Mechanical Systems and Signal Processing 229 (2025) 112469

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

Physics-Informed ensembles of dual extended kalman filters for online state and parameter estimation of mechatronic systems

Van Heck Cedric *, Vandeputte Matthias, Coene Annelies, Crevecoeur Guillaume

Dynamic Design Lab (D2LAB), Department of Electromechanical, Systems and Metal Engineering, Faculty of Engineering and Architecture, Ghent University, 9000 Ghent, Belgium

Associated with MIRO, Ghent University corelab of Flanders Make, the strategic research center for the manufacturing industry, Belgium

ARTICLE INFO

Communicated by S. Laflamme

Keywords:
Dual Kalman filter
Extended Kalman filter
Neural networks
Hybrid modeling
Physics inspired
Ensemble methods
Mechatronic systems

ABSTRACT

Kalman filters are widely used for real-time sensor data processing in mechatronic and production systems, but their effectiveness depends on accurate system models. These models also need updating in order to handle changing or unknown operating conditions. Data-driven methods excel in adapting models to match measurements without heavy reliance on expert knowledge, but they suffer from drawbacks such as reduced interpretability, limited extrapolatability, and poor reliability for long-term predictions. Hybrid methods that integrate traditional and data-driven modeling aim to combine the strengths of both approaches, yet achieving accurate convergence between the distinct physical and data-driven components remains a significant challenge during joint optimization. This paper introduces a novel methodology based on three key ideas. First, a dual extended Kalman filter is used to jointly estimate and adapt model parameters in real-time. Second, a hybrid model combining physics-based and data-driven parameters is embedded in the filter, resulting in a more generalizable and accurate model. Finally, an extension to ensembling techniques is proposed to improve convergence toward correct physical parameters. The proposed ensemble of hybrid dual extended Kalman filters is evaluated on a cam follower system working under a wide range of operating conditions. Results demonstrate its ability to learn both the cam shape and operating voltage during operation and converged models show a reduced error in physical parameter estimates of more than 60% compared to alternative filtering techniques. Moreover, these models outperform other methods in multi-step prediction, showcasing an improvement of more than 48% in prediction accuracy. With this work, we aim to demonstrate the efficacy of physics-informed models in a filtering context and address the challenge of correct parameter learning despite the overparameterization inherent in these models.

1. Introduction

Mechatronic systems are known to be highly dynamical systems whose behavior is dependent on the operating and environmental conditions they are placed in. These conditions are not always known in advance and might even be varying over time. By predicting the behavior of these systems, operation settings and control parameters can be altered in favor of quality, performance and

* Corresponding author at: Dynamic Design Lab (D2LAB), Department of Electromechanical, Systems and Metal Engineering, Faculty of Engineering and Architecture, Ghent University, 9000 Ghent, Belgium.
E-mail address: cevheck.vanheck@ugent.be (V.H. Cedric).

https://doi.org/10.1016/j.ymssp.2025.112469
Received 4 September 2024; Received in revised form 6 January 2025; Accepted 11 February 2025
Available online 8 March 2025
0888-3270/© 2025 Published by Elsevier Ltd.

efficiency [1,2]. For this purpose, a good model of the system is required. These models are generally derived from classical physics-based modeling strategies [3]. An alternative approach is to leverage the amount of sensor data in industry by building data-driven models. Although the latter has proven its applicability in mechatronics [4–6], physics-based modeling remains most prominent for modeling dynamic behavior. Such models tend to provide better extrapolation capabilities than their data-driven counterpart. This is attributed to their physical underpinnings and allows interpretation [7,8]. It comes as no surprise that many companies are reluctant to throw away this previously obtained expertise and want to re-use this a priori information to have a trustworthy base upon which models are used for design and operation. Therefore methods that reuse this a priori information lower the technological barrier. This leads us to the field of scientific machine learning [9], where data-driven methodologies are applied to work hand in hand with existing knowledge of traditional scientific mechanistic modeling [10–14]. By combining both methodologies, physical constraints and dynamics can be enforced to the model whilst the addition of data-driven methods enables the models to better align with the systems in reality.

Mechatronic systems are mostly well understood and captured in conventional modeling methodologies. Concepts such as force balances, momentum and heat exchange are well known and easily captured in mathematical equations. However, on a smaller scale there still exist many phenomena that are not completely understood or require a lot of modeling effort to capture. Some examples are fine modeling of turbulence [15], interaction forces [4,16] and friction [17,18]. A second type of uncertainty that arises in mechatronic systems is where the exact physics are known, however, some physical parameters are not known accurately. An example is the inertia of an object, where the mathematical formulations are known, however the exact mass or mass distribution of the object might be unknown or not accurately measured. The physics-informed or *hybrid* approach suggested in [7] attempts to separately capture both types of uncertainties by embedding the majority of dynamics in known equations whilst using neural networks to learn unmodeled phenomena from the data.

Given the joint optimization problem of physical and data-driven parameters, a prominent issue is the existence of local minima, potentially resulting in a potent total model, but with an incorrect underlying physical model. This issue becomes more pronounced with overparameterized data-driven methods like neural networks, where the data-driven component may overshadow the physical model, rendering it obsolete. Strategies to mitigate this problem typically involve minimizing the influence of the data-driven component. In some cases, the non-uniqueness of the parameter convergence is neglected, with sole focus on final performance [19]. In these situations, model performance can be increased by using techniques to avoid and escape local minima such as dedicated optimizers [20] and methods to explore multiple regions of the solution space (e.g. Genetic Algorithms [21]). [22] investigates both of these solutions applied to a physics informed neural network. However, when the accuracy of the retrieved physical parameters is of importance, this non-uniqueness can no longer be overlooked as addressed in [23]. A common attempt to enforce proper convergence, as previously noted, is to diminish the influence of the data-driven component, with ℓ_2-regularization being the foremost method [24]. APHYNITY [23] tackles the same issue by minimization of the action of the data-driven component. Both methods essentially revolve around augmenting the physical model *"when - and only when - it falls short [23]"*. However, these approaches rely on the assumption that a decent physical model can be learned with residual corrections, whereas most hybrid modeling techniques are employed in the presence of an unknown model mismatch. Within the framework of this paper, it is assumed that a properly converged model is defined by its ability to deliver high performance over a broad range of operating conditions. This characteristic aligns with the good extrapolation capabilities of physical models.

The aforementioned models can be used in online state estimation techniques to monitor the state and health of mechatronic applications. Bayesian filtering methods (e.g. Kalman filter [25]) are used to combine model predictions with noisy measurements to provide a real-time estimate of the state. Models used within this context of filtering are generally obtained offline based on extensive engineering effort and available historic datasets. On the contrary, a mechatronic system typically has varying operating conditions, suffers from quality variation due to the alternation of wear and maintenance, shows piece-to-piece variation and is influenced by unmodeled dependencies (e.g. temperature). Using a single model on this whole spectrum of system states appears inaccurate, whilst building a distinct model for each state is cumbersome. Therefore the need for an adaptive model arises.

Continual learning algorithms offer a first solution by dynamically adjusting models to address degradation or other influencing factors over time [26]. For scenarios like piece-to-piece variation or unknown operating conditions, a common approach involves starting with a pre-trained model and fine-tuning its parameters to optimize performance for the specific circumstances, typically refining its behavior to match that of the particular instance entering production or undergoing estimation. This approach, known mostly from machine learning, is called transfer learning [27]. However due to the batchwise training of most data-driven methodologies, updates still occur at certain discrete time instants to be determined by the user, rather than in a truly online manner. This introduces a delay, rendering it suboptimal for real-time monitoring, especially in mechatronic applications where systems often operate on fast timescales. An alternative that addresses these limitations is the Extended Kalman Filter (EKF). When compared to prevalent gradient based learning methodologies (e.g. backpropagation [28]), the EKF based approach for parameter tracking and inverse modeling proves to be faster and more robust in the presence of noise [29–32]. Additionally, this filtering-based method seamlessly integrates with high-rate and sample-by-sample data commonly encountered with encoders in mechatronic environments. Furthermore, it easily combines with state estimation techniques, leading to simultaneous state- and parameter estimation. When integrated with the hybrid model framework, this enables the tracking of changes in both modeled, but uncertain, physical parameters and changes in unknown phenomena that are not captured in equations, all within a noisy state environment.

Various methods to tackle this joint state- parameter estimation exist. In [33], two primary approaches based on the EKF are discussed. The first and more straightforward approach augments the state vector with the unknown model parameters into a single combined vector to estimate, resulting in what is known as the Augmented or Joint Kalman Filter (AKF). Alternatively, the Dual Kalman Filter (DKF), employs two separate Kalman Filter; one for state estimation and the other for parameter estimation. In this

approach, the filtered state estimate is used in the parameter filter and vice versa. Beyond these two methods, numerous other filtering techniques address this joint optimization problem. For instance, one can exploit known relationships between the state and parameter vectors. An example of this is presented in [34], where an information vector is utilized to identify the parameter vector using extended stochastic gradients and combined with an adaptive EKF for state- and parameter estimation. Furthermore, other filtering methods that extend beyond the EKF are also applicable, such as the Unscented Kalman Filter [33,35], Ensemble Kalman Filter [36], and Particle Filter [37], all of which can be applied in either a joint or dual framework.

When comparing those methodologies, the Dual Kalman Filter (DKF) [33,38,39], demonstrates greater stability and robustness in the presence of model inaccuracies [33,40]. When using the joint Kalman Filter, observability of the augmented state vector is essential to the success of the algorithm, as highlighted in [35,41,42]. Techniques for verifying observability can be found in [43], and an example specific to the AKF is provided in [44], where the emperical observability Gramian is used to verify the observability. In [41] it is shown that the successive structure of the DKF does not trigger the same unobservability issues reported on the application of the AKF. However, a rigorous theoretical justification for this observation is missing. Furthermore, [45] argues that, for the dual observer, the same observability analysis as the joint approach is not applicable. Instead, an ad-hoc approach is used to identify non- or weak observability. To the best of our knowledge, a general framework for analyzing observability in the dual framework is still an open research topic.

While the (Dual) Extended Kalman Filter offers a robust solutions to the challenges of parameter tracking and state estimation, relying on a single hybrid model may still be susceptible to convergence issues between physical and data-driven parameters, as previously discussed. To further mitigate these challenges, ensemble methods have emerged as a powerful tool [46,47]. Ensemble techniques involve the use of multiple models to capture a broader range of potential system behaviors, thereby exploring multiple local minima. In filtering applications, this concept is embodied in Multiple Model Kalman Filters (MMKFs), which employ a set of Kalman filters, each tuned to different model hypotheses [48]. Although MMKFs are typically used to switch between static models, the same principles can be used to systematically investigate and leverage multiple potential solutions. By dynamically aggregating the outputs of these filters, ensemble methods enhance the overall reliability and accuracy of filtering and parameter estimation.

In this paper we address the challenge of achieving correct physical parameter convergence and accurate filtering, both within an online setting. Therefore, we develop a novel combined online state estimation and learning procedure of a dual Kalman filter with a hybrid model under the form of a Neural Network Augmented Physics (NNAP) model [7]. We refer to this combination as a hybrid model based dual extended Kalman filter (H-DEKF). The proposed framework is better suited to adapt an initial model during operation to varying environments and operating conditions that occur in industrial settings. In such manner we go beyond existing methodologies such as neural extended Kalman filters [49], residual methods [50] and other state of the art methods [51–54]. Looking at most recent review papers concerning Kalman filters in AI context [55], the algorithm is a novel error compensation and parameter tuning technique, with additional attention for interpretability and physical regularization. A central challenge within this research and these physics-inspired models is to address the problem of overparameterization and local minima in hybrid models. To mitigate this, we propose an extension based on ensembling methods and investigate various ensemble aggregation techniques. The effectiveness of our approach is experimentally validated through its application to a cam follower mechanism. This mechatronic system, shown in Fig. 2(a), is ubiquitous in the automotive sector, where varying operating conditions and deterioration are often encountered. Furthermore its ability to convert rotary motion into a linear translation proves useful in other industry as well such as textile, food, medical, and other fields [56]. The proposed method allows to give an online estimate of physical parameters such as the operating voltage or the used cam shape whilst detecting and compensating for unknown phenomena such as friction in the system or unknown interaction forces.

The manuscript is structured as follows. Section 2 provides a detailed introduction to the Hybrid Model-based Dual Extended Kalman Filter (H-DEKF), including its core framework and two key extensions incorporating ensemble-based techniques to enhance model performance and avoid local minima. Section 3 provides a comprehensive overview of the cam follower system, including its governing equations. This Section also describes the experimental setup and the data used for validation. It also introduces and elaborates various models applied to the cam follower system, encompassing physics-based, data-driven, and hybrid approaches. Section 4 outlines the evaluation methods employed to assess the H-DEKF framework, with a focus on filtering performance and parameter estimation. It includes a comparison of the hybrid model with other models and evaluates different ensemble aggregation methods. The Section concludes with a detailed discussion of the experimental results. Finally, Section 5 summarizes the key findings and contributions of the study, addresses the main limitations of the proposed method and offers suggestions for future research.

2. Methodology

This Section provides a theoretical overview of the proposed method. The Hybrid Dual Extended Kalman Filter (H-DEKF) is an assembly of multiple subsystems. Each of these subsystems will be briefly explained within this Section. The Section ends with two extensions to the H-DEKF. First, an extension to adaptive Kalman Filtering for automatic estimation of process- and measurement noise covariances is handled. Secondly an ensemble-based approach with dedicated selection criteria for ensemble subselection is presented.

2.1. Kalman filter

The Kalman filter (KF) is one of, if not the most used Bayesian filtering [57] technique in practice. It provides an online updating methodology to obtain a Gaussian estimate (with mean $\hat{\mathbf{x}}_k$ and covariance P_k) of the real state, $\mathbf{x}_k$. This estimation procedure

combines model predictions ($\hat{\mathbf{x}}_k^-$) with noisy measurements ($\mathbf{y}_k$) and merges both into the estimated state of the system. It relies on the Markov property [58] to enforce conditional independence between subsequent states, i.e. given your current state estimate (at time k) and applied control signal $\mathbf{u}_k$, the past measurements become redundant. This property makes the KF a memory- and computationally efficient algorithm for online estimation [33]. For linear systems it can be proven that the KF obtains an optimal estimate [59] using the following equations:

1. State-space model

$$\begin{aligned} \mathbf{x}_{k+1} &= A_{k+1}\mathbf{x}_k + B_{k+1}\mathbf{u}_k + \mathcal{N}(0, Q) \\ \mathbf{y}_k &= C_k\mathbf{x}_k + \mathcal{N}(0, R) \end{aligned} \tag{1}$$

2. State estimate propagation

$$\hat{\mathbf{x}}_k^- = A_k\hat{\mathbf{x}}_{k-1} \tag{2}$$

3. Error covariance propagation

$$P_k^- = A_k P_{k-1} A_k^T + Q \tag{3}$$

4. Kalman Gain calculation

$$K_k = P_k^- C_k^T [C_k P_k^- C_k^T + R]^{-1} \tag{4}$$

5. State estimate update

$$\begin{aligned} \boldsymbol{\epsilon}_k &= \mathbf{y}_k - C_k\hat{\mathbf{x}}_k^- \\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + K_k\boldsymbol{\epsilon}_k \end{aligned} \tag{5}$$

6. Error covariance update

$$P_k = (I - K_k C_k) P_k^- \tag{6}$$

where

- Q, R are assumed to be diagonal covariance matrices, expressing zero-mean Gaussian noise on the state propagation and measurement respectively.
- A_k, B_k, C_k are the linear state transition matrix, the control-input matrix and observation matrix [33]

The KF is used in an online manner by defining starting conditions $\mathbf{x}_0$ and P_0. Afterwards, (2)–(6) are repeated as measurements come in. The matrices A_k, C_k, Q and R are assumed to be known quantities.

2.2. Dual extended Kalman filter

For the problem setting within this research, the original KF implementation does not suffice. The reasons for this are twofold.

A first shortcoming of the original KF is the linear assumptions present in the update Eqs. (1)–(6). In order to tackle this issue, multiple approaches exist. The most straightforward approach leads to the Extended Kalman Filter (EKF) [59]. In this approach the state transition is derived by the obtained non-linear model $\mathcal{F}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$, however in order to allow the usage of (3)–(6), linear matrix formulations of the state transition function and measurement function $\mathcal{H}(\mathbf{x}_k, \mathbf{w}_k)$ are required. These can be obtained by linearizing around the current state estimate, hence leading to

$$A_{k+1} = \frac{\partial \mathcal{F}(\mathbf{x}, \mathbf{u}_k, \mathbf{w}_k)}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_k}, \qquad C_k = \frac{\partial \mathcal{H}(\mathbf{x}, \mathbf{w}_k)}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_k^-} \tag{7}$$

Secondly, it is assumed that the (complete) model of the system is not available and hence the state transition function, $\mathcal{F}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$, is not exactly known. In order to learn this model one could resort to offline data-driven methods. However, as models can be time-varying in nature or have certain parameters that are unknown a priori, a more adaptive approach is to learn and modify these system parameters $\mathbf{w}$ in an online fashion. This results in an adaptive model of the system, which is updated as operational data comes in. One way to do so, is by using a KF to track the variable parameters i.e. a KF as defined in 2.1. Contrary to a KF used for state estimation, a model of parameter evolution (parameter transition matrix $A_{w,k+1}$) is not known, hence the evolution of parameters is not captured in a model since we do not know in advance how they will evolve. The parameter transition matrix is defined to be the identity matrix, resulting in a variation driven by noise as seen in (8).

$$\begin{aligned} \mathbf{w}_{k+1} &= A_{w,k+1}\mathbf{w}_k + \mathcal{N}(0, Q_w) \\ \mathbf{w}_{k+1} &= \mathbf{w}_k + \mathcal{N}(0, Q_w) \end{aligned} \tag{8}$$

This noise adds uncertainty to the parameters and hence enforces that the parameters are matched to the measurements in the parameter estimate update. Therefore the noise can be interpreted as a measure of flexibility in the model parameters. The parameter estimate update is defined similarly to (5). In the general case where $\mathcal{F}$ depends nonlinearly on the parameters w, C_k is defined

analogously to Eq. (7), resulting in Eq. (9). We introduce subscripts x and w to refer to variables corresponding to the state- and parameter Kalman filter respectively.

$$C_{w,k} = \frac{\partial \mathcal{H}(\mathbf{x}_k, \mathbf{w})}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} = \frac{\partial \mathcal{H}(\mathbf{x}, \mathbf{w}_k)}{\partial \mathbf{x}}|_{\mathbf{x}_k} \frac{\partial \mathbf{x}_k}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} = C_{x,k} \frac{\partial \mathcal{F}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w})}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} \tag{9}$$

If the estimation of the parameters is combined with the tracking of the state, a joint optimization problem arises. This joint optimization can be solved by augmenting the state vector with the variable parameters, resulting in a joint KF [33]. Another approach suggests to run two Kalman filters concurrently. The latter method, the Dual Kalman Filter (DKF), proves to be more stable and more robust in case of inaccuracies in the model [33,40] and hence is opted for within this research.

In the dual framework, the state, $\mathbf{x}_k$, in the parameter Kalman filter becomes the estimated state $\mathbf{x}_k^-$ and therefore Eq. (9) is rewritten as:

$$C_{w,k} = C_{x,k} \frac{\partial \hat{\mathbf{x}}_k^-}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} \tag{10}$$

With

$$\begin{aligned} \frac{\partial \hat{\mathbf{x}}_k^-}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} &= \frac{\partial \mathcal{F}(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1})}{\partial \mathbf{x}}|_{\hat{\mathbf{x}}_{k-1}} \frac{\partial \hat{\mathbf{x}}_{k-1}}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} + \frac{\partial \mathcal{F}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w})}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_{k-1}} \\ &= A_{x,k} \frac{\partial \hat{\mathbf{x}}_{k-1}}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} + \frac{\partial \mathcal{F}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1})}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_{k-1}} \end{aligned} \tag{11}$$

$$\frac{\partial \hat{\mathbf{x}}_{k-1}}{\partial \mathbf{w}}|_{\hat{\mathbf{w}}_k^-} = (I - K_{x,k-1}) \frac{\partial \hat{\mathbf{x}}_{k-1}^-}{\partial \mathbf{w}} \tag{12}$$

Important to note is the recursive nature of Eqs. (11) and (12). This recursive derivative needs to be added to account for the dependency of the output function to the previous state. Given that both Eqs. (7) and (10) are using the formulation of an EKF, the resulting dual Kalman filter is referred to as Dual Extended Kalman Filter (DEKF). In order to limit repetition of similar formulas, the resulting equations of the DEKF are summarized in Section 2.4.

2.3. Hybrid modeling

In the KF, a linear dependency on the state is imposed on the state transition function. The EKF relaxed this assumption to any system that can be linearized (7). Very complex networks such as deep neural networks [60] can thus be used to approximate the dynamic behavior of the system. However those networks tend to have an extensive amount of tunable parameters $\boldsymbol{\alpha}$, leading to a difficult optimization problem. Furthermore, without proper regularization, those function approximators tend to find very complex function mappings $\boldsymbol{\eta}(\boldsymbol{\alpha})$ which in turn lead to poor linear approximations and hence are not well suited for the current framework [61]. On the other hand physics-based models tend to capture a great deal of dynamics in only a few parameters $\mathbf{p}$. Great examples are lumped parameter models [62]. However those models often lack accuracy and require significant amounts of expert knowledge, hence model updates are rather rare. Therefore, we introduce a hybrid modeling approach in the KF. In hybrid modeling, the goal is to achieve the best of both worlds by capturing known dynamics in physics-based models and find unknown dynamics through general function approximators. In this paper, an implementation as described in [7] is employed. Known dynamics of the system are captured in ordinary differential equations (ODE) wherein possibly some physical quantities $\mathbf{p}$ are not known accurately or are known to change over time. Additionally, we capture unknown interactions within the system by neural networks, which are inserted in a systematic manner in the ODE with the aim to reduce the tendency and ability of overfitting and to keep the advantageous properties of physical modeling such as extrapolatibility and interpretability. A significant challenge in these models is the simultaneous optimization of the physical parameters and the neural network parameters, as they are generally not decoupled in the ODE. This issue is central to this research, and various methods have been proposed to address it, as discussed in Sections 2.5.2, 4.3 and 4.4.

Fig. 1 shows a general implementation of a hybrid model $\mathcal{F}$ (shaded in gray), already embedded in the extended dual Kalman filter framework. As mentioned above, the dynamics are captured in an ODE $\mathbf{f}$, whose output depends on the current state $\mathbf{x}_k$, control $\mathbf{u}_k$, physical interpretable parameters $\mathbf{p}_k$ and the neural network output $\mathbf{z}_k$. The latter is in its turn dependent on the neural network inputs $\mathbf{q}_k$ and model parameters $\boldsymbol{\alpha}_k$. This leads to a total set of tunable parameters $\mathbf{w} = \{\mathbf{p}, \boldsymbol{\alpha}\}$. The ODE output can be combined with traditional solver schemes such as Forward Euler [63] to result in a discrete state transition function $\mathcal{F}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$.

$$\mathbf{x}_{k+1} = \mathcal{F}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) = \mathbf{x}_k + \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{p}_k, \mathbf{z}_k)\Delta t \tag{13}$$

2.4. H-DEKF: Equations

A summary of the above Sections is provided in the form of the final algorithmic equations of the H-DEKF. In comparison with Eqs. (1)–(6), subscripts x and w are introduced to denote variables associated with state and parameter Kalman Filters respectively. The equations are formulated assuming full state measurement in correspondence with the paper's use case. Therefore

$$\begin{aligned} \mathcal{H}(\mathbf{x}, \mathbf{w}_k) &= \mathbf{x}_k \\ C_k &= I \end{aligned} \tag{14}$$

This leads to the following Ensemble H-DEKF equations:

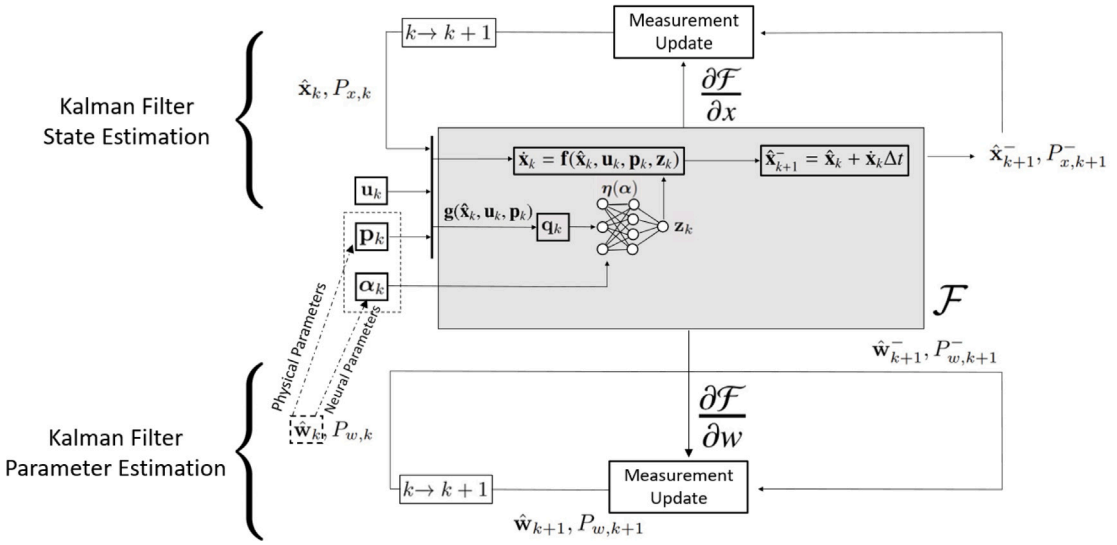


Fig. 1. Overview of the proposed Hybrid-Dual Extended Kalman filter. A Hybrid model $\mathcal{F}$ is both used for state estimation and parameter estimation. For state estimation, a best belief of the current state $\hat{\mathbf{x}}_k$ is obtained. During parameter estimation the physical parameters $\mathbf{p}$ and neural network parameters $\boldsymbol{\alpha}$ are jointly optimized in the parameter vector $\mathbf{w}$.

1. State-space model and parameter evolution

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathcal{F}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) + \mathcal{N}(0, Q_x) \\ \mathbf{y}_k &= \mathbf{y}_{x,k} = \mathbf{x}_k + \mathcal{N}(0, R) \end{aligned} \tag{15}$$

$$\begin{aligned} \mathbf{w}_{k+1} &= \mathbf{w}_k + \mathcal{N}(0, Q_w) \\ \mathbf{y}_{w,k} &= \mathbf{y}_{x,k} \end{aligned} \tag{16}$$

   where the measured signal $\mathbf{y}_k$ is a direct measurement of the states corrupted by white noise, representing errors on the measurements.
2. State estimate propagation

$$\begin{aligned} \hat{\mathbf{x}}_k^- &= \mathcal{F}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \\ \hat{\mathbf{w}}_k^- &= \hat{\mathbf{w}}_{k-1} \end{aligned} \tag{17}$$

3. Error covariance propagation

$$P_{x,k}^- = A_k P_{x,k-1} A_k^T + Q_x \tag{18}$$

$$P_{w,k}^- = P_{w,k-1} + Q_w \tag{19}$$

4. Kalman Gain calculation

$$K_{x,k} = P_{x,k}^- [P_{x,k}^- + R]^{-1} \tag{20}$$

$$K_{w,k} = P_{w,k}^- C_{w,k}^T [C_{w,k} P_{w,k}^- C_{w,k}^T + R]^{-1} \tag{21}$$

5. State estimate update

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \hat{\mathbf{x}}_k^- \tag{22}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_{x,k} \boldsymbol{\epsilon}_k \tag{23}$$

$$\hat{\mathbf{w}}_k = \hat{\mathbf{w}}_k^- + K_{w,k} \boldsymbol{\epsilon}_k \tag{24}$$

6. Error covariance update

$$P_{x,k} = (I - K_{x,k}) P_{x,k}^- (I - K_{x,k})^T + K_{x,k} R K_{x,k}^T \tag{25}$$

$$P_{w,k} = (I - K_{w,k} C_{w,k}) P_{w,k}^- (I - K_{w,k} C_{w,k})^T + K_{w,k} R K_{w,k}^T \tag{26}$$

Note that the formulas presented in Section 2.1 assume the usage of an optimal Kalman gain. Here we removed this assumption by replacing (6) with the Joseph form [64] for increased numerical stability, resulting in the alternative form of Eqs. (25) and (26).

The entire implementation is conducted using JAX in Python [65]. Leveraging this framework enables easy calculation of the required jacobians (i.e. Eqs. (7) and (10)) using automatic differentiation and allows for Just In Time (JIT) compilation, leading to significant speed enhancements for the algorithms. The code necessary to replicate the findings in this paper can be accessed at,[1] along with more comprehensive result visualizations.

2.5. Extensions to the H-DEKF

2.5.1. Adaptive process- and measurement covariance estimation

Yet another assumption of the Kalman Filter which is often not met is the knowledge of the noise characteristics of both model predictions and measurements. Given the setting, as described in Section 2.2, of an unknown process model, it is apparent that knowledge of the uncertainty on your model predictions is also absent. Moreover, if the process behavior and hence the process model vary over time, the process covariance matrix, Q_x, also becomes time-varying. Hence there is need for an extension of the H-DEKF framework to allow for variable covariance estimation. [66] gives a comprehensive overview on existing methods to address this issue. In this paper we adopt the approach outlined in [67] due to its direct extension to nonlinear filtering problems, particularly EKF. Both process- and measurement covariances are thus updated iteratively at each timestep by using Eqs. (27)–(31). This update follows the error covariance update from Eqs. (25) and (26).

$$P_k^e = A_k^{-1}(\epsilon_{k+1}\epsilon_k^T + A_k K_{x,k}\epsilon_k\epsilon_k^T) \tag{27}$$

$$Q_k^e = P_k^e - A_{k-1}P_{k-1}A_{k-1}^T \tag{28}$$

$$R_k^e = \epsilon_k\epsilon_k^T - P_k^- \tag{29}$$

$$Q_{k+1} = Q_k + \delta(Q_k^e - Q_k) \tag{30}$$

$$R_{k+1} = R_k + \delta(R_k^e - R_k) \tag{31}$$

where δ is taken to be fixed at $\frac{1}{1000}$. This value has been selected to provide a stable evolution, yet fast enough to ensure convergence.

2.5.2. Ensemble H-DEKF with ensemble aggregation

Due to the nonlinear nature of the proposed H-DEKF, the optimization problem becomes susceptible to local minima in the parameter learning problem. In scenarios where prior knowledge of model parameters is lacking, or when the initial guess significantly influences parameter convergence, the probability of converging to such a local minimum is increased. To mitigate this issue, a potential solution involves initializing and evaluating the model across various regions of the parameter space. While methods such as Particle Filtering [35,68] offer a means to address this challenge, they do so at the expense of increased computational complexity. In light of real-time constraints, a more pragmatic approach was opted for, initiating an ensemble of H-DEKF instances, each operating independently.

The challenge of local minima also arises in the integration of physical models with data-driven techniques, i.e. hybrid models. Given an overparameterized data-driven estimator, e.g. neural network, joint optimization with the physical parameters can lead to poor estimates on the latter. Conventional approaches to handle this issue involve regularization of the data-driven component, as explained in Section 1. In the context of a hybrid model, however, the data-driven element often serves to compensate for unmodeled components of the physical model. Thus, minimizing this component leads to superfluous adjustments in physical parameters to compensate for this. Given a hybrid model, it is anticipated that the physically modeled component demonstrates good extrapolation properties, provided that the correct physical parameters are used. Therefore, in this paper, we assume that a well-converged model, which incorporates accurate physics, exhibits a consistently low model error (measured as innovation error ϵ) across a wide range of operating conditions and over an extended period. Consequently, the adaptive process noise covariance, Q_x, previously described, serves as an appropriate metric, as it quantifies the expected error of the model at each point in time.

Using an ensemble of M independent H-DEKF instances, a collective prediction based on this metric can be obtained for both state and parameter estimates. This is done by employing the aforementioned assumption and selecting the top-performing ensemble member based on the process uncertainty $Q_{\mathbf{x}}$. Later, in Section 4.3, various alternative aggregation functions to combine the instances within the ensemble are evaluated and compared to our current assumption. To maintain generality in the equations, we introduce the arbitrary ensemble aggregation function as $\mathcal{U}$, leading to the following formulas for updating the state and parameter estimates:

$$\hat{\mathbf{x}}_k^{\mathcal{U}} = \mathcal{U}(\hat{\mathbf{x}}_{k,1}, \hat{\mathbf{x}}_{k,2}, \dots, \hat{\mathbf{x}}_{k,M}) \tag{32}$$

$$\hat{\mathbf{w}}_k^{\mathcal{U}} = \mathcal{U}(\hat{\mathbf{w}}_{k,1}, \hat{\mathbf{w}}_{k,2}, \dots, \hat{\mathbf{w}}_{k,M}) \tag{33}$$

With

[1] https://github.com/cevheck/Physic-Informed-DEKF.

- $\hat{\mathbf{x}}_k^{\mathcal{U}}, \hat{\mathbf{w}}_k^{\mathcal{U}}$, the ensemble prediction of state and parameter vector at time k.
- $M = 10$, the size of the ensemble. Chosen as the largest value that still allows for real-time computation.

3. Implementation and data acquisition on a cam follower system

3.1. Experimental setup: A cam follower system

In this Section, the experimental setup, as depicted in Fig. 2(a), its dynamical model and the obtained datasets will be elaborated on. The setup is comprised of a load with variable mass, m, moved by a cam-follower mechanism converting the rotational movement of the driving Direct Current (DC) motor into a translational displacement. The quantities of interest within this system are the movements of both cam- and follower. These two displacements are mapped to each other by the displacement function, h. The cams studied within this research are limited to those with a cycloidal displacement function without dwell [69], and therefore are parameterized by two parameters, H and β, respectively the cam height and skewness (see Fig. 2(b)), resulting in the displacement function (34). The state can therefore be summarized in the angular position (θ) and velocity (ω) of the cam. Additionally, linear velocity ($\dot{h}$) and acceleration ($\ddot{h}$) can be derived, using Eqs. (35) and (36).

$$h = \begin{cases} \frac{H}{\beta}\theta - \frac{H}{2\pi}\sin(\frac{2\pi\theta}{\beta}), & \theta < \beta \\ H\left(1 + \frac{\beta-\theta}{2\pi-\beta} - \frac{\sin(\frac{2\pi(\beta-\theta)}{2\pi-\beta})}{2\pi}\right), & \theta \geq \beta \end{cases} \tag{34}$$

$$\dot{h} = \frac{dh}{d\theta}\omega \tag{35}$$

$$\ddot{h} = \frac{d^2h}{d\theta^2}\omega^2 + \frac{dh}{d\theta}\dot{\omega} \tag{36}$$

3.2. Dynamic model of a cam follower system

As explained above, movement of the system originates from the driving torque of the motor T_d, resulting in a rotational movement of both rotor shaft and the attached cam. Taking into account system inertia, J, and friction, T_f, this results in (37), where F_y is the vertical component of the contact force between cam and follower. This force originates from the cam pushing against the follower and vice versa. From first principle modeling we can define the force balance on the follower system (38).

$$J\dot{\omega} = T_d - T_f - F_y r \tag{37}$$

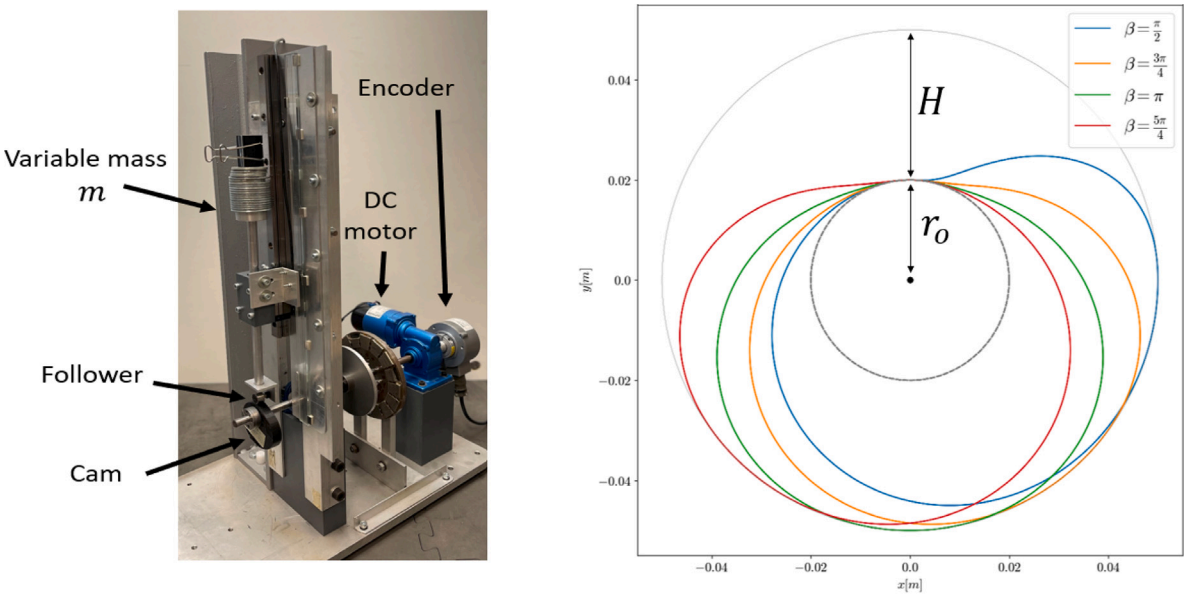


Fig. 2. (a) Experimental setup of a cam-follower mechanism driven by a DC motor. (b) Visualization of cam profile for $H = 0.03$ m and different values of β. All cam profiles have an inner radius r_0 of 0.02 m.

$$m\ddot{h}(t) = F_y(t) - mg \tag{38}$$

In the formulas above, the variable r represents the instantaneous moment arm of the y-component of the force acting perpendicular to the contact surface, defined as (39). For a comprehensive derivation of this equation, please consult Appendix A. The driving torque exerted by the motor on the camshaft is obtained from known DC motor laws [70], relating the motor torque (T_m) to the DC motor voltage (V) (Eq. (40)) along with the known gearbox ratio, τ, gearbox efficiency, ν.

$$r = dh/d\theta \tag{39}$$

$$T_m = V\frac{K\phi}{R_a} - \frac{(K\phi)^2}{R_a}\omega_m \tag{40}$$

$$T_d = \nu\ \tau\ T_m \tag{41}$$

$$\omega_m = \tau\omega \tag{42}$$

where $\nu = 0.9$, $\tau = 8$ and the motor constant $K\phi = 0.0305$. The armature resistance, R_a, in Eq. (40) is variable in function of the temperature and therefore modeled as a variable ($\in \mathbf{p}$).

Combining these equations, together with (35) and (36), the following system of ODEs is obtained:

$$\dot{\mathbf{x}} = \begin{bmatrix}\dot{\theta}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix}\omega\\ \frac{T_m - T_f - rm(\frac{d^2h}{d\theta^2}\omega^2 + g)}{J + rm\frac{\partial h}{\partial\theta}}\end{bmatrix} \tag{43}$$

where a final unknown is the frictional torque T_f. Modeling of friction phenomena is known to pose a significant challenge. A commonly used assumption posits that friction is linearly proportional to the (angular) velocity, e.g. viscous friction [18]. Contemporary alternatives tend to look at data-driven methods to model friction forces. In the continuation of this paper, both of these approaches will be examined. The linear friction assumption, characterized by a friction coefficient D (i.e. $T_f = D\omega$), results in a model derived solely from physical equations, henceforth named *physics model*. In order to obtain a better match to the real system, the friction term and inherently other unmodeled interactions are captured by the data and predicted by a neural network with weights $\boldsymbol{\alpha}$ by replacing the term T_f with z_k as described in Section 2.3. This leads to a *hybrid model*, where the inputs to the neural network, $\mathbf{q}_k$ from Fig. 1 are defined as:

$$\mathbf{q}_k = [\sin(\theta), \cos(\theta), \omega, V, H, \beta] \tag{44}$$

In a third and final approach, one could omit all information provided by the ODEs, the *neural model* directly predicts the angular acceleration, $\dot{\omega}$, using a neural network and therefore

$$\dot{\mathbf{x}} = \begin{bmatrix}\omega\\ z(\mathbf{q}, \boldsymbol{\alpha})\end{bmatrix} \tag{45}$$

An overview of the three distinct models to predict the angular velocity is depicted below, see Fig. 3. The angle itself is always predicted using the known relation $\dot{\theta} = \omega$. Therefore the selection metric $Q_{\mathbf{x}}$ is replaced by the process uncertainty on the angular velocity, Q_ω, as this component is predicted by the models. This variation is used to create the results in Section 4 and further discussed in Section 4.3.

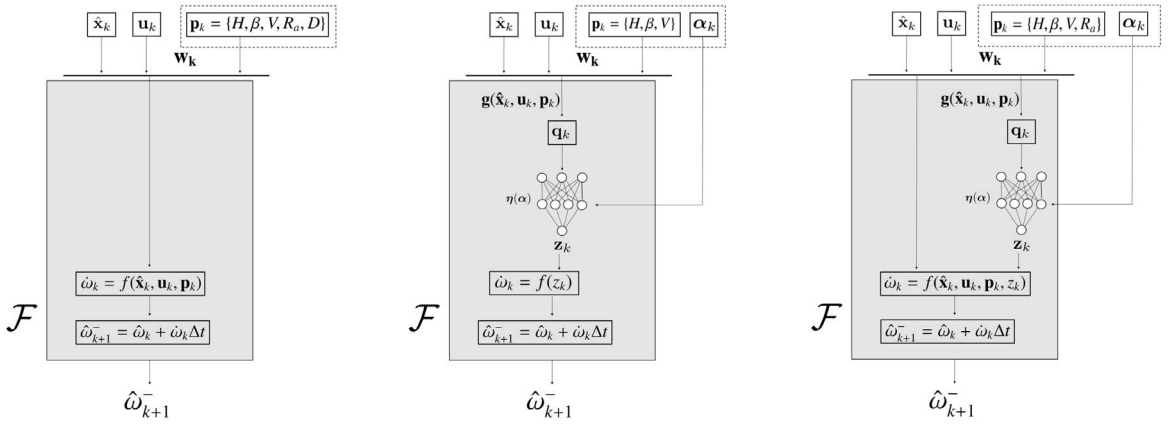


Fig. 3. Schematic view of the three model types used for prediction of the angular velocity evolution.

3.3. Dataset of a cam follower system

Previously, an extensive dataset was generated on this experimental setup, focusing on the prediction of cam detachment [10]. Various cam profiles were fabricated, encompassing a full factorial of cam heights $H \in \{0.01, 0.03, 0.05, 0.07\}$ m and skewnesses $\beta \in \{\pi/2, 3\pi/4, \pi, 5\pi/4\}$. Each profile underwent testing across different voltage levels and attached masses, hence resulting in multiple distinct runs for each of the profiles, as seen in . For further insights into the data campaign, please refer to [10]. In the scope of this paper, the primary goal is retrieving correct physical parameter convergence within an online setting. Therefore the cam-profile parameters (H, β) and the applied DC motor voltage (V) are treated as unknowns in the model. Collectively they constitute the parameter vector $\mathbf{p} = \{H, \beta, V\}$. In the case of both *hybrid model* and *physics model*, the armature resistance (R_a) is added to this vector. Moreover, for the *physics model*, the damping coefficient (D) is employed to approximate friction, therefore also becoming a part of $\mathbf{p}$. Utilizing the dual filtering framework, these values are estimated, allowing for comparison with their ground truth for the initial three parameters. As this research does not address the identification of cam detachment, the focus remains on nominal behavior, where cam detachment is not considered. The subset of nominal data from the dataset is extracted and is available on our Github page. A resulting dataset of 548 experiments, each 60 s downsampled at $dt = 0.005$ s is obtained with the following distribution:

Table 1
Dataset distribution given a subselection of nominal operating points.

H \ β	$\pi/2$	$\frac{3*\pi}{4}$	π	$\frac{5*\pi}{4}$
0.01	48	48	48	48
0.03	44	47	48	38
0.05	31	37	30	22
0.07	17	17	15	10

4. Results

This Section evaluates the proposed method through two main objectives. Firstly the filtering performance is assessed, quantified by the innovation error ϵ (Eq. (22)) together with the quality of the converged model, quantified by the multi-step prediction performance on the dataset. Subsequently, the primary concern of this paper, online detection of the physical parameters, is analyzed. These objectives are examined on all proposed model variants outlined in Section 3.2 and compared. In Section 4.3, these results are repeated for different ensemble aggregation functions, $\mathcal{U}$. We compare the previously established selection criterion, as outlined in Section 2.5.2, with alternative approaches and justify our decision. Finally, the Section concludes with a brief discussion on the obtained results.

4.1. Case study: Filtering experiments

The presented H-DEKF is evaluated when a new, unknown, cam profile is placed into production with unknown operating conditions (driving voltage, temperature). It is assumed historic data on the cams from is available with known parameters, i.e. these cams have already been in production. A *hybrid model* is trained on this data and is used as a starting point, similar to transfer learning methodologies [27]. Afterwards one of the cams is placed into production and the previously described ensemble of pre-trained H-DEKF (H-PDEKF) is used to both filter the state and learn the new model parameters $\mathbf{w} = \{\mathbf{p}, \boldsymbol{\alpha}\}$. To provide a comparative analysis, two alternative (non-hybrid) approaches, namely the *physics model* and the *neural model*, are pre-trained, updated and evaluated analogously. These models result in the p-PDEKF and n-PDEKF, respectively. The resulting time evolution of state- and parameter estimation of this proposed case study, wherein a globally trained model undergoes real-time adaptation using data from a single experiment, yields the outcomes elaborated upon in the subsequent Section 4.2.

For all of the methods, the neural network structure and Kalman filter variables need to be defined. The neural network is taken to be a multilayer perceptron with a single hidden layer of 32 neurons. The activation function at hidden- and output layer are respectively Mish [71] and linear. As mentioned above, the networks are pre-trained as a starting point for the filter, therefore only limited training is performed. Thanks to the limited pretraining and small network size, overfitting was neither expected nor observed, as confirmed by a 70–30 training-validation split. If necessary, techniques such as dropout or L2-regularization could be employed to further reduce overfitting, the author refers to [72] for a recent survey on regularization in neural networks.

For the Kalman filter variables, the initial state is assigned an arbitrary mean with a large (uncertain) covariance matrix, reflecting high initial uncertainty that decreases over time. For the state- and measurement noise, initial values are required despite their dynamic adjustments (recall Section 2.5.1). Since these values are still dynamically adapted, an informed initial guess suffices without the need for extensive tuning. Early large model errors, due to unadapted model parameters, justify a large initial process uncertainty, while measurement uncertainty is set small, reflecting the expected accuracy of encoder measurements.

- $x_0 = [\pi, 10]^T$
- $P_{x,0} = \text{diag}(100, 100)$
- $Q_{x,0} = \text{diag}(0.05\text{d}t, 0.05)$
- $R_0 = \text{diag}(0.001\text{d}t, 0.001)$

For other variables a grid search was performed to ensure optimal performance for each method individually. An overview is seen below.

- $P_{w,0}$ =diag($P_{\alpha,0}, P_{R_a,0}, P_{H,0}, P_{\beta,0}, P_{V,0}$)
- Q_w =diag($Q_\alpha, Q_{R_a}, Q_H, Q_\beta, Q_V$)

For each physical parameter, we search for an optimal value on a logarithmic scale, for both the initial uncertainty and the process noise. Neural network parameters share a single tunable value to avoid an impractically large grid search. The grid search is performed on a subset of , where 20 experiments are selected randomly with equal representation of possible H, β and m values. Optimal parameters are selected based on parameter convergence errors and filter innovation errors. For more details and the accompanying results, we refer to our Github page. A noteworthy hyperparameter proved to be Q_α. For the results in the upcoming Section, the parameter has been set to 10^{-12}. This choice has been made to favor the convergence on the physical parameters, as discussed in 4.4, where the importance of this parameter will be explored in depth.

4.2. Experimental results

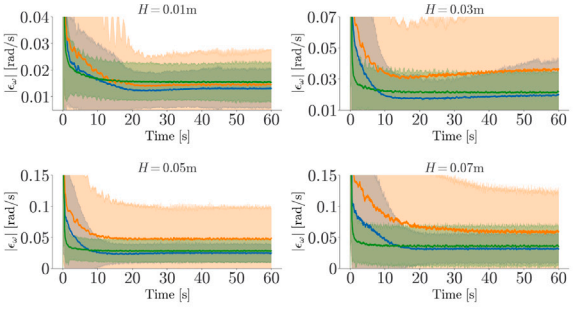
4.2.1. Filtering performance

As a first evaluation of the proposed methodology, the ability to track the system's angular velocity is analyzed. Therefore the state filtering results are examined and compared over the different strategies. Fig. 4(a) shows tracking performance, quantified by the innovation error ϵ_ω. Each method's filtering efficacy is assessed across all instances of the experimental data outlined in . The graph displays the average absolute deviation of predictions, smoothed using a moving window of 100 samples (0.5 s) for enhanced clarity. The results are presented in four separate plots, each plot representing a subset of the data based on cam height H, thus encapsulating varied dynamic behaviors. All results exhibit similar trends and one can see that fast convergence is achieved. As time progresses, tracking performance improves as the model adapts to the system's dynamics. The results show that the regularization embedded in the H-PDEKF, achieved through constraining the model with the physical ODEs, has a small positive impact on filtering performance. Moreover, the additional modeling degrees of freedom compared to the p-PDEKF show their merit by outperforming this model substantially.

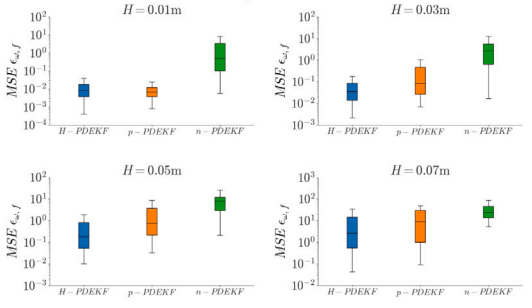
At the end of the filtering stage, the converged model is assessed by evaluating its overall predictive performance on the current experiment. Fig. 4(b) illustrates the Mean Squared Error (MSE) on the angular velocities, obtained from repeated open-loop multi-step ahead predictions spanning 1000 samples, denoted as $\epsilon_{\omega,f}$ (i.e. the model estimate is reset to the measurement every 1000 samples to prevent divergence). These results affirm that the proposed methodology obtains the best model representation. Especially the difference with the data-driven model is remarkable. Despite having similar results as the n-PDEKF during filtering (recall Fig. 4(a)), this analysis demonstrates that a superior and more robust model is attained compared to the n-PDEKF. The latter exhibits diverging predictions due to the absence of physical constraints, highlighting the advantageous regularizing effect imposed by the system equations. For a more comprehensive analysis in function of the prediction horizon, j, and the hyperparameter Q_{alpha}, please refer to Appendix B. Similar results show that the hybrid model consistently scores best on multi-step predictions.

4.2.2. Online detection of physical parameters

Section 4.2.1 showed that a better global model is achieved in the case of the H-PDEKF. This outcome is likely attributed to its effective convergence on the physical model, ensuring robust performance across diverse operating conditions. To validate this

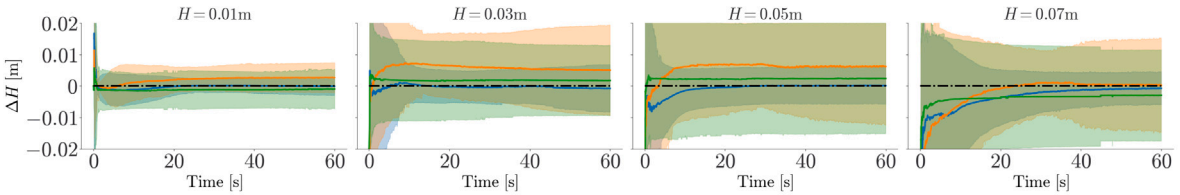


(a) Average absolute innovation error during filtering for different cam heights. Results are displayed with a moving window average of 100 samples and 95% confidence bounds.

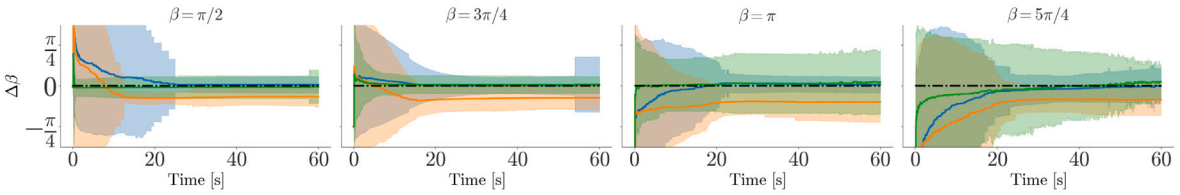


(b) MSE of the final converged models, evaluated on a repeated forward prediction of 1000 samples or 5s.

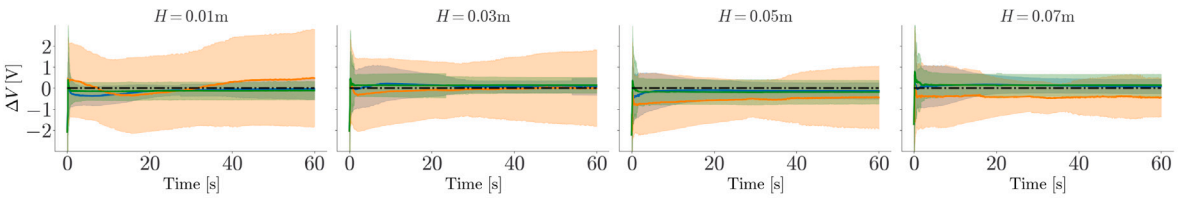
Fig. 4. Model performance metrics on angular velocity predictions. (a) during filtering (b) on the converged model. •: H-PDEKF, •: p-PDEKF, •: n-PDEKF.



(a) Parameter tracking on the cam height H.



(b) Parameter tracking on the cam scewness β.



(c) Parameter tracking on the operating voltage V.

Fig. 5. Physical parameter tracking. Average results are depicted with 95% confidence bounds. •: H-PDEKF, •: p-PDEKF, •: n-PDEKF, ⋅ ⋅: $\mathbf{p}_{\text{true}}$.

hypothesis, this Section compares the parameter convergence to the ground-truth values for the parameters H, β, and V. Fig. 5 illustrates the temporal evolution of the error on these variables. In Fig. 5(a) each subplot presents the results from a subset of data experiments categorized by cam height, H, as indicated in the title. The average prediction error and 95% confidence bounds are depicted for each subset. A similar approach is wielded for Fig. 5(b). However, Fig. 5(c) introduces a slight modification. Given the 12 distinct voltage setpoints, the plot is not shown for each individual voltage subset. Instead, it is displayed based on the different subselections of data corresponding to the cam height.

The results indicate that, on average, the proposed *hybrid* method delivers the best and most reliable estimates, exhibiting both the lowest errors and the lowest standard deviations in Fig. 5. This is followed by the *neural* method. The *physical* model, however, continues to be outperformed, demonstrating once more that incorrect adjustments are being made to the physical estimates to align them with the measurements and to compensate unmodeled phenomena. It is important to note that this analysis was conducted using a specific value (1e−12) of the hyperparameter Q_α. Section 4.4 offers a comparison of the methods across a range of this hyperparameter.

4.3. Evaluating different ensemble aggregation methods

In this part, multiple metrics to combine the different ensemble members to a joint prediction of both state and system parameters are discussed. The metrics are focused on the angular velocity, ω, given that the prediction model of the other state variable, θ is trivial (see (43)). The following list of selection methods, $\mathcal{U}$, are compared

- Q_ω: The ensemble member with the lowest process uncertainty, Q_ω is taken as the sole dominating predictor.
- $|\epsilon_\omega|$: The ensemble member with the lowest absolute innovation error, $|\epsilon_\omega|$ is taken as the sole dominating predictor.
- $\mathcal{A}$: The combined prediction of the ensemble is calculated as the average of each ensemble member's prediction.
- $\mathcal{WA}$: The combined prediction of the ensemble is calculated as the weighted average of each ensemble member's prediction. Weights are determined by (46).
- $\mathcal{A}(Q_{\omega,3})$: The combined prediction of the ensemble is calculated by first selecting the best 3 ensemble members, based on the process uncertainty Q_ω. Afterwards the average $\mathcal{A}$ is calculated.

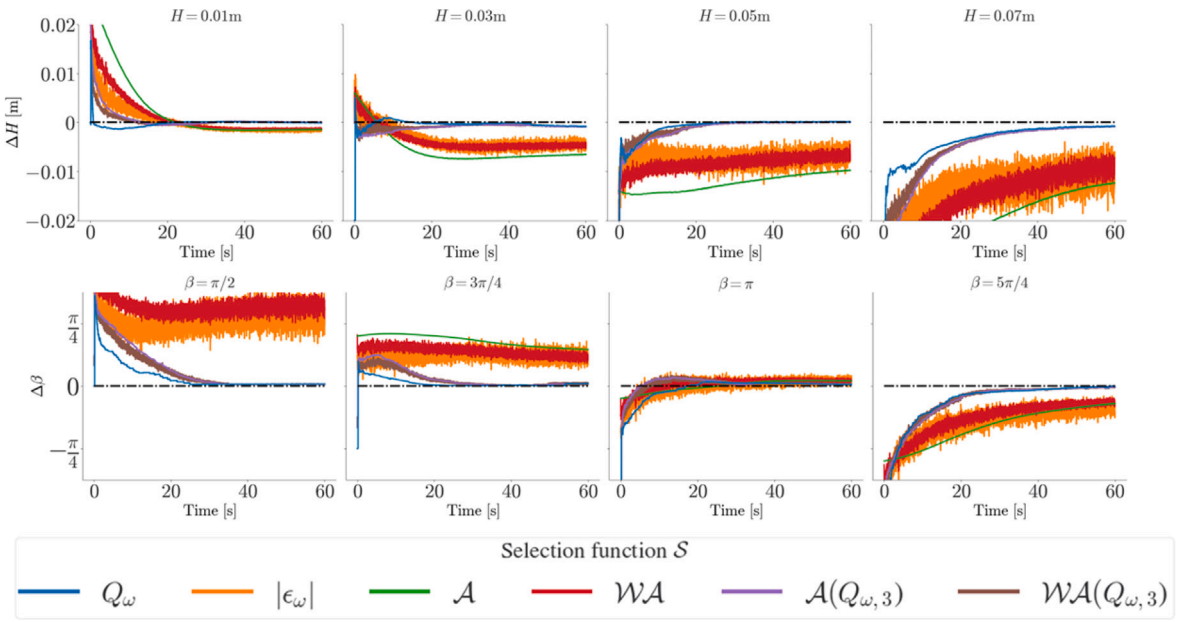


Fig. 6. Evaluating different metrics to combine the H-PDEKF ensemble parameter estimates to a joint belief.

- $\mathcal{WA}(Q_{\omega,3})$: The combined prediction of the ensemble is calculated by first selecting the best 3 ensemble members, based on the process uncertainty Q_ω. Afterwards the weighted average $\mathcal{WA}$ is calculated.

$$\lambda_{i,n} = \frac{\lambda_i}{\sum \lambda_i} \qquad \text{, where } \lambda_i = \frac{1}{|\epsilon_{\omega,i}|} \tag{46}$$

Fig. 6 illustrates the H-PDEKF ensemble prediction of both H and β (similar to Fig. 5), given the different selection methods. Standard deviations on the predictions are omitted to maintain clarity. The version incorporating standard deviations, along with the results for ΔV, are available and reproducible through the shared Github page. Results clearly indicate that the outcomes are significantly influenced by the employed metric. Furthermore, selecting based on Q_ω proves to be an efficient strategy, where the utilization of fewer ensemble members based on this metric generally outperforms averaging methods. Therefore, as already mentioned in Section 2.5.2, this metric was opted for in the previous results (Section 4.2).

An additional observation is that utilizing a sole ensemble member results in an average performance identical to that of $\mathcal{A}$ (when repeating the simulations for each ensemble member), albeit with substantially larger standard deviations attributed to the wide variance in potential outcomes when one would sample a random ensemble member compared to utilizing the average output. Consequently, this observation substantiates the necessity of the proposed ensembling method.

4.4. Discussion

This Section demonstrated the suitability of the DEKF-framework for online filtering and model learning due to its rapid convergence in both tracking error and parameter estimation. When comparing the various models employed within the DEKF, results show that there is an inherent trade-off in model design. Physics-based models lack the capacity to describe the real system behavior. Therefore not only model quality and tracking performance will be inadequate (refer to Figs. 4(a) and 4(b)), moreover physical parameters will be altered to incorrect values in order to compensate for the errors in the unmodeled or insufficiently modeled phenomena (see Fig. 5). Although improvements can be achieved with sufficient experience and modeling effort, certain systems and phenomena still pose challenges for accurate modeling. Purely data driven models (i.e. n-PDEKF) offer the most modeling flexibility, whilst resulting in lower tracking errors. Additionally, by using physical parameters as input to the neural network, an estimate of these parameters can be updated and, given a good initial training set, ground truth physical parameters can even be retrieved in such models (see Fig. 5). However, their lack of adherence to physical constraints leads to less accurate parameter estimation and unstable multi-step predictions. The drawbacks of both methods are circumvented by incorporating a

data-driven component into a physically regularized framework, as observed in the results of the H-PDEKF. Fig. 5 shows consistently better and more narrow predictions on the physical parameters, where the average final convergence accuracy increases with 60.36% compared to the n-PDEKF and 85.33% to the p-PDEKF.[2] The obtained model also exhibits better long term (multi-step) prediction capabilities, as shown in Fig. 4(b), outperforming the n-PDEKF and p-PDEKF with 99.22% and 48.80% respectively.[3] In this *hybrid* model however, the interplay between the neural and physical part deemed crucial to obtaining good results. Each component can potentially compensate for deficiencies in the other, resulting in two key challenges.

A first problem is the presence of local minima, mitigated by using an ensemble method and subselection based on the process uncertainty Q_ω, as elaborated in Section 2.5.2. These results highlight the necessity of adopting an ensemble technique with appropriate aggregation function $\mathcal{U}$. Due to the presence of local minima, model parameter convergence may result in highly varying estimates across the different ensemble members. Averaging out these minima does not yield a reliable approach, as (some of) the models may be trapped in suboptimal solutions. Therefore, selecting the single best-scoring ensemble member based on the selected metric proves to be a more suitable strategy. This approach relies on the assumption that one or a few of the identified local minima likely represent more favorable solutions, mitigating the impact of suboptimal convergence paths.

A second problem becomes apparent upon further investigation of the hyperparameter sensitivities, especially the uncertainty of the neural network parameters, Q_α. This parameter intuitively governs the extent to which the neural parameters compensate for errors. Fig. 7 depicts the relationship of both parameter error and process uncertainty (on ω) as the hyperparameter is adjusted logarithmically from 1e−14 to 1e−5. The procedure described in Fig. 5 is repeated for various values of Q_α. To address discrepancies in the error scales, the geometric mean is computed across all simulations and different physical parameters. This is then compared to the process uncertainty, also represented as a geometric mean. Fig. 7 illustrates the correlation between physical parameter estimation error and process uncertainty, averaged across the parameters H, β, and V. The direction of increasing Q_α is indicated by the arrows on the plotting line. This clearly shows that, for our use case, an increase in Q_α leads to an increase in physical parameter errors. On the contrary, the process uncertainty generally has a downward trend in function of this parameter. Consequently, a user-defined trade-off emerges, akin to a Pareto front, determined by the design choice of Q_α and dependent on the relative importance of state tracking to parameter estimation. Higher values of Q_α amplify the neural network's sensitivity to errors, leading it to assume more responsibility. This increased sensitivity may cause the neural network to overcompensate for changes that should be reflected by the physical parameters, resulting in poorer physical parameter estimates when Q_α is excessively high. However, this heightened reactivity of the neural network parameters also enhances the overall adaptivity of the model, reducing innovation errors and consequently lowering process uncertainty. Important to note is that, although higher physical errors are obtained with increased freedom in the neural network, the presence of the neural network notably enhances physical parameter estimates compared to the p-DEKF (recall Fig. 5). This improvement occurs because the p-PDEKF erroneously adjusts the physical parameters to compensate for unmodeled errors, whilst these errors are captured by the neural network within the hybrid model. Furthermore, when compared to the n-PDEKF, the incorporation of physical relations in the form of ODEs proves to be crucial, embedding meaningful relationships and gradients in the model, thereby allowing to achieve more accurate physical parameter estimates. On the other hand, when physical parameter estimation is of less importance, the *neural* method shows the ability to outperform the *hybrid* method based on the previously defined process uncertainty metric. Note that this metric is proportional to the innovation error and therefore the one-step ahead prediction error (see Eqs. (27)–(31)). If we compare the models based on the multi-step prediction error (see Appendix B), the hybrid model consistently performs better in multi-step predictions, regardless of the values of Q_α. This further suggests that hybrid models generally outperform purely data-driven methods in long-term predictions.

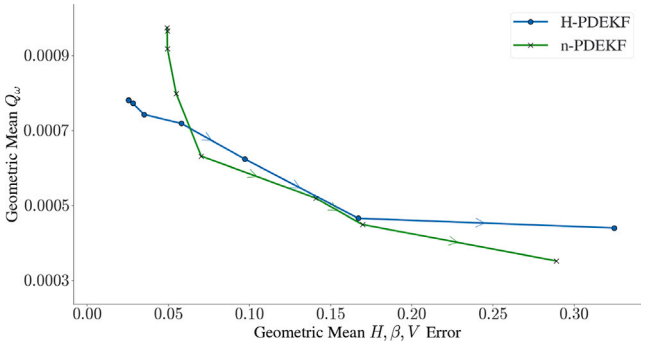


Fig. 7. Average parameter tracking results in function of the process uncertainty, Q_ω for different values of the hyperparameter Q_α. Arrows on the plotting line indicate the direction of increasing Q_α.

[2] Let $MAEE(p_{j,i})$ be defined as the mean absolute error of variable p_j (averaged over all 548 experiments) as estimated by model i at the end of the filtering stage (Time = 60s). Percentages are calculated as $\frac{1}{3}\sum_{p_j\in\{H,\beta,V\}} \frac{MAEE(p_{j,hybrid})-MAEE(p_{j,i})}{MAEE(p_{j,i})}$ for $i \in \{neural, physics\}$.

[3] Let $MMSE_{j,i}$ be defined as the mean of the multi-step MSEs on experiments with $H = j$ as predicted by model i (see Fig. 4(b)), with $j \in \mathcal{J} = \{0.01, 0.03, 0.05, 0.07\}$. Percentages are calculated as $\frac{1}{4}\sum_{j\in\mathcal{J}} \frac{MMSE_{j,hybrid}-MMSE_{j,i}}{MMSEj,i}$ for $i \in \{neural, physics\}$.

5. Conclusion

In this paper we proposed an ensemble of hybrid, pretrained, dual extended Kalman filters (H-PDEKF). The framework leverages on properties from Kalman filtering to process sensor data in real-time. It allows for online model identification and adaptation, concurrent with accurate filtering of the system's state. Experimental results on a cam follower system demonstrate that the hybrid model achieves significantly better convergence of physical parameters, outperforming physical- and neural models by more than 60%. Although closely matched by the data-driven approach in the context of instantaneous filtering, the attained model shows better extrapolation and robust predictions. The hybrid model's advantages stem from its ability to adapt to unmodeled phenomena through the neural network while ensuring physically plausible outputs by integrating the neural network into a physically bounded and informed equation. Main limitations of the H-PDEKF are the importance of initialization and the need for hyperparameter tuning concerning the concurrent physical and data-driven parts. We addressed the initialization issue by introducing a metric for ensemble subselection to avoid local minima inherent to the joint optimization problem of a hybrid model. For hyperparameter tuning, a case-specific trade-off is required in order to trade neural network freedom against physical enforcement to achieve the desired balance between filtering- and physical tracking performance. Future work could explore replacing the grid search for parameter tuning with adaptive estimation of parameter uncertainties, akin to the adaptive handling of process uncertainty, or employ other Kalman filter variants, such as the unscented Kalman filter (UKF), to better manage strong nonlinearities. However, these approaches may increase computational complexity, posing challenges for real-time applications in high-rate systems.

CRediT authorship contribution statement

Van Heck Cedric: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Vandeputte Matthias:** Writing – review & editing, Methodology, Conceptualization. **Coene Annelies:** Writing – review & editing, Funding acquisition, Conceptualization. **Crevecoeur Guillaume:** Writing – review & editing, Resources, Funding acquisition, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used ChatGPT in order to refine wording and improve grammatical clarity. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Flanders Make project QUASIMO and the Flanders AI Research Programme, Belgium. A.C. is supported by the Research Foundation - Flanders (FWO), Belgium through a senior postdoctoral fellowship (12Z4722N). This work continuous on the publication of W. De Groote and makes use of the data produced in [10].

Appendix A. Instantaneous moment arm derivation

The derivation of the instantaneous moment arm, r, follows from Pythagoras' law in the triangle created with the force, F, acting perpendicular to the cam profile and the line connecting the cams center of rotation (at position (0,0)) with the point of contact, assuming an infinitesimal contact point at position $(0, R(\theta, 0°))$. R is, by design, the sum of the base circle, r_0 and the displacement function, $h(\theta+\psi)$, where ψ denotes the polar angle and θ the rotation of the cam w.r.t. the initial orientation $\theta_0 = 0$. The situation is sketched in Fig. A.8. The $(\overline{x}, \overline{y})$ position of each point on the cam circumference is given by

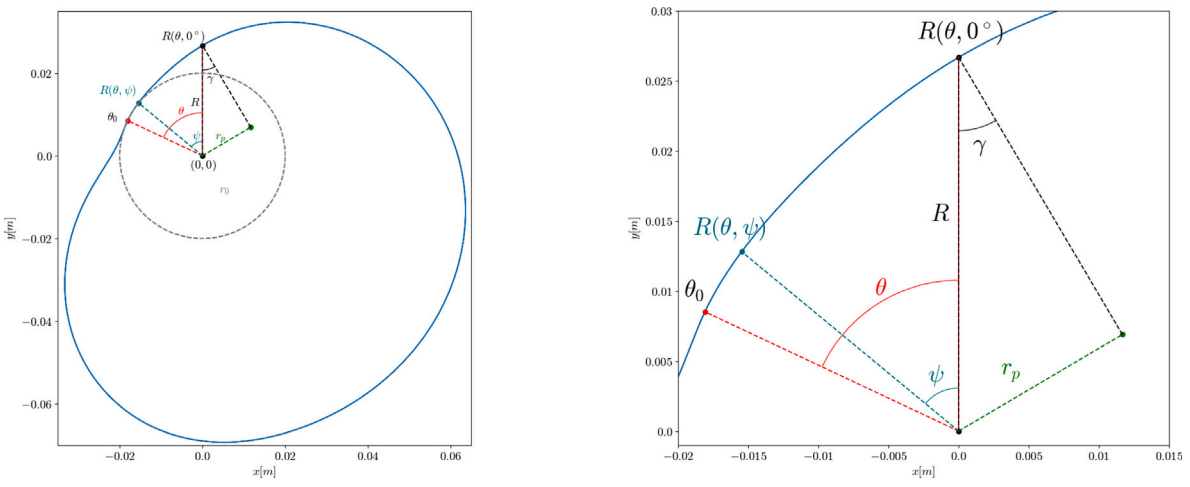
$$\begin{aligned} R(\theta+\psi) &= r_0 + h(\theta+\psi) \\ \overline{x} &= R(\theta+\psi)\sin(\psi) \\ \overline{y} &= R(\theta+\psi)\cos(\psi) \end{aligned} \tag{A.1}$$

The slope of the cam profile can therefore be written as

$$\frac{dy}{dx} = \frac{-R(\theta+\psi)\sin(\psi) + dh/d\theta\cos(\psi)}{R(\theta+\psi)\cos(\psi) + dh/d\theta\sin(\psi)} \tag{A.2}$$

Given that we are interested in the point of contact, situated at $\psi = 0$, the slope, and thus the orientation perpendicular to the F, becomes

$$\frac{dy}{dx} = \frac{dh/d\theta}{R(\theta, 0°)} \triangleq \frac{dh/d\theta}{R} \tag{A.3}$$



(a) (b)

Fig. A.8. Cam profile and relevant quantities for a cam with geometrical parameters $H = 0.05$, $\beta = \frac{5\pi}{4}$. (a) shows the full cam profile, whilst (b) shows a clear zoom.

The angle of the slope, γ, and the moment arm of the perpendicular force, r_p are now derived to be

$$\begin{aligned} \gamma &= \tan^{-1}(dy/dx) \\ r_p &= R\sin(\gamma) \end{aligned} \tag{A.4}$$

However, Eqs. (37) and (38) only use the y-component of the force and therefore should be adjusted to lead us to the final formula

$$\begin{aligned} & F_y = F\cos(\gamma) \\ \Leftrightarrow & F_y r = F\cos(\gamma) r = F r_p \\ \Leftrightarrow & r = \frac{r_p}{\cos(\gamma)} \\ \Leftrightarrow & r = R\tan(\gamma) \\ \Leftrightarrow & r = dh/d\theta \end{aligned} \tag{A.5}$$

Appendix B. Multi-step ahead comparison

In Fig. 4(b) we compared the j-step ahead prediction for a given value of j (= 1000) and Q_{alpha} (= 1e−12). In this Section we validate that the conclusions can be extended for all values of j and Q_{alpha}. Fig. B.9 shows the results in function of the prediction horizon, j. For the *hybrid* and *neural* models, multiple realizations of the model are depicted, each representing a chosen value of Q_{alpha}. A general trend becomes clear, where the *hybrid* model consistently obtains the lowest errors, independent of the selected value of Q_{alpha}.

Data availability

I have shared data and reproducing code using a Github link. The code can be found at https://github.com/cevheck/Physic-Informed-DEKF.

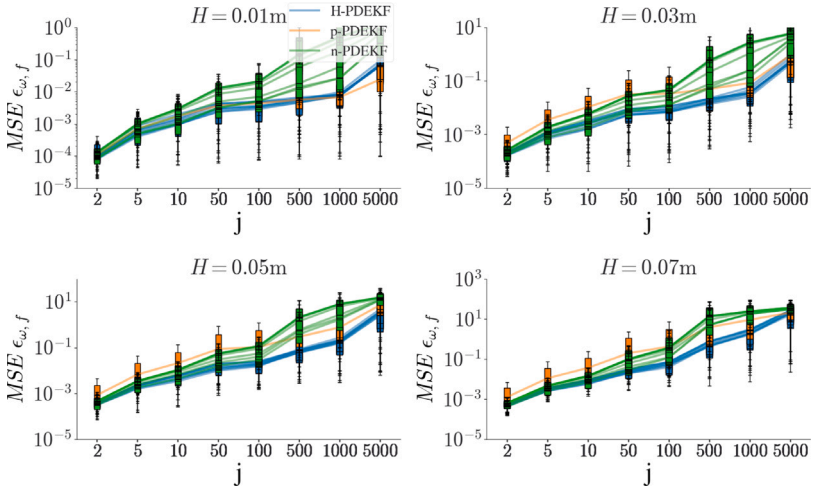


Fig. B.9. MSE of the final converged models, evaluated on a repeated forward prediction of j samples and for multiple values of Q_{alpha}.

References

[1] D.H.B. de Sousa, F.R. Lopes, A.W. do Lago, M.A. Meggiolaro, H.V.H. Ayala, Hybrid gray and black-box nonlinear system identification of an elastomer joint flexible robotic manipulator, Mech. Syst. Signal Process. 200 (2023) 110405, http://dx.doi.org/10.1016/j.ymssp.2023.110405.

[2] W. Guo, C. He, P. Shao, A novel system identification method for servo-hydraulic shaking table using physics-guided long short-term memory network, Mech. Syst. Signal Process. 178 (2022) 109277, http://dx.doi.org/10.1016/j.ymssp.2022.109277.

[3] J. Awrejcewicz, Ordinary Differential Equations and Mechanical Systems, Springer, Cham, 2014, http://dx.doi.org/10.1007/978-3-319-07659-1.

[4] J. Ma, S. Dong, G. Chen, P. Peng, L. Qian, A data-driven normal contact force model based on artificial neural network for complex contacting surfaces, Mech. Syst. Signal Process. 156 (2021) 107612, http://dx.doi.org/10.1016/j.ymssp.2021.107612.

[5] S. Mariani, A. Kalantari, R. Kromanis, A. Marzani, Data-driven modeling of long temperature time-series to capture the thermal behavior of bridges for SHM purposes, Mech. Syst. Signal Process. 206 (2024) 110934, http://dx.doi.org/10.1016/j.ymssp.2023.110934.

[6] M. Kitahara, Y. Kakiuchi, Y. Yang, T. Nagayama, Adaptive Bayesian filter with data-driven sparse state space model for seismic response estimation, Mech. Syst. Signal Process. 208 (2024) 111048, http://dx.doi.org/10.1016/j.ymssp.2023.111048.

[7] W. De Groote, E. Kikken, E. Hostens, S. Van Hoecke, G. Crevecoeur, Neural network augmented physics models for systems with partially unknown dynamics : application to slider-crank mechanism, IEEE- ASME Trans. Mechatronics 27 (2021) 103–114, http://dx.doi.org/10.1109/TMECH.2021.3058536.

[8] W. De Groote, S. Van Hoecke, G. Crevecoeur, Physics-based neural network models for prediction of cam-follower dynamics beyond nominal operations, IEEE/ASME Trans. Mechatronics 27 (2022) 2345–2355, http://dx.doi.org/10.1109/TMECH.2021.3101420.

[9] S. Cuomo, V.S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics–Informed neural networks: Where we are and what's next, J. Sci. Comput. 92 (2022) 88, http://dx.doi.org/10.1007/s10915-022-01939-z.

[10] W. De Groote, S. Van Hoecke, G. Crevecoeur, Prediction of follower jumps in cam-follower mechanisms: The benefit of using physics-inspired features in recurrent neural networks, Mech. Syst. Signal Process. 166 (2022) 108453, http://dx.doi.org/10.1016/j.ymssp.2021.108453.

[11] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707, http://dx.doi.org/10.1016/j.jcp.2018.10.045.

[12] R.T.Q. Chen, Y. Rubanova, J. Bettencourt, D.K. Duvenaud, Neural ordinary differential equations, Conf. Neural Inf. Process. Syst. 31 (2018) 6572–6583, arXiv:1806.07366.

[13] M. Lutter, J. Peters, Combining physics and deep learning to learn continuous-time dynamics models, Int. J. Robot. Res. 42 (2023) 027836492311694, http://dx.doi.org/10.1177/02783649231169492.

[14] S. Goswami, M. Yin, Y. Yu, G.E. Karniadakis, A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials, Comput. Methods Appl. Mech. Engrg. 391 (2022) 114587, http://dx.doi.org/10.1016/j.cma.2022.114587.

[15] K. Hanjalić, B. Launder, Modelling Turbulence in Engineering and the Environment: Second-Moment Routes to Closure, Cambridge University Press, Cambridge, 2011, http://dx.doi.org/10.1017/CBO9781139013314.

[16] M. Wu, J. Wang, Prediction of 3D contact force chains using artificial neural networks, Eng. Geol. 296 (2022) 106444, http://dx.doi.org/10.1016/j.enggeo.2021.106444.

[17] Y. Wan, T. Dodd, C. Wong, R. Harrison, K. Worden, Kernel based modelling of friction dynamics, Mech. Syst. Signal Process. 22 (2008) 66–80, http://dx.doi.org/10.1016/j.ymssp.2007.07.014.

[18] A. Shaw, G. Gatti, P. Gonçalves, B. Tang, M. Brennan, Frictional phenomena within a quasi zero stiffness vibration device, Mech. Syst. Signal Process. 211 (2024) 111113, http://dx.doi.org/10.1016/j.ymssp.2024.111113.

[19] V. Mehta, I. Char, W. Neiswanger, Y. Chung, A. Nelson, M. Boyer, E. Kolemen, J. Schneider, Neural dynamical systems: Balancing structure and flexibility in physical prediction, IEEE Conf. Decis. Control. (CDC) (2021) 3735–3742, http://dx.doi.org/10.1109/CDC45484.2021.9682807.

[20] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, Int. Conf. Learn. Represent. (2014) http://dx.doi.org/10.48550/arXiv.1412.6980.

[21] S. Le Guisquet, M. Amabili, Identification by means of a genetic algorithm of nonlinear damping and stiffness of continuous structures subjected to large-amplitude vibrations. part I: single-degree-of-freedom responses, Mech. Syst. Signal Process. 153 (2021) 107470, http://dx.doi.org/10.1016/j.ymssp.2020.107470.

[22] B. Lu, C. Moya, G. Lin, NSGA-PINN: A multi-objective optimization method for physics-informed neural network training, Algorithms 16 (4) (2023) http://dx.doi.org/10.3390/a16040194.

[23] Y. Yin, V.L. Guen, J. Dona, E. de Bézenac, I. Ayed, N. Thome, P. Gallinari, Augmenting physical models with deep networks for complex dynamics forecasting, J. Stat. Mech. Theory Exp. 2021 (2021) 124012, http://dx.doi.org/10.1088/1742-5468/ac3ae5.

[24] J. Winz, S. Engell, Reliable nonlinear dynamic gray-box modeling by regularized training data estimation and sensitivity analysis, IFAC- Pap. 55 (2022) 86–93, http://dx.doi.org/10.1016/j.ifacol.2022.07.426.
[25] R.E. Kalman, A new approach to linear filtering and prediction problems, Trans. ASME– J. Basic Eng. 82 (Series D) (1960) 35–45, http://dx.doi.org/10.1115/1.3662552.
[26] T. Diethe, T. Borchert, E. Thereska, B. Balle, N. Lawrence, Continual learning in practice, NeurIPS 2018 Work. Contin. Learn. (2018) http://dx.doi.org/10.48550/arXiv.1903.05202.
[27] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He, A comprehensive survey on transfer learning, Proc. IEEE 109 (2021) 43–76, http://dx.doi.org/10.1109/JPROC.2020.3004555.
[28] P. Werbos, Applications of advances in nonlinear sensitivity analysis, in: R. Drenick, F. Kozin (Eds.), System Modeling and Optimization, in: Lecture Notes in Control and Information Sciences, Springer, Heidelberg, 1982, pp. 762–770, http://dx.doi.org/10.1007/BFb0006203.
[29] L. Luttmann, P. Mercorelli, Comparison of backpropagation and Kalman filter-based training for neural networks, Int. Conf. Syst. Theory Control. Comput. (ICSTCC) (2021) 234–241, http://dx.doi.org/10.1109/ICSTCC52150.2021.9607274.
[30] S. Singhal, L. Wu, Training multilayer perceptrons with the extended Kalman algorithm, Conf. Neural Inf. Process. Syst. 1 (1989) 133–140.
[31] E. Wan, A. Nelson, Dual Kalman filtering methods for nonlinear prediction, smoothing, and estimation, Adv. Neural Inf. Process. Syst. 9 (1996).
[32] A. Chernodub, Training dynamic neural networks using the extended Kalman filter for multi-step-ahead predictions, Artificial Neural Networks, Springer, Cham, 2015, pp. 221–243, http://dx.doi.org/10.1007/978-3-319-09903-3_11,
[33] S. Haykin, et al., Kalman Filtering and Neural Networks, John Wiley & Sons, Ltd, New Jersey, 2001, http://dx.doi.org/10.1002/0471221546.
[34] Y. Ye, Z. Li, J. Lin, X. Wang, State-of-charge estimation with adaptive extended Kalman filter and extended stochastic gradient algorithm for lithium-ion batteries, J. Energy Storage 47 (2022) 103611, http://dx.doi.org/10.1016/j.est.2021.103611.
[35] T. Yu, Z. Wang, J. Wang, An iterative augmented unscented Kalman filter for simultaneous state-parameter-input estimation for systems with/without direct feedthrough, Mech. Syst. Signal Process. 205 (2023) 110793, http://dx.doi.org/10.1016/j.ymssp.2023.110793.
[36] H. Moradkhani, S. Sorooshian, H.V. Gupta, P.R. Houser, Dual state–parameter estimation of hydrological models using ensemble Kalman filter, Adv. Water Resour. 28 (2) (2005) 135–147, http://dx.doi.org/10.1016/j.advwatres.2004.09.002.
[37] W. Shu, Z. Zheng, Neural dual particle filter and its application in speech enhancement, 2005, pp. 451–454, http://dx.doi.org/10.1109/SIPS.2005.1579911.
[38] P. Nguyen Phuc, D. Bozalakov, H. Vansompel, K. Stockman, G. Crevecoeur, Rotor temperature virtual sensing for induction machines using a lumped-parameter thermal network and dual Kalman filtering, IEEE Trans. Energy Convers. 36 (2021) 1688–1699, http://dx.doi.org/10.1109/TEC.2021.3060478.
[39] A. De Keyser, D. Stroobandt, G. Crevecoeur, Adaptive state space representations enabling reliable and robust decision-making in asynchronous drives for mechatronic applications, in: IEEE International Conference on Advanced Intelligent Mechatronics (AIM), 2017, pp. 1501–1507, http://dx.doi.org/10.1109/AIM.2017.8014231.
[40] A. Popovici, P. Zaal, D. Pool, Dual extended Kalman filter for the identification of time-varying human manual control behavior, Model. Simul. Technol. Conf. (2017) 3666, http://dx.doi.org/10.2514/6.2017-3666.
[41] S. Eftekhar Azam, E. Chatzi, C. Papadimitriou, A dual Kalman filter approach for state estimation via output-only acceleration measurements, Mech. Syst. Signal Process. 60–61 (2015) 866–886, http://dx.doi.org/10.1016/j.ymssp.2015.02.001.
[42] S. Stubberud, K. Kramer, Effects of measurement unobservability on neural extended Kalman filter tracking, Proc SPIE 7336 (2009) http://dx.doi.org/10.1117/12.818384.
[43] A.F. Villaverde, Observability and structural identifiability of nonlinear biological systems, Complexity 2019 (1) (2019) 8497093, http://dx.doi.org/10.1155/2019/8497093, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1155/2019/8497093.
[44] S. Boersma, S. van Mourik, B. Xin, G. Kootstra, D. Bustos-Korts, Nonlinear observability analysis and joint state and parameter estimation in a lettuce greenhouse using ensemble Kalman filtering, IFAC- Pap. 55 (32) (2022) 141–146, http://dx.doi.org/10.1016/j.ifacol.2022.11.129, 7th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2022.
[45] N. Wassiliadis, J. Adermann, A. Frericks, M. Pak, C. Reiter, B. Lohmann, M. Lienkamp, Revisiting the dual extended Kalman filter for battery state-of-charge and state-of-health estimation: A use-case life cycle analysis, J. Energy Storage 19 (2018) 73–87, http://dx.doi.org/10.1016/j.est.2018.07.006.
[46] M. Ganaie, M. Hu, A. Malik, M. Tanveer, P. Suganthan, Ensemble deep learning: A review, Eng. Appl. Artif. Intell. 115 (2022) 105151, http://dx.doi.org/10.1016/j.engappai.2022.105151.
[47] Y. Meng, C. Shao, Physics-informed ensemble learning for online joint strength prediction in ultrasonic metal welding, Mech. Syst. Signal Process. 181 (2022) 109473, http://dx.doi.org/10.1016/j.ymssp.2022.109473.
[48] S. Dingler, State estimation with the interacting multiple model (IMM) method, 2022, arXiv:2207.04875.
[49] J. Wang, Y. Alipouri, B. Huang, Dual neural extended Kalman filtering approach for multirate sensor data fusion, IEEE Trans. Instrum. Meas. 70 (2021) 1–9, http://dx.doi.org/10.1109/TIM.2020.3041825.
[50] R. Zhan, J. Wan, Neural network-aided adaptive unscented Kalman filter for nonlinear state estimation, IEEE Signal Process. Lett. 13 (2006) 445–448, http://dx.doi.org/10.1109/LSP.2006.871854.
[51] S. Hosseinyalamdary, Deep Kalman filter: Simultaneous multi-sensor integration and modelling; a GNSS/IMU case study, Sensors 18 (2018) http://dx.doi.org/10.3390/s18051316.
[52] D. Li, J. Zhou, Y. Liu, Recurrent-neural-network-based unscented Kalman filter for estimating and compensating the random drift of MEMS gyroscopes in real time, Mech. Syst. Signal Process. 147 (2021) 107057, http://dx.doi.org/10.1016/j.ymssp.2020.107057.
[53] D. Xinyuan, W. Jinjie, W. Sufeng, C. Ting, An improved CDKF algorithm based on RBF neural network for satellite attitude determination, Int. Conf. Image Anal. Signal Process. (2012) 1–7, http://dx.doi.org/10.1109/IASP.2012.6425013.
[54] J. Ki Lee, C. Jekeli, Neural network aided adaptive filtering and smoothing for an integrated INS/GPS unexploded ordnance geolocation system, J. Navig. 63 (2010) 251–267, http://dx.doi.org/10.1017/S0373463309990397.
[55] S. Kim, I. Petrunin, H.-S. Shin, A review of Kalman filter with artificial intelligence techniques, in: Integrated Communication, Navigation and Surveillance Conference (ICNS), 2022, pp. 1–12, http://dx.doi.org/10.1109/ICNS54818.2022.9771520.
[56] Z. shan Fu, C. ling Yu, Design and kinematics analysis of disc cam with translation flat-face follower, J. Phys.: Conf. Ser. 2044 (2021) 012139, http://dx.doi.org/10.1088/1742-6596/2044/1/012139.
[57] Z. Chen, Bayesian filtering: From Kalman filters to particle filters, and beyond, Statistics 182 (2003) http://dx.doi.org/10.1080/02331880309257.
[58] M.A. Pinsky, S. Karlin, 3 - Markov chains: Introduction, in: An Introduction To Stochastic Modeling (Fourth Edition), fourth ed., Academic Press, Boston, 2011, pp. 79–163, http://dx.doi.org/10.1016/B978-0-12-381416-6.00003-4.
[59] Y. Pei, S. Biswas, D.S. Fussell, K. Pingali, An elementary introduction to Kalman filtering, Commun. ACM 62 (2019) 122–133, http://dx.doi.org/10.1145/3363294.
[60] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444, http://dx.doi.org/10.1038/nature14539.
[61] G. Ortiz-Jiménez, S.-M. Moosavi-Dezfooli, P. Frossard, What can linearized neural networks actually say about generalization? Adv. Neural Inf. Process. Syst. (2021) http://dx.doi.org/10.48550/ARXIV.2106.06770.
[62] A.P. Ramallo-González, M.E. Eames, D.A. Coley, Lumped parameter models for building thermal modelling: An analytic approach to simplifying complex multi-layered constructions, Energy Build. 60 (2013) 174–184, http://dx.doi.org/10.1016/j.enbuild.2013.01.014.

[63] U.M. Ascher, L.R. Petzold, Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, Society for Industrial and Applied Mathematics, Philadelphia, 1998, http://dx.doi.org/10.1137/1.9781611971392.fm.
[64] R. Mortensen, Filtering for stochastic processes with applications to guidance, IEEE Trans. Autom. Control 17 (1972) 184–185, http://dx.doi.org/10.1109/TAC.1972.1099917.
[65] J. Bradbury, R. Frostig, P. Hawkins, M.J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang, JAX: composable transformations of python+numpy programs, 2018.
[66] J. Dunik, O. Straka, O. Kost, J. Havlík, Noise covariance matrices in state-space models: A survey and comparison of estimation methods—Part I, Internat. J. Adapt. Control Signal Process. 31 (2017) 1505–1543, http://dx.doi.org/10.1002/acs.2783.
[67] T. Berry, T. Sauer, Adaptive ensemble Kalman filtering of non-linear systems, Tellus A: Dyn. Meteorol. Ocean. 65 (2013) 20331, http://dx.doi.org/10.3402/tellusa.v65i0.20331.
[68] M. Impraimakis, A.W. Smyth, A new residual-based Kalman filter for real time input–parameter–state estimation using limited output information, Mech. Syst. Signal Process. 178 (2022) 109284, http://dx.doi.org/10.1016/j.ymssp.2022.109284.
[69] H.A. Rothbart (Ed.), CAM Design Handbook, first ed., McGraw-Hill Education, New York, 2004.
[70] M. Vidlak, L. Gorel, P. Makys, M. Stano, Sensorless speed control of brushed DC motor based at new current ripple component signal processing, Energies 14 (2021) http://dx.doi.org/10.3390/en14175359.
[71] D. Misra, Mish: A self regularized non-monotonic activation function, Br. Mach. Vis. Conf. (2019) http://dx.doi.org/10.48550/ARXIV.1908.08681.
[72] M.M. Bejani, M. Ghatee, A systematic review on overfitting control in shallow and deep neural networks, Artif. Intell. Rev. 54 (8) (2021) 6391–6438, http://dx.doi.org/10.1007/s10462-021-09975-1.