provides diagonalized observability and controllability Gramians but not necessarily equal, as denoted in the following expressions:

$$\hat{W}_o = T^* W_o T = \Sigma_o^2 \tag{37}$$

$$\hat{W}_c = T^{-1} W_c (T^{-1})^* = \Sigma_c^2 \tag{38}$$

To make Gramians equal and diagonal, the resultant eigenvectors of the products of Gramians should be scaled by a factor [51]. This scaling factor is defined as follows:

$$\Sigma_s = \Sigma_c^{1/2}\Sigma_o^{-1/2} \tag{39}$$

The scaled transformation T_s is defined as $T_s = T\Sigma_s$, and a balanced state-space realization of the system can be computed as

$$A_{bal} = T_s^{-1} A T_s, \tag{40}$$

$$B_{bal} = T_s^{-1} B, \tag{41}$$

$$C_{bal} = C T_s. \tag{42}$$

With the treatment of continuous-time systems having been discussed, the balanced state-space realization is addressed for discrete-time systems. First, consider a discrete-time state-space model of the form

$$x[k+1] = A_k x[k] + B_k u[k], \tag{43}$$

$$y[k] = C_k x[k] + D_k u[k] \tag{44}$$

where A_k is the discrete state-transition matrix, B_k is the input matrix, C_k is the discrete output matrix, and $D_k = 0$ is the feed-through matrix. The pairs (A_k, B_k) and (A_k, C_k) are both assumed to be controllable and observable, and the controllability and observability Gramians satisfy the following Lyapunov equations:

$$A_k W_c A_k^\top - W_c + B_k B_k^\top = 0, \tag{45}$$

$$A_k^\top W_o A_k - W_o + C_k^\top C_k = 0. \tag{46}$$

Notice that the equations defining the controllability and observability Gramians differ from their continuous-time counterparts (see 32 and 33). Nevertheless, the computation of T is performed by calculating the eigendecomposition of the product of Gramians as in the continuous-time domain, and T should be scaled to ensure that the Gramians are equal and diagonal.

This study applied a balanced state-space realization to the state-space representation of the LTI component of the Hammerstein model, following its transformation from polynomial form. This approach was used to improve the condition number of the state-transition matrix and minimize numerical errors that could arise when truncating the model coefficients for implementation on a 32-bit platform. As noted in [44], [52], and [53], balanced state-space realizations are numerically well-conditioned compared to companion form and canonical controllable/observable forms, making them better suited for precise computations in constrained hardware platforms.

Besides, simple state scaling cannot fully resolve numerical stability issues associated with an ill-conditioned state-transition matrix in LTI systems. State scaling applies a similarity transformation to the state matrix,

$$A_s = SAS^{-1}, \tag{47}$$

where S is a nonsingular scaling matrix; this transformation changes only the coordinate representation of the state variables while preserving the system dynamics, eigenvalues, controllability, and observability properties.

The numerical sensitivity of the state matrix can be characterized by its condition number,

$$\kappa(A_s) = \|A_s\| \, \|A_s^{-1}\|, \tag{48}$$

where a large value of $\kappa(A_s)$ indicates that small perturbations in the matrix coefficients or numerical operations may result in significant errors in the computed state response. Although state scaling can reduce disparities in the magnitude of state variables, it does not modify the underlying coupling between states. Therefore, if the original realization contains strongly correlated state directions or poorly observable/controllable modes, the transformed matrix may remain close to singular, causing $\|A_s^{-1}\|$ to remain large and limiting improvements in numerical conditioning.

In contrast, balanced realization provides a systematic transformation of the state-space representation based on the controllability and observability Gramians.

Unlike canonical realizations obtained directly from polynomial representations, such as controllable or observable canonical forms, balanced realizations produce a state-space representation in which states are ordered according to their dynamical importance and have comparable controllability and observability energies. This reduces numerical disparities among state variables, improves matrix conditioning, and provides a more robust representation for numerical simulation and model reduction.

Therefore, balanced realization does not alter the physical dynamics of the LTI system; instead, it identifies a numerically favorable coordinate system that preserves the dominant dynamic behavior while avoiding the poor scaling and hidden dependencies often present in canonical state-space representations.

F. HARDWARE IMPLEMENTATION OF THE HAMMERSTEIN MODEL

The Hammerstein model was tested by performing a Processor-in-the-Loop (PIL) simulation through MATLAB-Simulink R2023b. PIL is a type of hardware-in-the-loop (HIL) testing that consists of executing the designed code, model, or controller on the target processor to validate its performance. PIL is primarily used to test deployability in embedded systems. This ensures that the designed controller or algorithm will behave correctly when executed on the target processor for the final application [54]. The PIL test allows estimating the average execution time of the model and its CPU utilization ratio with respect to the model sampling period. This study tested the model through PIL on a Texas Instruments C2000 LaunchPadXL TMS320F28069M, a 32-bit inverter-grade microcontroller easily found in automotive and industrial electric machine drives. This platform has a 90 MHz clock; it natively supports single-precision floating-point operations. This microcontroller has 256 KB of flash memory and 100 KB of RAM.

Since MATLAB requires double-precision data to estimate the Hammerstein model, converting these into single precision is mandatory to accommodate the limitations of the selected processor. However, to avoid truncation errors that can change the fitted model's dynamic behavior, the LTI state matrix needs to be well-conditioned. In this respect, the balancing transformation of the linear block of the Hammerstein model is used to improve the condition number of the state-transition matrix and avoid significant truncation errors when transforming the model parameters from double to single precision.

The PIL simulation was also used to compare the execution time of the Hammerstein model to that of the LPTN model. The LPTN model PIL simulation was implemented using two approaches: (i) LTI LPTN as presented in this work, and (ii) a PV LPTN. As stated previously, the LTI model presented in (20) to (25) only required a single offline discretization. This step was performed for zero-order hold and Tustin methods at 4, 2, 1, and 0.5 seconds of sampling period. It is worth noting that, when used alone, the discrete LTI LPTN does not require balancing. However, when implemented within the Hammerstein model, the linear block is transformed from a state-space form to a transfer function representation and then back to state-space representation, introducing numerical errors to the model [44].

Additionally, the following considerations were taken into account when discretizing the PV LPTN. First, the state-transition matrix A was discretized according to:

$$A_\mathrm{d} = e^{A\Delta t} \tag{49}$$

where A_d is the discrete state-transition matrix and Δt is the sampling period.

The exact discretization of the input matrix of a continuous-time state-space model can be computed based on

$$B_\mathrm{d} = \int_0^{\Delta t} e^{A\tau} B \, d\tau. \tag{50}$$

where B_d is the discrete input matrix.

Nonetheless, the PV LPTN model requires discretization each time the scheduling parameters (i.e., the motor speed and coolant temperature) change, but computing the exact discretization is computationally demanding. According to the suggestion in [40], the PV LPTN was discretized with a first-order Euler approximation for A_d and B_d. The above led to the following expressions [55]:

$$\begin{aligned} A_\mathrm{d} &= I + \Delta t A, \\ B_\mathrm{d} &= \Delta t B. \end{aligned} \tag{51}$$

In this case, the continuous-time state-space matrices of the PV LPTN should be formulated according to the details provided in [40] for the LPTN provided in Fig. 4a.

III. RESULTS AND DISCUSSION

A. NUMERICAL STABILITY ANALYSIS

The numerical stability of the fitted model was measured by computing the condition number of the state-transition matrix of the LTI block in the Hammerstein model after fitting the model with MATLAB's System Identification Toolbox. The state-transition matrix was computed by transforming the polynomial model to state-space form. Furthermore, the eigenvalues of the model were calculated by testing the model at different sample periods, using the Tustin and zero-order hold discretization methods, and truncating the values of the state-transition matrix parameters from a double-precision to a single-precision representation.

The eigenvalues of the state-transition matrix before balancing are presented in Table 2 for each discretization method and sampling period using double- and single-precision representations. One can observe significant changes in the eigenvalues, especially when changing from double precision to single precision. In the worst cases, the reported eigenvalues are unstable, as their magnitude goes slightly beyond the discrete unit-circle locus. These differences can introduce errors in the fitted model dynamics, as appreciated in Fig. 6, where the model behavior is disrupted from what was initially expected.

TABLE 2. Eigenvalues for the LTI LPTN used within the Hammerstein model for different discretization methods and sampling periods in single and double precision.

Discretization	Precision	Δt [s]	Eigenvalues			
Zero-order hold	Single	0.5	1.0085	0.9864	0.9864	0.9343
		1	1.0095	0.9751	0.9751	0.8755
		2	0.9950	0.9660	0.9570	0.7678
		4	0.9812	0.9548	0.9028	0.5897
	Double	0.5	0.9978	0.9940	0.9874	0.9361
		1	0.9955	0.9880	0.9749	0.8763
		2	0.9911	0.9762	0.9505	0.7679
		4	0.9823	0.9530	0.9035	0.5897
Tustin	Single	0.5	1.0123	0.9845	0.9845	0.9344
		1	1.0018	1.0018	0.9537	0.8779
		2	0.9855	0.9855	0.9468	0.7668
		4	0.9825	0.9527	0.9036	0.5822
	Double	0.5	0.9978	0.9940	0.9874	0.9361
		1	0.9955	0.9880	0.9749	0.8761
		2	0.9911	0.9762	0.9505	0.7667
		4	0.9823	0.9530	0.9034	0.5822

When comparing the behavior at increasing sampling steps, the difference between single- and double-precision eigenvalues diminishes (see Table 2). However, increasing the sampling period reduces the number of samples available to reconstruct the machine's thermal response, which may introduce bias into the temperature estimation. Analyzing the

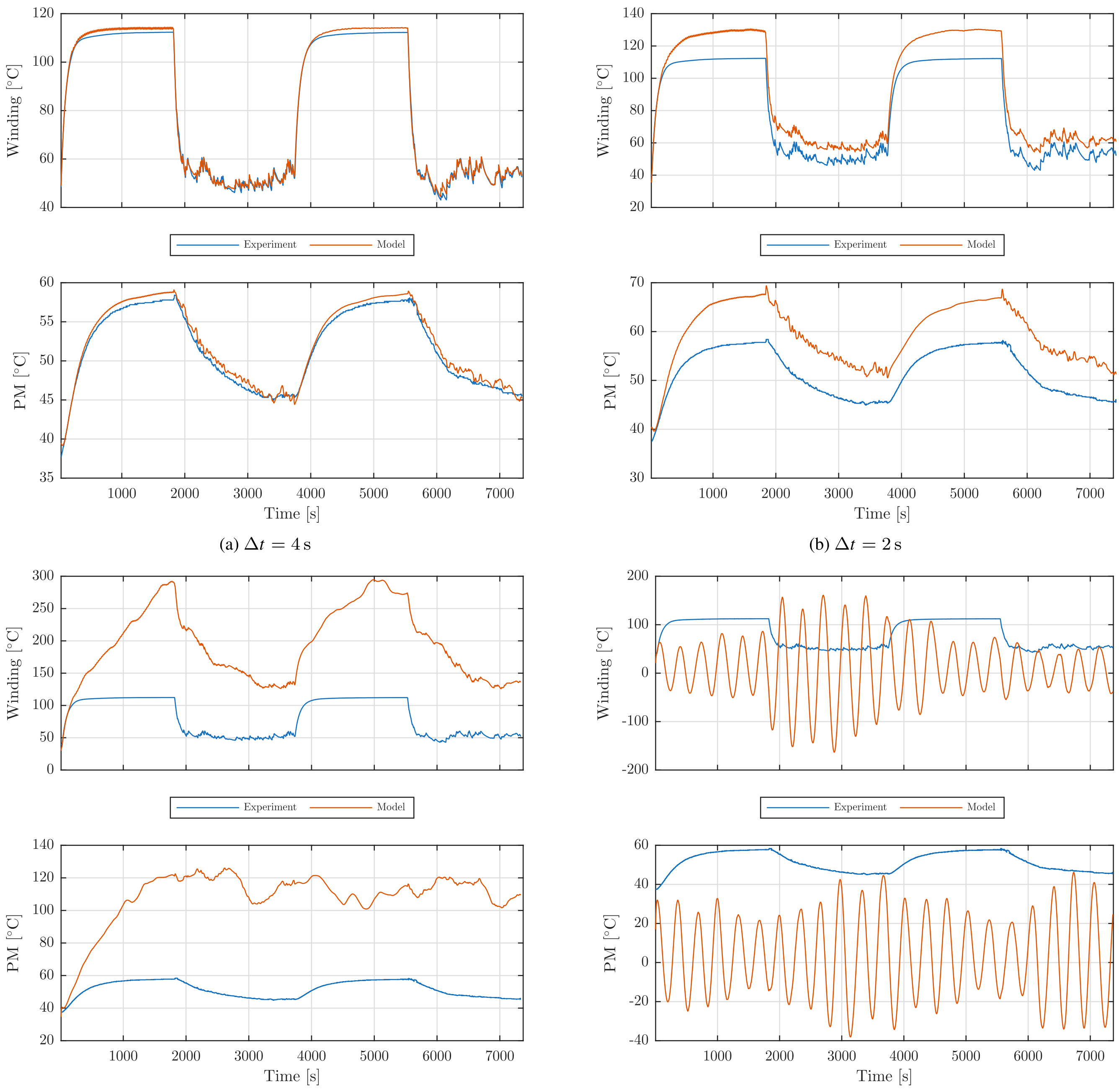


FIGURE 6. Temperature estimation of the Hammerstein model against experimental reference for different sampling periods before balancing the linear block of the model. Temperatures at the winding (top panel) and the PM (bottom panel) are shown.

response of the single-precision Hammerstein model fitted with a four-second sampling period reveals that the response does not significantly differ from the single-precision representations obtained with two- and one-second sampling periods. This outcome is attributed to the fact that, with higher sampling periods, the system's discrete poles become more widely spaced, as shown in Fig. 6. Nevertheless, a large sampling period also reduces the model's responsiveness in estimating the machine's component temperatures, which could hinder timely corrective actions to prevent thermal overloading during motor operation.

The deviations in the resultant model dynamics and eigenvalues are related to the numerical instability of the linear block of the Hammerstein model. Table 3 compares the condition number of the state-transition matrix by discretizing the model using the zero-order hold and Tustin methods and different sampling periods. The condition number of the state-transition matrix was computed before

TABLE 3. LPTN state-matrix condition numbers using zero-order hold and Tustin discretization techniques and different sampling periods before and after balancing.

Discretization	Δt [s]	Condition number	
		Original	Balanced
Zero-order hold	0.5	32.6076	1.0711
	1	32.7589	1.1457
	2	33.3686	1.3133
	4	35.8775	1.7202
Tustin	0.5	32.6076	1.0709
	1	32.7595	1.1464
	2	33.3773	1.32
	4	36.0291	1.7474

TABLE 4. Eigenvalues for the balanced LTI LPTN used within the Hammerstein model for different discretization methods and sampling periods in single and double precision.

Discretization	Precision	Δt [s]	Eigenvalues			
Zero-order hold	Single	0.5	0.9978	0.9940	0.9874	0.9361
		1	0.9955	0.9880	0.9749	0.8763
		2	0.9911	0.9762	0.9505	0.7679
		4	0.9823	0.9530	0.9035	0.5897
	Double	0.5	0.9978	0.9940	0.9874	0.9361
		1	0.9955	0.9880	0.9749	0.8763
		2	0.9911	0.9762	0.9505	0.7679
		4	0.9823	0.9530	0.9035	0.5897
Tustin	Single	0.5	0.9978	0.9940	0.9874	0.9361
		1	0.9955	0.9880	0.9749	0.8761
		2	0.9911	0.9762	0.9505	0.7667
		4	0.9823	0.9530	0.9034	0.5822
	Double	0.5	0.9978	0.9940	0.9874	0.9361
		1	0.9955	0.9880	0.9749	0.8761
		2	0.9911	0.9762	0.9505	0.7667
		4	0.9823	0.9530	0.9034	0.5822

and after balancing the model. Based on the obtained condition numbers, one can deem the linear block of the Hammerstein model as less well-conditioned than the balanced realization, with values exceeding 32 for different sampling periods and discretization methods. The original realization has condition numbers above 32 and also exhibits substantial pole changes after single-precision coefficient conversion; these pole changes are assessed directly in Table 2. The high condition number is also associated with the state-space model's companion form when converting the Hammerstein model's linear component from its polynomial representation [44].

The condition number of the balanced realization is significantly lower and close to one, as presented in Table 3. Furthermore, increasing the sampling frequency improves the condition number. Table 4 compares the eigenvalues of the state-transition matrix when truncating the model from double to single precision, this time for the balanced case. The results show that the poles remain close and inside the unit circle after coefficient conversion. Eigenvalues remain similar for all sampling periods and discretization methods, which is supported by the improvement in the condition number of the state-transition matrix as presented in Table 3. Furthermore, Fig. 7 shows the model's response after balancing the state-space form of the linear block of the Hammerstein model and converting the model parameters to single precision. The model can track the actual measured data without altering its dynamic behavior, as its eigenvalues remain unchanged after truncation (see Table 4).

The level of applicability of the proposed method must be clarified. The balancing procedure can be extended to stable and minimal LTI block of thermal models developed for other machines or cooling topologies. Nonetheless, the balancing transformation must be recomputed from the state-space of each characterized model, and thus cannot be transferred directly from the motor under study. In particular, the developed Hammerstein model and its derived results are specific to the 200-kW out-runner PMSM, cooling configuration, driving cycles, and coolant-temperature range. Therefore, application to other machine topologies or operating envelopes may need additional experimental data, model reidentification, recomputing of the balancing transformation, and validation. Consequently, this work demonstrates the transferability of the procedure, but not the generalizability of the model to other machines.

1) RELATION TO ALTERNATIVE REALIZATIONS

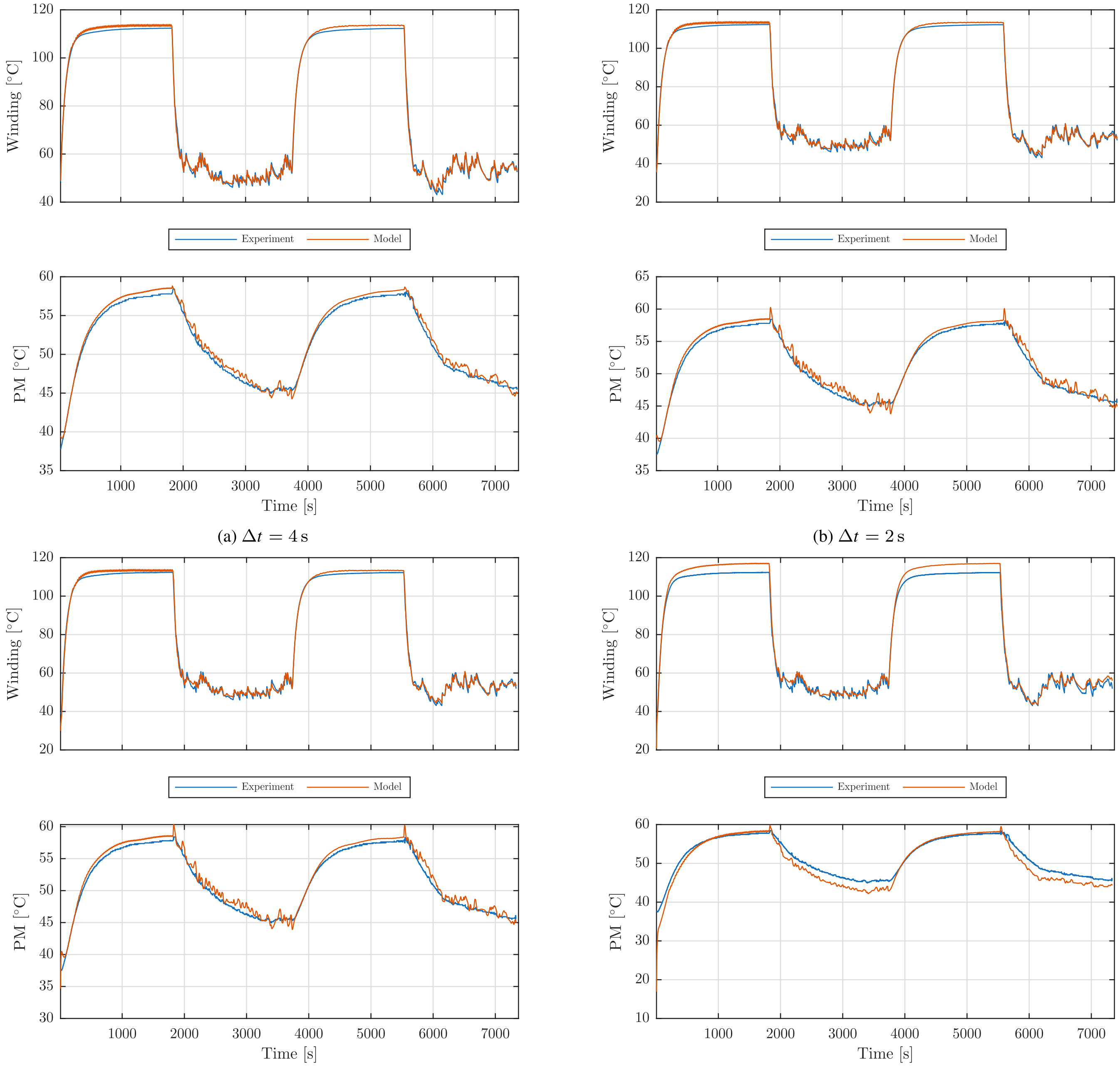
For a nonsingular coordinate transformation $\mathbf{x} = \mathbf{T}\mathbf{z}$:

$$\mathbf{A}_{\mathrm{T}} = \mathbf{T}^{-1}\mathbf{A}\mathbf{T}, \quad \mathbf{B}_{\mathrm{T}} = \mathbf{T}^{-1}\mathbf{B}, \quad \mathbf{C}_{\mathrm{T}} = \mathbf{C}\mathbf{T}$$

and the transfer function and poles are preserved in exact arithmetic. Diagonal state scaling restricts $\mathbf{T}$ to a diagonal matrix and may improve coefficient scaling, but it cannot generally equalize the controllability and observability Gramians. In contrast, for a stable and minimal realization, Gramian balancing determines a full transformation satisfying $W_{\mathrm{c,T}} = W_{\mathrm{o,T}} = \Sigma$, which provides energy-based state scaling and motivates its use in this work. Although the balanced internal states are transformed coordinates, the physical temperature outputs are preserved through the transformed output matrix $\mathbf{C}_{\mathrm{T}}$. For the evaluated LPTN-based Hammerstein implementation, balancing improved the observed finite-precision behavior relative to the original realization. However, Gramian balancing does not guarantee minimum pole sensitivity or superiority over optimized diagonal scaling. Diagonal scaling and direct state-space Hammerstein identification were not evaluated quantitatively. A direct state-space approach could avoid the polynomial conversion and remains a relevant alternative for future comparison.

B. HAMMERSTEIN MODEL PERFORMANCE

The Hammerstein model's fitting capability was evaluated through the mean squared error (MSE), root mean squared error (RMSE), and the normalized root mean squared error (NRMSE) at different sampling periods. These metrics were



(a) $\Delta t = 4\,\mathrm{s}$ (b) $\Delta t = 2\,\mathrm{s}$ (c) $\Delta t = 1\,\mathrm{s}$ (d) $\Delta t = 0.5\,\mathrm{s}$

FIGURE 7. **Temperature estimation of the Hammerstein model against experimental reference for different sampling periods after balancing the linear block of the model. Temperatures at the winding (top panel) and the PM (bottom panel) are shown.**

computed according to the following expressions:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{52}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{53}$$

$$\mathrm{NRMSE} = \frac{1}{\sigma_y}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{54}$$

where N is the total number of data points, y_i represents the experimental reference values, $\hat{y}_i$ represents the estimated values, and σ_y is the standard deviation of the actual measured values. The MSE and RMSE were used as evaluation metrics since they have been used in the related research. Moreover, the RMSE compares performance in the same units as the measured data. Alternatively, NRMSE is a dimensionless metric that standardizes the RMSE by the variability of the measured data, represented by its standard deviation σ_y. This normalization facilitates comparisons among outputs or datasets with different levels of variability.

TABLE 5. Temperature estimation performance on the winding and PM for different methods and sampling periods.

Domain	Method	Δt [s]	MSE [°C^2]	RMSE [°C]	NRMSE [-]
Winding	LPTN	0.5	10.0813	3.1751	0.1103
		1	10.1268	3.1823	0.1106
		2	10.2843	3.2069	0.1114
		4	10.0890	3.1763	0.1104
	Hammerstein	0.5	1.3078	1.1436	0.0397
		1	1.4519	1.2050	0.0419
		2	1.4000	1.1832	0.0411
		4	1.3003	1.1403	0.0396
PM	LPTN	0.5	0.6269	0.7918	0.1584
		1	0.6299	0.7937	0.1588
		2	0.6318	0.7949	0.1590
		4	0.6432	0.8020	0.1604
	Hammerstein	0.5	0.5162	0.7185	0.1438
		1	0.5176	0.7194	0.1439
		2	0.4785	0.6918	0.1384
		4	0.3422	0.5850	0.1170

The fitting of the Hammerstein model was performed by discretizing the linear model via a zero-order hold since it has previously provided the best condition number after applying the balancing transformation when compared to the Tustin method. The performance of the Hammerstein model was compared with that of the LPTN across different sampling periods to evaluate the sensitivity of the aggregate validation performance to the sampling period. The LPTN and Hammerstein model performance is shown in Table 5 for different sampling periods, evaluated for the temperatures in the winding and PM. For each model and temperature domain, the error metrics change only slightly across the evaluated sampling periods.

It is worth noting that no universal condition number threshold guarantees a single-precision implementation. Therefore, the condition number must be evaluated together with eigenvalue sensitivity. In this work, a state-space realization is sufficiently robust only if, after 32-bit truncation, its eigenvalues remain within the stable region. The proposed balanced realization achieves this objective, ensuring that the system maintains dynamic stability and preserves asymptotic stability after single-precision coefficient conversion under the evaluated conditions.

For all the evaluated sampling periods, the Hammerstein model reduced the winding temperature RMSE by approximately 62-64 %, whereas the PM temperature RMSE decreased by only 0.07-0.22 °C. The modest PM improvement is aligned with the large thermal capacitance of the lumped rotor/PM node, reported in Table 1, and the large thermal resistance of the air gap, which produces a smooth PM-temperature response that the baseline LPTN tracks closely. Therefore, the main contribution is the evaluation of the Hammerstein model finite-precision implementation, not the formulation of a novel or significantly more precise Hammerstein thermal model. For the evaluated realizations, balancing reduced the state-matrix condition number and retained the discrete poles inside the unit circle after single-precision coefficient conversion.

TABLE 6. Processor-in-the-loop timing performance and memory usage of LTI LPTN, PV LPTN, and Hammerstein temperature estimation methods.

Method	LTI LPTN	PV LPTN	Hammerstein
Initialization [μs]	4.27	2.47	94.52
Execution [μs]	60.75	74.43	161.42
Termination [μs]	1.06	1.18	1.06
Flash memory [kB]	9.07	9.74	10.13
RAM [kB]	1.9	1.72	2.29

C. HARDWARE IMPLEMENTATION

Timing results of the PIL simulation of the tested models are presented in Table 6. The Hammerstein model has higher initialization, execution, and termination times than the LTI LPTN and the PV LPTN. The above can be related to the complexity that the Hammerstein model introduces, having more parameters than the LPTN models (i.e., 136 parameters for the Hammerstein model compared to the LPTN model with only 22 parameters). Besides, the Hammerstein model requires pre-processing the inputs associated with the power losses through the use of the sigmoid network, which also influences the computational time of the model. Nevertheless, all three models fulfill an execution time $t_{\text{exe}} < 200\,\mu$s. These timings yield a CPU utilization $t_{\text{exe}}/\Delta t < 0.04\%$ even for models executed at $\Delta t = 0.5$ s. In terms of memory usage, all three methods demand similar Flash and random-access memory (RAM) space, with the Hammerstein model requiring slightly larger allocation than the other methods (10.13 kB of flash memory and 2.29 kB of RAM). In this latter case, the memory utilization is 2.29% for RAM, while flash memory utilization is 3.96%. These results demonstrate that the Hammerstein model can be implemented within the memory and computational capabilities of microcontrollers used in industrial and automotive applications. When comparing both LPTN-based methods, the LTI variant executes slightly faster than its PV counterpart.

Compared to other data-driven methods, the proposed Hammerstein model achieves a lower execution time on a constrained hardware platform. For instance, in [23], recurrent neural networks (RNN) and temporal convolutional networks (TCN) were evaluated on a Raspberry Pi 3 B+, reporting median inference times of 14.4 ms and 8.5 ms, respectively. Additionally, in [28], a hybrid model combining a multilayer neural network with a two-node LPTN achieved an execution time of 0.4 ms on a digital signal processor. Similarly, Cao et al. [15] tested a fifth-order LPTN, reporting an execution time of 86 μ s. Thus, the Hammerstein model in this implementation demonstrates a competitive execution time compared to other data-driven methods and LPTN models reported in the literature. Note that while these execution times and utilization ratios might seem very contained, these models must run sequentially together with the motor control task. A reduction in CPU usage is always beneficial in this context.

TABLE 7. Contextual comparison of representative LPTN, neural-network, and hybrid PMSM temperature-estimation methods.

Study and class	Reported accuracy	Model complexity	Execution time and platform	Memory	Embedded evidence
Cao et al. [15] LPTN-based magneto-thermal model	Validation maximum temperature error below $5\,^\circ$C; MSE and RMSE N/R	Fifth-order LPTN; 19 LPTN parameters and 43 total magneto-thermal parameters	$86.2\,\mu$s at 1 Hz; TMS320F28377D, 200 MHz	N/R	Online execution on the controller board was demonstrated.
Kirchgässner et al. [23] Deep residual NN	Mean MSE over PM, yoke, tooth, and winding: TCN $3.42\,\mathrm{K}^2$; RNN $8.70\,\mathrm{K}^2$	Unified TCN: approximately 7,000 parameters; unified RNN: approximately 14,000 parameters; 108 engineered input features	TCN: 8.5 ms; RNN: 14.4 ms; Raspberry Pi 3 B+	N/R	Execution was demonstrated on embedded-like Linux hardware; MCU deployment and deterministic real-time execution were not demonstrated.
Kirchgässner et al. [26] Thermal neural network	Small TNN: MSE $3.18\,\mathrm{K}^2$; maximum absolute error $5.84\,\mathrm{K}$	LPTN-structured TNN with 64 trainable parameters and five input features	N/R	N/R	The architecture is compact and physically interpretable, but target-processor timing was not reported.
Jin et al. [28] LPTN–ANN compensator	Stator/rotor MSE: $0.0708/0.2238\,^\circ\mathrm{C}^2$; maximum absolute error: $2.83/5.12\,^\circ\mathrm{C}$	Two-node LPTN with 11 parameters; 10–50–2 ANN, corresponding to 652 ANN coefficients; gradient smoothing	Approximately 0.4 ms at 2 Hz; TMS320F28335, 150 MHz	N/R	DSP execution was demonstrated using replayed experimental data.
Liu et al. [1] LPTN-informed LSTM	Overall MSE $2.04\,^\circ\mathrm{C}^2$; maximum absolute error $4.70\,^\circ$C for 10 temperatures	Third-order LPTN plus a one-layer LSTM with 20 hidden units; 2,390 total parameters	Inference time N/R; training required 4 min 32 s on a PC	N/R	Implementation on a low-cost target controller was identified as future work.
Martínez-Ríos et al. [29] Previous LPTN-initialized Hammerstein model	Three-fold average MSE (RMSE): winding $1.994\,^\circ\mathrm{C}^2$ ($1.322\,^\circ$C); PM $0.696\,^\circ\mathrm{C}^2$ ($0.799\,^\circ$C)	Fourth-order LPTN plus one-neuron sigmoid input nonlinearities; 136 parameters	$161\,\mu$s; TMS320F28069M, 90 MHz; PIL at $\Delta t = 2$ s	N/R; reported to fit the available flash.	PIL timing was demonstrated, but full concurrent integration with the motor-control tasks was not evaluated.
Present LTI LPTN	At $\Delta t = 0.5$ s, MSE (RMSE): winding $10.0813\,^\circ\mathrm{C}^2$ ($3.1751\,^\circ$C); PM $0.6269\,^\circ\mathrm{C}^2$ ($0.7918\,^\circ$C)	Fourth-order LPTN; 22 parameters	$60.75\,\mu$s; TMS320F28069M, 90 MHz; PIL	Flash: 9.07 kB; RAM: 1.90 kB	PIL implementation with measured execution time and memory allocation.
Present Hammerstein model	At $\Delta t = 0.5$ s, MSE (RMSE): winding **1.3078°C²** **(1.1436°C)**; PM **0.5162°C²** **(0.7185°C)**	Fourth-order LPTN plus sigmoid input nonlinearities; **136 parameters**	**161.42 µs**; TMS320F28069M, 90 MHz; PIL; CPU utilization below 0.04%	Flash: **10.13 kB**; RAM: **2.29 kB**	PIL implementation with measured resource allocation; single-precision dynamic stability was verified after state-space balancing.

N/R: not reported. TNN: thermal neural network. TCN: temporal convolutional network. RNN: recurrent neural network. ANN: artificial neural network. LSTM: long short-term memory.

Table 7 presents a comparison between conventional LPTN, neural-network, and hybrid temperature estimators. Since the studies use different machines, temperature targets, datasets, validation methods, and processors, the reported accuracy and execution times do not represent a controlled benchmark. However, the comparison indicates that low-order LPTNs require fewer parameters and shorter execution times, whereas neural and hybrid methods introduce additional complexity to describe nonlinear or unmodeled thermal behavior. Martinez-Ríos et al. [29] previously demonstrated PIL execution of the LPTN-initialized Hammerstein estimator. Thus, the contribution of this work is not the first processor execution of the Hammerstein structure. Instead, the present study extends that work through state-space conditioning and balancing, analysis of coefficient truncation and single-precision arithmetic, verification of discrete-time stability over sampling periods from 0.5 to 4 s, and quantitative characterization of execution time, CPU utilization, flash memory allocation, and RAM allocation. These results establish the numerical and computational implications of deploying an ill-conditioned estimator on an inverter-grade 32-bit microcontroller.

Although the PIL implementation demonstrates computational feasibility, it requires further analysis for product qualification. These include, but are not limited to, verification of flash, RAM, and stack usage, validation of sensor calibration, filtering delays, diagnostics, fallback behavior, and jitter analysis under concurrent motor-control tasks. Additional changes in the machine, cooling system, aging state or operating envelope also would require revalidation [3]. For automotive deployment, the validation must be guided by functional-safety life cycles, such as the ISO 26262 [56]. Consequently, the results must be considered as evidence of embedded feasibility rather than automotive or industrial qualifications.

Despite the results presented above, certain drawbacks are essential to consider. For instance, the ill-conditioned state-transition matrix of the linear block of the Hammerstein model has been generated principally due to the linear model manipulation and conversion between model types (i.e., conversion between state-space form to polynomial form and back to state-space form), which is unsuitable for fitting and implementing the model in a 32-bit processor [44]. Hence, generating a Hammerstein model implementation that fits the model without converting the linear block from the state-space form to the transfer function form could avoid introducing numerical stability issues. Despite the above, the balancing transformation provides an effective method to improve the numerical stability of the model, disregarding the numerical manipulations that it experiences. This is demonstrated in the results shown in Fig. 7 and Tables 3 and 4. Moreover, the Hammerstein model, as shown in Fig. 5, is more computationally demanding than the LPTN model, so the introduction of higher complexity of the models should be justified by evaluating how the improvement presented in Table 5 holds for the different working conditions of a specific PMSM. Additionally, while the balancing transformation improved the condition number of the state-transition matrix in the linear block of the Hammerstein model, the effectiveness of this method for other types of electric machines remains to be quantified.

The present work does not include a perturbation analysis covering sensor noise, parameter uncertainty, and thermal drift. The estimator directly used the signals recorded at 5 Hz without implementing an additional signal filtering step. Additionally, no dedicated noise injection was performed. Consequently, robustness to measurement noise, sensor bias, and drift, as well as the effects of potential filtering and its associated delay, were not quantified. The balanced realization enhances the numerical conditioning of the state-space representation, thus leading to numerical stability in the single-precision embedded implementation under the evaluated operating profiles. However, it does not provide robustness against noise or variations in temperature-dependent parameters. The validation dataset at 40 °C provides evidence of performance under different conditions but does not replace a dedicated robustness analysis. Therefore, future work will quantify sensitivity to signal perturbations and parameter variations and evaluate suitable filtering strategies and their associated delays.

IV. CONCLUSION AND FUTURE WORK

This paper analyzed the condition number of the state-transition matrix of a Hammerstein model linear block initialized with a fourth-order LPTN of an out-runner PMSM used to estimate the temperature in the winding and PM. The above was carried out to analyze whether the linear block of the Hammerstein model is well-conditioned and whether its dynamics remain invariant when truncating its parameters to single precision for microcontroller embedding purposes. The original state-transition matrix of the linear block of the Hammerstein model had a condition number greater than 32. The condition number of the state-transition matrix was improved by performing a balancing transformation, obtaining a value lower than 1.75 in the worst case. This operation provided unaltered dynamics when truncating its parameters to a single-precision representation. Furthermore, the Hammerstein model was tested, and the average execution time of the model was lower than 0.2 ms, leading to a CPU utilization ratio below 0.04%. Future work can explore using Hammerstein models to estimate the temperature of other PMSM topologies, such as interior permanent-magnet synchronous motors, or to initialize the Hammerstein model in higher-order LPTNs.

REFERENCES

[1] Z. Liu, W. Kong, X. Fan, Z. Li, K. Peng, and R. Qu, "Hybrid thermal modeling with LPTN-informed neural network for multinode temperature estimation in PMSM," *IEEE Trans. Power Electron.*, vol. 39, no. 9, pp. 10897–10909, Sep. 2024.

[2] Y. Park, D. Fernandez, S. B. Lee, D. Hyun, M. Jeong, S. K. Kommuri, C. Cho, D. D. Reigosa, and F. Briz, "Online detection of rotor eccentricity and demagnetization faults in PMSMs based on Hall-effect field sensor measurements," *IEEE Trans. Ind. Appl.*, vol. 55, no. 3, pp. 2499–2509, May 2019.

[3] O. Wallscheid, T. Huber, W. Peters, and J. Böcker, "A critical review of techniques to determine the magnet temperature of permanent magnet synchronous motors under real-time conditions," *EPE J.*, vol. 26, no. 1, pp. 11–20, Jan. 2016.

[4] D. Liang, Z. Q. Zhu, J. Feng, Y. Li, and S. Guo, "Initial temperature determination for real-time thermal models in permanent magnet synchronous machines," *IEEE Trans. Transport. Electrific.*, vol. 11, no. 1, pp. 2204–2218, Feb. 2025.

[5] D. D. Reigosa, F. Briz, M. W. Degner, P. Garcia, and J. M. Guerrero, "Magnet temperature estimation in surface PM machines during six-step operation," *IEEE Trans. Ind. Appl.*, vol. 48, no. 6, pp. 2353–2361, Nov. 2012.

[6] D. Reigosa, D. Fernandez, M. Martinez, J. M. Guerrero, A. B. Diez, and F. Briz, "Magnet temperature estimation in permanent magnet synchronous machines using the high frequency inductance," *IEEE Trans. Ind. Appl.*, vol. 55, no. 3, pp. 2750–2757, May 2019.

[7] Z. Song, R. Huang, W. Wang, S. Liu, and C. Liu, "An improved dual iterative transient thermal network model for PMSM with natural air cooling," *IEEE Trans. Energy Convers.*, vol. 37, no. 4, pp. 2588–2600, Dec. 2022.

[8] A. Boglietti, A. Cavagnino, D. Staton, M. Shanel, M. Mueller, and C. Mejuto, "Evolution and modern approaches for thermal analysis of electrical machines," *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 871–882, Mar. 2009.

[9] O. Wallscheid, T. Huber, W. Peters, and J. Böcker, "Real-time capable methods to determine the magnet temperature of permanent magnet synchronous motors—A review," in *Proc. IECON 40th Annu. Conf. IEEE Ind. Electron. Soc.*, Oct. 2014, pp. 811–818.

[10] L. He, Y. Feng, Y. Zhang, and B. Tong, "Methods for temperature estimation and monitoring of permanent magnet: A technology review and future trends," *J. Brazilian Soc. Mech. Sci. Eng.*, vol. 46, no. 4, p. 174, Apr. 2024.

[11] O. Wallscheid, "Thermal monitoring of electric motors: State-of-the-art review and future challenges," *IEEE Open J. Ind. Appl.*, vol. 2, pp. 204–223, 2021.

[12] Z. Q. Zhu, D. Liang, and K. Liu, "Online parameter estimation for permanent magnet synchronous machines: An overview," *IEEE Access*, vol. 9, pp. 59059–59084, 2021.

[13] Y. Guo, R. Xu, and P. Jin, "A real-time temperature rise prediction method for PM motor varying working conditions based on the reduced thermal model," *Case Stud. Thermal Eng.*, vol. 47, Jul. 2023, Art. no. 103098.

[14] Z. Sheng, D. Wang, J. Fu, and J. Hu, "A computationally efficient spatial online temperature prediction method for PM machines," *IEEE Trans. Ind. Electron.*, vol. 69, no. 11, pp. 10904–10914, Nov. 2022.

[15] L. Cao, X. Fan, D. Li, W. Kong, R. Qu, and Z. Liu, "Improved LPTN-based online temperature prediction of permanent magnet machines by global parameter identification," *IEEE Trans. Ind. Electron.*, vol. 70, no. 9, pp. 8830–8841, Sep. 2023.

[16] J. Feng, D. Liang, Z. Q. Zhu, S. Guo, Y. Li, and A. Zhao, "Improved low-order thermal model for critical temperature estimation of PMSM," *IEEE Trans. Energy Convers.*, vol. 37, no. 1, pp. 413–423, Mar. 2022.

[17] A. Boglietti, M. Cossale, M. Popescu, and D. A. Staton, "Electrical machines thermal model: Advanced calibration techniques," *IEEE Trans. Ind. Appl.*, vol. 55, no. 3, pp. 2620–2628, May 2019.

[18] M. S. Baumann, A. Steinboeck, W. Kemmetmüller, and A. Kugi, "Real-time capable thermal model of an automotive permanent magnet synchronous machine," *IEEE Open J. Ind. Electron. Soc.*, vol. 5, pp. 501–516, 2024.

[19] J. G. Urbieta, B. Rodríguez, A. J. Rodríguez, P. Díaz, S. Armentia, and F. González, "Sensitivity analysis of lumped-parameter thermal networks for the experimental calibration of eMotor models," *IEEE Trans. Transport. Electrific.*, vol. 10, no. 3, pp. 6210–6220, Sep. 2024.

[20] G. Feng, C. Lai, and N. C. Kar, "Expectation-maximization particle-filter- and Kalman-filter-based permanent magnet temperature estimation for PMSM condition monitoring using high-frequency signal injection," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 1261–1270, Jun. 2017.

[21] G. Feng, C. Lai, and N. C. Kar, "Speed harmonic based modeling and estimation of permanent magnet temperature for PMSM drive using Kalman filter," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1372–1382, Mar. 2019.

[22] D. E. G. Erazo, O. Wallscheid, and J. Böcker, "Improved fusion of permanent magnet temperature estimation techniques for synchronous motors using a Kalman filter," *IEEE Trans. Ind. Electron.*, vol. 67, no. 3, pp. 1708–1717, Mar. 2020.

[23] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Estimating electric motor temperatures with deep residual machine learning," *IEEE Trans. Power Electron.*, vol. 36, no. 7, pp. 7480–7488, Jul. 2021.

[24] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Data-driven permanent magnet temperature estimation in synchronous motors with supervised machine learning: A benchmark," *IEEE Trans. Energy Convers.*, vol. 36, no. 3, pp. 2059–2067, Sep. 2021.

[25] H. Jing, Z. Chen, X. Wang, X. Wang, L. Ge, G. Fang, and D. Xiao, "Gradient boosting decision tree for rotor temperature estimation in permanent magnet synchronous motors," *IEEE Trans. Power Electron.*, vol. 38, no. 9, pp. 10617–10622, Sep. 2023.

[26] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Thermal neural networks: Lumped-parameter thermal modeling with state-space machine learning," *Eng. Appl. Artif. Intell.*, vol. 117, Jan. 2023, Art. no. 105537.

[27] S. Zhang, O. Wallscheid, and M. Porrmann, "Machine learning for the control and monitoring of electric machine drives: Advances and trends," *IEEE Open J. Ind. Appl.*, vol. 4, pp. 188–214, 2023.

[28] L. Jin, Y. Mao, X. Wang, L. Lu, and Z. Wang, "A model-based and data-driven integrated temperature estimation method for PMSM," *IEEE Trans. Power Electron.*, vol. 39, no. 7, pp. 8553–8561, Jul. 2024.

[29] E. A. Martinez-Ríos, I. S. Aguilar-Zamorate, S. Pakštys, R. Galluzzi, and N. Amati, "Estimating temperature in a permanent-magnet synchronous motor using Hammerstein and nonlinear autoregressive models initialized via thermal networks," *IEEE Trans. Ind. Appl.*, vol. 62, no. 1, pp. 985–997, Jan. 2026.

[30] J. F. Whidborne, R. S. H. Istepanian, and J. Wu, "Reduction of controller fragility by pole sensitivity minimization," *IEEE Trans. Autom. Control*, vol. 46, no. 2, pp. 320–325, Feb. 2001.

[31] S. Keyumarsi, A. Nobakhti, and M. S. Tavazoei, "Non-fragile H_∞ order reduction of LTI controllers," *IEEE Control Syst. Lett.*, vol. 5, no. 1, pp. 163–168, Jan. 2021.

[32] T. Hinamoto, A. Doi, and W.-S. Lu, "Minimization of weighted pole and zero sensitivity for state-space digital filters," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 1, pp. 103–113, Jan. 2016.

[33] F. S. V. Bazán, "Eigensystem realization algorithm (ERA): Reformulation and system pole perturbation analysis," *J. Sound Vibrat.*, vol. 274, nos. 1–2, pp. 433–444, Jul. 2004.

[34] MathWorks. (2024). *Conversion Between Model Types*. MathWorks. Accessed: Jul. 17, 2026. [Online]. Available: https://www.mathworks.com/help/control/ug/conversion-between-model-types.html

[35] L. H. Keel and S. P. Bhattacharyya, "Robust, fragile, or optimal?" *IEEE Trans. Autom. Control*, vol. 42, no. 8, pp. 1098–1105, Aug. 1997.

[36] E. A. Martinez-Ríos, I. S. Aguilar-Zamorate, S. Pakštys, R. Galluzzi, and N. Amati, "Hammerstein models for rotor and winding temperature estimation of a permanent magnet synchronous motor," in *Proc. Int. Symp. Electromobility (ISEM)*, Sep. 2024, pp. 1–8.

[37] M. Schoukens and K. Tiels, "Identification of block-oriented nonlinear systems starting from linear approximations: A survey," *Automatica*, vol. 85, pp. 272–292, Nov. 2017.

[38] F. Li and L. Jia, "Parameter estimation of Hammerstein–Wiener nonlinear system with noise using special test signals," *Neurocomputing*, vol. 344, pp. 37–48, Jun. 2019.

[39] X.-M. Chen and H.-F. Chen, "Recursive identification for MIMO Hammerstein systems," *IEEE Trans. Autom. Control*, vol. 56, no. 4, pp. 895–902, Apr. 2011.

[40] O. Wallscheid and J. Böcker, "Global identification of a low-order lumped-parameter thermal network for permanent magnet synchronous motors," *IEEE Trans. Energy Convers.*, vol. 31, no. 1, pp. 354–365, Mar. 2016.

[41] S. Xue, J. Feng, S. Guo, J. Peng, W. Q. Chu, and Z. Q. Zhu, "A new iron loss model for temperature dependencies of hysteresis and eddy current losses in electrical machines," *IEEE Trans. Magn.*, vol. 54, no. 1, pp. 1–10, Jan. 2018.

[42] K. J. Åström and B. Wittenmark, "Computer-controlled systems: Theory and design," *Computer-Controlled Systems: Theory Design*, 3rd ed. Garden City, NY, USA: Dover, Courier Corporation, 2011.

[43] F. N. Deniz, B. B. Alagoz, N. Tan, and M. Koseoglu, "Revisiting four approximation methods for fractional order transfer function implementations: Stability preservation, time and frequency response matching analyses," *Annu. Rev. Control*, vol. 49, pp. 239–257, Jun. 2020.

[44] A. Laub and J. Little, *MATLAB Control System Toolbox User's Guide*. Natick, MA, USA: MathWorks, Inc., 1986.

[45] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*. Oxford, U.K.: Oxford Univ. Press, 1988.

[46] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed., Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2002.

[47] D. J. Higham, "Condition numbers and their condition numbers," *Linear Algebra Appl.*, vol. 214, pp. 193–213, Jan. 1995.

[48] W. S. Levine, Ed., *Control System Fundamentals*, 1st ed., Boca Raton, FL, USA: CRC Press, 2019.

[49] A. Laub, M. Heath, C. Paige, and R. Ward, "Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms," *IEEE Trans. Autom. Control*, vol. AC-32, no. 2, pp. 115–122, Feb. 1987.

[50] W. Gawronski and J.-N. Juang, "Model reduction in limited time and frequency intervals," *Int. J. Syst. Sci.*, vol. 21, no. 2, pp. 349–376, Feb. 1990.

[51] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge, U.K.: Cambridge Univ. Press, 2022.

[52] R. Jirasek, T. Schauer, D. Su, T. Nagayama, and A. Bleicher, "Experimental linear parameter-varying model identification of an elastic kinetic roof structure," *Eng. Struct.*, vol. 297, Dec. 2023, Art. no. 116986.

[53] K. Glover and X. Yuan, "Balanced realisations of all-pass systems using exact arithmetic with application to the approximation of delay systems," in *Proc. IEEE 51st IEEE Conf. Decis. Control (CDC)*, Dec. 2012, pp. 4942–4946.

[54] H. Vardhan, B. Akin, and H. Jin, "A low-cost, high-fidelity processor-in–the loop platform: For rapid prototyping of power electronics circuits and motor drives," *IEEE Power Electron. Mag.*, vol. 3, no. 2, pp. 18–28, Jun. 2016.

[55] J. Böcker, "Discrete-time model of an induction motor," *Eur. Trans. Electr. Power*, vol. 1, no. 2, pp. 65–71, 1991.

[56] *Road Vehicles—Functional Safety*, document ISO 26262:2018, International Organization for Standardization, Geneva, Switzerland, 2018. [Online]. Available: https://www.iso.org/standard/68383.html

ERICK AXEL MARTINEZ-RÍOS was born in Mexico City, Mexico, in May 1995. He received the B.S. degree (Hons.) in mechatronics engineering, and the M.S. (Hons.) and Ph.D. degrees in engineering science from Instituto Tecnológico y de Estudios Superiores de Monterrey, Mexico City, in 2017, 2022, and 2026, respectively. From 2018 to 2020, he was a Project Specialist with Instituto Tecnológico y de Estudios Superiores de Monterrey. During this stay, he worked on research and development projects in the field of robotics. He is currently a Senior Scientist with PepsiCo Inc., providing cross-functional statistical analysis and data science support across research and development initiatives. He is the author of 16 journal articles, five conference papers, and one book chapter. His research interests include statistics, signal processing, machine learning, and control theory.

IRVING S. AGUILAR-ZAMORATE received the B.Sc. degree in mechanical engineering and the M.S. degree in engineering science from Tecnológico de Monterrey, Mexico City, in 2020 and 2022, respectively, where he is currently pursuing the Ph.D. degree in engineering science, with a research stay at the Mechatronics Laboratory, Politecnico di Torino. His research interests include thermal management, temperature estimation, design, and control of electric machines for automotive and micro-mobility applications.

SAULIUS PAKŠTYS received the B.Sc. and M.Sc. degrees in mechanical engineering from Politecnico di Torino, Turin, Italy, in 2021 and 2022, respectively, where he is currently pursuing the Ph.D. degree in mechanical engineering with the Mechatronics Laboratory. His research interests include energy management strategies for hybrid energy storage systems, alternative automotive powertrains, highly integrated battery systems design and their thermal management, and electric machine thermal analysis.

RENATO GALLUZZI (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in mechatronics from Politecnico di Torino, Italy, in 2010 and 2014, respectively. Since 2011, he has been the Active Collaborator of the Mechatronics Laboratory, Politecnico di Torino. He is currently a Research Professor with the School of Engineering and Sciences, Tecnológico de Monterrey, Mexico. He is the author of numerous journal publications and conference papers. He also holds inventorship in multiple international patents. His research interests include vibration control and damping systems, power actuators, electric machinery, and energy harvesting. He is a member of the National Research Fellows System Level 1 (SNII-1), SECIHTI, Mexico.

LUIS M. CASTELLANOS MOLINA received the Engineering degree in automation from Las Villas Central University, Villa Clara, Cuba, in 2011, the M.Sc. degree in energy efficiency from Cienfuegos University, Cuba, and the Ph.D. degree in mechanical engineering from Politecnico di Torino, Italy, in 2020. He was a Technician in automation with the Oil Refinery of Cienfuegos, Cuba, from 2011 to 2013. He was a Research Assistant and a Junior Lecturer with Cienfuegos University, from 2013 to 2015. He is currently a Postdoctoral Researcher with Politecnico di Torino. His research interests include model predictive control and its applications to magnetic suspension systems.

NICOLA AMATI received the Ph.D. degree in machine design from the Department of Mechanical Engineering, Politecnico di Torino, Turin, Italy, in 2001. In the same year, he joined the Faculty of Politecnico di Torino, where he is currently a Full Professor with the Mechanical and Aerospace Engineering Department. He is also a Principal Investigator with the Center of Automotive Research and Sustainable Mobility, Politecnico di Torino. His research interests include the analysis, design, and control of electromechanical systems, with an emphasis on rotating machinery, active and passive magnetic bearings and dampers, and more-electric systems for automotive applications.

ROGELIO BUSTAMANTE-BELLO was born in Orizaba, Veracruz, in 1965. He received the B.S. degree in electronic instrumentation from the Faculty of Physics and Mathematics, Universidad Veracruzana, Xalapa, Veracruz, the master's degree in electronics and telecommunications from CICESE, Ensenada, Baja California, Mexico, and the Ph.D. degree in telecommunications and electronics from SEPI, Instituto Politécnico Nacional. He has worked for more than 12 years in advanced bioinstrumentation systems, applying them to driving assistance systems (ADAS) and the development of healthy active spaces (emotional domotics) and exoskeletons regarding the movement of lower limbs and bioinstrumentation systems. He has filed multiple patents with IMPI and WIPO.