Nonlinear Model Learning for Compensation and Feedforward Control of Real-World Hydraulic Actuators Using Gaussian Processes

Abdolreza Taheri, *Graduate Student Member, IEEE*, Pelle Gustafsson, Marcus Rösth, Reza Ghabcheloo, and Joni Pajarinen

***Abstract*—This paper presents a robust machine learning framework for modeling and control of hydraulic actuators. We identify several important challenges concerning learning accurate models of the dynamics for real machines, including noise and uncertainty in state measurements, nonlinear effects, input delays, and data-efficiency. In particular, we propose a dual-Gaussian process (GP) model architecture to learn a surrogate dynamics model of the actuator, and showcase the accuracy of predictions against the piecewise and neural network models that have been widely used in the literature. In addition, we provide robust techniques for learning neural network inverse models and controllers by batch GP inference in an automated, seamless and computationally fast manner. Finally, we demonstrate the performance of the trained controllers in real-world feedforward and tracking control applications.**

***Index Terms*—Hydraulic actuators, machine learning for robot control, model learning for control.**

I. Introduction

All real-world mechanical systems that provide a means to automation today are powered by integrated controllers. These controllers are carefully designed and fine-tuned to provide the best possible performance for the intended application, and their development follows a conventional process. First, the system dynamics are modelled, wherein prior knowledge and general understanding of the physics of the underlying process culminate in a set of mathematical equations. The resulting model is often deterministic, in which a set of parameters within admissible ranges can describe the behavior of individual systems of the same type collectively. The next step is to identify the best set of parameters for each system, since two systems are never completely identical in the real-world sense, e.g. due to subtle differences in the material properties or manufacturing tolerances. Identifying the parameters that match the model to the real system accurately is a complex process, it requires design of experiments, controlled data collection procedures, data filtering and feature extraction, and iterative optimization. Finally, the control system is designed by employing a suitable approach among the rich literature of well-developed methodologies.

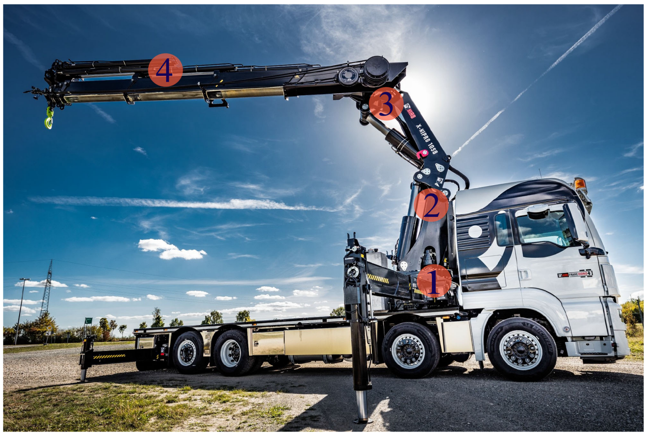

Fig. 1. Hydraulic actuators that drive heavy-duty machines have complex dynamics and are difficult to model from real data. We present an automated framework for learning a predictive model directly from noisy state measurements, and demonstrate techniques to use the model for trajectory tracking control and to eliminate deadband, asymmetric, and off-centered operation by compensating for nonlinearities. Our approach outperforms parametric models, and is tested across limited-resolution angle sensors in revolute joints ①-②-③ to high-resolution measurements in prismatic links ④.

Despite its long history of developments and improvements, this classical approach to model-based control is not devoid of pitfalls and challenges in the case of hydraulic actuators in heavy-duty machines. Most system identification approaches in modelling the dynamics of hydraulic actuators assume either a physical model with few parameters [1], [2] that are unable to capture complex nonlinear effects and may require excessive

Manuscript received 24 February 2022; accepted 22 June 2022. Date of current version 26 July 2022. This letter was recommended for publication by Associate Editor N. Figueroa and Editor J. Kober upon evaluation of the reviewers' comments. This work was supported by the European Union's Horizon 2020 Research and Innovation Programme Marie Sklodowska Curie under Grant 858101. *(Corresponding author: Abdolreza Taheri.)*

Abdolreza Taheri is with Control Systems R&D, HIAB, 824 50 Hudiksvall, Sweden, and also with the Faculty of Engineering and Natural Sciences, Tampere University, 33100 Tampere, Finland (e-mail: reza.taheri@tuni.fi).

Pelle Gustafsson and Marcus Rösth are with Control Systems R&D, HIAB, 824 50 Hudiksvall, Sweden (e-mail: pelle.gustafsson@hiab.com; marcus.rosth@hiab.com).

Reza Ghabcheloo is with the Faculty of Engineering and Natural Sciences, Tampere University, 33100 Tampere, Finland (e-mail: reza.ghabcheloo@tuni.fi).

Joni Pajarinen is with the Department of Electrical Engineering and Automation, Aalto University, 02150 Espoo, Finland, and also with Intelligent Autonomous Systems, TU Darmstadt, 64289 Darmstadt, Germany (e-mail: joni.pajarinen@aalto.fi).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3190808, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3190808

controller tuning, or neural networks (NNs) [3]–[7] that contain many parameters, yet require lots of data for training. On the other hand, the least-squares objective of optimization does not explicitly account for measurement uncertainties, e.g. sensor noise, varying input delays, vibrations, transient dynamics, and other nonlinear effects, which are always present in the real system. Consequently, for physical models [8], [9], measurement data have to be smoothed prior to optimizing the models, and experiments need to be conducted under considerations that facilitate filtering and identification, e.g., steadily increasing/decreasing inputs [1], [9]–[11]. Closed-loop control can remedy model errors to some extent, but it requires accurate sensory feedback which may not be accessible in all actuator types.

One more important challenge in model learning and control of real machines is the amount of data required for accurate system identification. Machines and their actuators come in different sizes and characteristics, so it is difficult, if not virtually impossible, to spend hours or days experimenting with each individual machine to collect sufficient data for learning a dynamics model, and to do so within the safety limits. This is contrary to studies in simulation [12], [13], where generating up to millions of interactions is cheap with regards to computation time, energy, and safety. Fortunately, recent studies have provided frameworks for learning controllers from surrogate Gaussian process (GP) models [14] trained on limited amount of data [8]. In the framework of model-based policy optimization, a surrogate model of the dynamics can be effectively used for learning policies, which are directly applicable to the real system [15]–[17]. Surrogate models can speed up control design, owing to the recent advances in algorithms and efficient hardware implementations [15], [18], [19], and allow for verifying and testing the controllers for correctness of the solution, smoothness and consistency of the outputs before deploying the logic on real machine hardware [15].

Contributions: We describe a framework to train surrogate models of hydraulic actuators from noisy measurements of actuator state (e.g. position, angle). Specifically, we train a nonlinear probabilistic model of the actuators using the latest scalable GP inference methods via maximizing the log marginal likelihood objective and showcase prediction accuracy against parametric models (NN and piecewise functions) trained using the mean-squared-error (MSE) objective. We then demonstrate the use of these models in training non-linear inverse and feedforward controllers by querying the GP models. Additionally, our work provides:

- A sound approach to modelling actuator dynamics and control using non-uniformly sampled, unfiltered, noisy measurements without constraining the underlying physical model to a parametric form.
- Techniques to enhance the data-efficiency and accuracy of predictions and automatically handling input-delays, by decomposing the system into two components and using a dual-GP model architecture for prediction.
- A robust workflow for training NN inverse models by leveraging a novel monotonicity loss that ensures physical-plausibility by penalizing negative gradients in the model.
- Fast controller optimization on GPU hardware that takes place in minutes, by leveraging the latest scalable GP inference and automatic differentiation [15], [18]–[20].
- Considering hydraulics control to be a safety critical application, our approach ensures that the open-loop controllers and inverse models can be constrained to smoothness and monotonic behavior, and inspected for these properties before running on the real machine.
- Results of real experiments on a loader crane for various control applications as well as benchmarking the performance of the models and controllers across different actuators with varying sensor resolutions.

The rest of this paper is organized as follows. A review of related works and challenges are presented in Section II. Section III offers a detailed description of our proposed method for dynamics model learning from data, filtering methods and data collection for validation, followed by techniques to train inverse models and controllers from the models. Section IV summarizes the results of our experiments with a real machine, and Section V concludes the paper.

II. Related Work

The trend towards utilizing machine learning (ML) for complex control applications in the field of heavy-duty machines is becoming more and more visible in recent years [3], [4], [7], [12], [13], [21]. Some studies are carried out either entirely in simulation [12], [13], which can involve learning very complex tasks from up to millions of steps in simulation, or demonstrate control experiments on the real machine, where some form of model-based optimization [11], [15] or learning from demonstration [5], [6] is studied. ML is a less well-explored field of research compared to classical control, and safety considerations are a key limiting factor in testing new methodologies on heavy machines.

Recent works highlight potential applications of NNs in control of heavy-duty machines. In [3], a learned model of a hydraulic excavator and its inverse have been utilized for precision tip control. The model is structured into three components, where different customized units account for input delays and deadband, and the remaining nonlinear effects are captured by a large feedforward NN. Tip control is then demonstrated by model inversion and proportional control. A model-based reinforcement learning approach to end-effector position tracking control was carried out in [11], using a larger NN to model the dynamics of four actuators on a hydraulic excavator and data collected with varied-frequency sinusoidal and ramp inputs. A recurrent NN structure was used in [22] for online tracking and compensation control applied to hydraulic excavators. Nevertheless, closed-loop control demonstrations obfuscate the inaccuracies of the surrogate models, and these approaches require velocity data for training and closed-loop feedback, which may not be available for all hydraulic joints in a heavy-duty machine. Implementing filters to explicitly smooth the outputs [22] might be one solution, yet as we demonstrate in Section IV-A, low-pass filtering is prone to signal inaccuracies, vibrations, and noise, and the introduced lag from filtering can negatively affect model learning. On the other hand, derivative-free modeling and control is an interesting area to pursue for robotics applications, as shown in [23] through the use of GPs and customized kernels.

Since NN models of the dynamics require lots of data to achieve a good level of performance, some approaches train a policy using smaller NNs, using less but qualitative, task-specific expert demonstrations [5]–[7]. Nevertheless, the capacity of the small models are limited for capturing all nonlinearities in a machine. In [4], a time-delayed NN is proposed for learning demonstrations of bucket filling operations for a wheel loader. The effect of time delay in inputs were concluded to be a significant factor and the most relevant inputs were identified after training by analyzing the NN parameters. However, calculating sensitivity to inputs in larger fully connected networks may not be easy. In our approach, we use the autoregressive GP structure from [24], [25] with automatic relevance determination [14] in the kernel length scales (denoted $\mathbf{\Lambda}$ in Section III-B) to handle time delays for each actuator, which are utilized by a high-level planner in our experiments in Section IV-D.

The mathematically-derived equations of flow that describe the dynamics of actuators [8], [9], or the piecewise functions that are characterized by physically meaningful parameters (such as the maximum input currents within deadband) [1], [2], provide us with models that contain very few parameters which can be tuned for a real actuator by nonlinear optimization [1]. One practical aspect of the piecewise models is that since the function is non-decreasing by design, the inverse is very straightforward to obtain [2] for control applications. However, the capacity of these models are further limited, for instance, a fixed parameter for actuator deadzone [2], [22] may not be the ideal solution for accurate control. More importantly, the training objective for these models uses the same least-squares objectives as NNs, which does not explicitly account for the uncertainties and noise in measurements.

Our approach is inspired by the recent works in model learning and policy optimization using GPs [8], [14], [15], [17], [24]. GP models have been shown to effectively learn from limited amount of data [16], [17], have very few hyperparameters to optimize, provide an estimate of uncertainty in the predictions, and take measurement noise explicitly into account. A study of autoregressive GPs on a simulation model of hydraulic actuators has been provided in [8], [24]. However, the applicability of the models for control was not discussed, it had also remained an open question whether the method could cope with the effects of nonuniform and low-resolution measurements from the real system. The recent algorithmic advancements and efficient hardware-accelerated implementations have significantly sped up computations for GP inference [18], [19]. A recent study on GPU-accelerated policy optimization using GPs [15] highlighted fast controller learning for position tracking control of a wheel-loader actuator, yet the model was trained only on two discrete speeds for demonstration. In this work, we put forward a more detailed analysis on constructing the full dynamic characteristics of hydraulic actuators, including deadband, asymmetry, saturation, and the ability of GPs in handling the uncertainty in measurements using real machine data.

III. Automated Dynamics Modelling From Data

This section develops a framework to train and validate predictive state-transition models from noisy observations of an actuator's state. We consider directly driven proportional spool valves [9], where an input current u is applied to a solenoid driving an inner sliding spool, and the resulting spool displacement delivers a certain rate of flow that puts the actuator in motion. The spool overlap presents a deadband region [9], where the actuator stays motionless despite current being applied. Real-world applications of the proposed framework, such as feedforward control for eliminating deadband and other nonlinearities in the actuator are presented in Section IV.

A. Data Collection & Filtering

In a purely classical control approach to constructing a model of actuator response when only state measurements are accessible, the data is collected by executing passive input currents to the actuator. The noisy measurements of position are then passed through a filter to be smoothed and an estimate of the velocity is obtained in the process. Filters that use present and past data are causal, which are suitable for online estimation. Filters that take future measurements into computations are termed acausal (noncausal), which can only be utilized offline but provide much better estimation [23]. In particular, we compare two filters with the predictive model. The chosen causal filter is an adaptive low-pass filter [26]. Its formulation takes into account the sampling time intervals and is computationally inexpensive, making it suitable for use in real-time applications. For the acausal filter, we use the Savitzky-Golay (SG) filter extended to non-uniformly sampled data [27]. The SG-filter works by fitting an n$^{\text{th}}$ order polynomial to a window of samples around a point via least squares. The non-uniform SG-filter is computationally more expensive than the standard uniformly-sampled SG [28] but provides a better estimate for our validation analysis with real data, especially with lower resolution measurements. For the complete derivation of the filters, refer to [26]–[28].

Filtering data for training predictive models has disadvantages, e.g. the induced lag [26], which increases as the cut-off frequency is lowered to filter out high frequency noise. In our application, this lag degrades the prediction accuracy of models and policy performance. Moreover, the actuation and data collection have to follow steady trends (e.g. ramp or step inputs) for these filters to achieve consistent estimations. We aim to avoid these drawbacks by training a predictive model directly on noisy, unfiltered measurements which can be validated with filtered ramp experiments. The results are presented and discussed in Section IV-A.

B. Dual-GP Dynamics Model for Hydraulic Actuators

This section describes our approach to modelling the hydraulic actuator dynamics by leveraging Gaussian Processes (GPs) [14], [24]. The dynamic characteristics of hydraulic actuators depend on many uncertain factors, and are intrinsically stochastic and nonlinear. Therefore, instead of constraining the dynamics with a parametric form [1], we treat the measurements directly as noisy observations $\hat{f}(\boldsymbol{u}) = f(\boldsymbol{u}) + \varepsilon$ of a nonlinear function that expresses transitions $\dot{\boldsymbol{x}} = f(\boldsymbol{u})$ of actuator state $\boldsymbol{x}$ on input variables $\boldsymbol{u}$, then train a GP to model the function as a collection of random variables $\hat{f}(\boldsymbol{u}) \sim \mathcal{GP}(\mu, k(\boldsymbol{u}_i, \boldsymbol{u}_j))$. The distribution of $\hat{f}(\boldsymbol{u})$ at different inputs are then jointly Gaussian and defined by a mean $\mu : \mathbb{R}^d \to \mathbb{R}$ and covariance function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. A widely used covariance function for robotics applications [17] is the exponentiated quadratic kernel:

$$k(\boldsymbol{u}_i, \boldsymbol{u}_j) = \sigma_k^2 \exp\left[-\frac{1}{2}(\boldsymbol{u}_i - \boldsymbol{u}_j)^T \boldsymbol{\Lambda}(\boldsymbol{u}_i - \boldsymbol{u}_j)\right] \quad (1)$$

where σ_k adjusts the variance and $\boldsymbol{\Lambda} = \{l_1, \ldots, l_d\}$ is a diagonal matrix containing length-scales for each dimension d of inputs. To account for input-to-actuation delays, the models are trained over a sequence of past training control inputs $\boldsymbol{u_k} = \{u_{k-d+1}, \ldots, u_{k-1}, u_k\}$, an idea that has been used in autoregressive GP models [8], [24], [25]. After optimization of parameters, the values of length-scale for each delayed input can determine the relevance of that input in predictions, automatically emphasizing the most appropriate input for predictions. We further propose to decouple the dynamics into a cascaded process of two GPs: $\mathcal{GP}_I$ - a model of the input currents ($\boldsymbol{u}$) to spool displacement (x_s), associated with the electromagnetic (solenoid) and the inner spool dynamics, and $\mathcal{GP}_{II}$ - a model to characterize the hydraulic oil flow and actuator mechanics, which predicts the rate of change in actuator state (which is proportional to flow rate) from spool position. Doing so enhances the accuracy of predictions, reduces the dimensionality of inference and complexity of training. Moreover, each component model can be closely examined after training for insights on the real system, allowing us to identify certain characteristics such as delays, saturation values, center points and asymmetry (refer to [9] for a detailed description of spool characteristics). To account for non-uniformly sampled data (e.g., measurements in real experiments), we define the observations as the discrete rate of change of states $y_k = \frac{\Delta x_k}{\Delta t_k} = \frac{x_{k+1} - x_k}{t_{k+1} - t_k}$. An overview of sampling from the proposed dual-GP model architecture for predictions is depicted in blue in Fig. 2.

Prediction at test inputs $\boldsymbol{u}^*$ can be performed through the posterior distribution $p(\hat{f}(\boldsymbol{u}^*)|U, \boldsymbol{y})$, which is Gaussian with mean and variance defined as:

$$\mathbb{E}\left[\hat{f}(\boldsymbol{u}^*)|U, \boldsymbol{y}\right] = \mu(\boldsymbol{u}^*) + \boldsymbol{k}_{U\boldsymbol{u}^*}^T \hat{K}_{UU}^{-1} \boldsymbol{y} \quad (2)$$

$$\text{var}\left[\hat{f}(\boldsymbol{u}^*)|U, \boldsymbol{y}\right] = k(\boldsymbol{u}^*, \boldsymbol{u}^*) - \boldsymbol{k}_{U\boldsymbol{u}^*}^T \hat{K}_{UU}^{-1} \boldsymbol{k}_{U\boldsymbol{u}^*} \quad (3)$$

where U denotes the dataset of training inputs, K_{UU} is the covariance matrix with elements $k(\boldsymbol{u}_i, \boldsymbol{u}_j)$, $\hat{K}_{UU} = (K_{UU} + \sigma_\varepsilon^2 I)$, σ_ε^2 relates to the noise in observations, and $\boldsymbol{k}_{U\boldsymbol{u}^*}$ is a vector of the kernel evaluated over all possible input pairs and test points. Altogether, the dual-GP model contains the functional parameters $\{\boldsymbol{\phi}_I, \boldsymbol{\phi}_{II}\}$, where $\boldsymbol{\phi} = [\boldsymbol{\Lambda}, \sigma_k, \sigma_\varepsilon]$ and the subscripts I and II are dropped for clarity. These parameters are optimized through log marginal likelihood maximization using gradient descent [19], [29]

$$\mathcal{L}_{\mathcal{G}} = \log p(\boldsymbol{y}|U, \boldsymbol{\phi}) \propto -\boldsymbol{y}^T \hat{K}_{\boldsymbol{\phi},UU}^{-1} \boldsymbol{y} - \log|\hat{K}_{\boldsymbol{\phi},UU}| \quad (4)$$

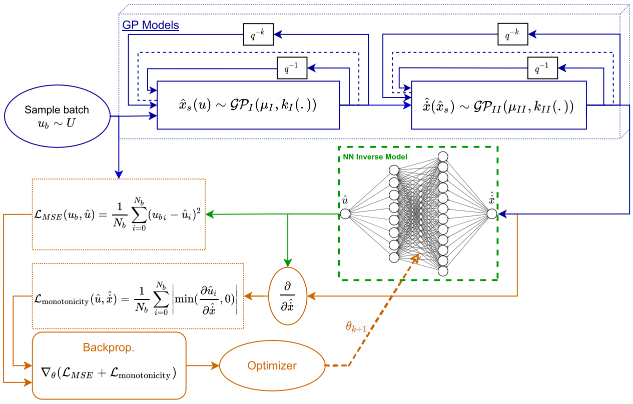
GP Models
q^{-k}
q^{-1}
q^{-k}
q^{-1}
Sample batch
$u_b \sim U$
$\hat{x}_s(u) \sim \mathcal{GP}_I(\mu_I, k_I(.))$
$\hat{\dot{x}}(\hat{x}_s) \sim \mathcal{GP}_{II}(\mu_{II}, k_{II}(.))$
NN Inverse Model
$\hat{u}$
$\hat{\dot{x}}$
$\mathcal{L}_{MSE}(u_b, \hat{u}) = \frac{1}{N_b}\sum_{i=0}^{N_b}(u_{bi} - \hat{u}_i)^2$
$\mathcal{L}_{\text{monotonicity}}(\hat{u}, \hat{\dot{x}}) = \frac{1}{N_b}\sum_{i=0}^{N_b}\left|\min(\frac{\partial \hat{u}_i}{\partial \hat{\dot{x}}}, 0)\right|$
$\frac{\partial}{\partial \hat{\dot{x}}}$
θ_{k+1}
Backprop.
$\nabla_\theta(\mathcal{L}_{MSE} + \mathcal{L}_{\text{monotonicity}})$
Optimizer

Fig. 2. Overview of the dataflow in training an inverse model of the actuator dynamics. The approach comprises three parts: Simulating the surrogate dynamics by batch querying the GP models (blue), prediction of corresponding input commands by the NN (green), backpropagation and NN weights update forcing monotonicity (orange).

$$\frac{\partial \mathcal{L}_{\mathcal{G}}}{\partial \boldsymbol{\phi}} \propto \boldsymbol{y}^T \hat{K}_{\boldsymbol{\phi},UU} \frac{\partial \hat{K}_{\boldsymbol{\phi},UU}^{-1}}{\partial \boldsymbol{\phi}} \hat{K}_{\boldsymbol{\phi},UU} \boldsymbol{y} - \text{tr}\left\{\hat{K}_{\boldsymbol{\phi},UU} \frac{\partial \hat{K}_{\boldsymbol{\phi},UU}}{\partial \boldsymbol{\phi}}\right\} \quad (5)$$

There are a handful of techniques available for efficiently computing the predictive mean and covariances that have drastically reduced the time and memory complexity of GPs, mainly by computing decent approximations to the large matrix inverse terms. These include batched conjugate gradients, efficient matrix-vector multiplication routines that utilize GPU hardware, and the Lanczos algorithm [18]. Refer to [19] for details of the algorithms and implementation in GPyTorch, and [15] for a detailed study of the speedups in training and inference for policy optimization on GPU hardware.

C. NN Inverse Model and Control

An inverse model ($\hat{u} = f^{-1}(\dot{x}) : \mathbb{R} \to \mathbb{R}$) for the spool valve actuator defines a function determining the estimated amount of input current to the actuator $\hat{u}$ that when applied to the system, achieves a desired flow rate $\dot{x}$ at steady-state. An inverse model is very useful in that it can be directly applied in many control problems to infer the commands for trajectory tracking or compensation of undesired behavior in the system [3], as we will later demonstrate with our results in Section IV-B and IV-D. Given that a GP defines a distribution over random variables, there is no straightforward way to obtain the inverse solution, neither does a unique inverse exist (this can be inferred from the characteristic plots shown in Fig. 5 and the naively inverted mean of the GP in Fig. 6 which is not a one-to-one function). On the other hand, for application in control it is crucial for the inverse model to be physically plausible, for instance, there should be no abnormal discontinuities in the model. An important physical property that we aim to have in our case in the inverse model is that a higher input current results in an equal or higher flow rate, enforcing a non-decreasing constraint on the inverse function $\forall \dot{x} \in \mathbb{R} : \frac{d}{d\dot{x}} f^{-1}(\dot{x}) \geq 0$.

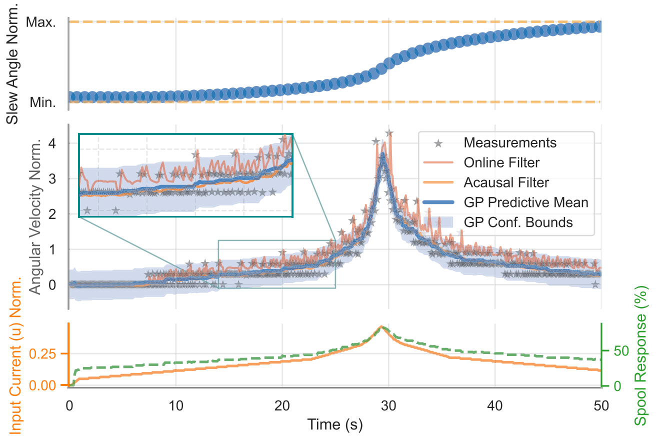

Fig. 3. A standard experiment using ramp inputs on a crane slew actuator ①. Top: actuator state (angle). Middle: estimated, predicted, and measured actuator velocity. Bottom: actuator input and spool displacement. The predictive model is trained on raw data, yet the predictions follow closely the smoothed (acausal) estimates.

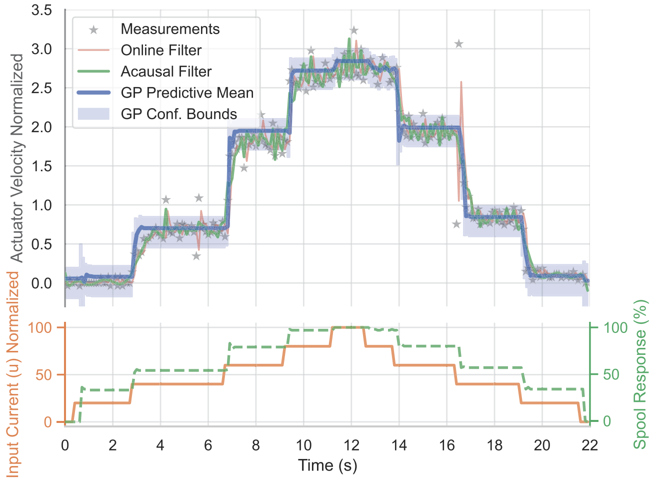

Fig. 4. Real experiment with frequently changing step inputs. The inputs before $t = 3\ s$ and after $t = 19\ s$ are within the actuation deadband, and saturation of the output can be seen at the peak. The estimates from filtering are prone to data fluctuations, and do not provide any advantage for learning the model, while GP model performs well using only raw measurements.

One possible method to finding a suitable inverse is to subject the GP modelling to inequality constraints such as monotonicity [30]. This can be achieved, e.g. by imposing a constraint on function derivatives [31], but is an overly complicated process that requires alternations to the maximum log marginal likelihood objective (4), variational inference, or algorithms to sample under constraints. These approaches are not well-explored for control, and enforcing monotonicity on the model itself still does not provide any guarantees on the plausibility of the inverse. Therefore, we take a more solid constrained optimization approach to obtaining the inverse by leveraging fast GP inference and automatic differentiation with computational graphs [20]. An overview of the optimization is depicted in Fig. 2. Let an NN define the inverse model, since our GP inference method (Section III-B) is fast and scalable [15], we can continuously sample batches of repeated (normalized) control values $u_b \in \mathbb{R}^{N_b \times d}, \{u_{b_{i,k-d+1}} = \ldots = u_{b_{i,k}} \in [-1, 1]\}$, and query the GPs in succession to obtain the corresponding estimates of $\hat{\dot{x}}$, the output of the dual-GP model. Then, proceed to evaluate the commonly used MSE loss (6) for the NN using the training set $\{\hat{\dot{x}}, u_b\}$:

$$\mathcal{L}_{MSE}(u_b, \hat{u}) = \frac{1}{N_b} \sum_{i=0}^{N_b} (u_{bi} - \hat{u}_i)^2 \tag{6}$$

In addition, to ensure that the inverse is a physically plausible, non-decreasing function we propose an additional loss term, which we will refer to as the monotonicity loss:

$$\mathcal{L}_{\text{monotonicity}}(\hat{u}, \hat{\dot{x}}) = \frac{1}{N_b} \sum_{i=0}^{N_b} \left| \min\left(\frac{\partial \hat{u}_i}{\partial \hat{\dot{x}}}, 0 \right) \right| \tag{7}$$

this loss retrieves the gradient of the model with respect to the inputs $\frac{\partial \hat{u}}{\partial \hat{\dot{x}}}$, which are accessible in the computational graph of Fig. 2 by automatic differentiation, then penalizes the mean absolute value of negative gradients. The effect of this loss in training the inverse model is later shown in Section IV-B. Finally, the NN inverse model is trained by backpropagation and weight updates to minimize the sum of both losses using gradient descent [29]. It is worth noting that the inverse model can also be trained via the traditional velocity-error minimizing objective: By sampling velocity values, feeding them through the inverse model and through the GP models to estimate the achieved velocities. However, such approach requires gradients to propagate through the GP models, which consumes more time and memory [15]. Our training workflow in Fig. 2 is designed to avoid this by isolating NN backpropagation away from the GP models.

IV. Application to Real-World Control

In this section, we demonstrate how well the surrogate GP models for each individual hydraulic actuator are constructed using real measurements (Section IV-A), and how they can be utilized for training inverse models (Section IV-B) and control in heavy-duty machines from compensating for nonlinear actuator dynamics to achieve smooth and symmetric behavior (Section IV-C), to open-loop trajectory tracking control with the actuators in multi-joint motions (Section IV-D).

A. GP Predictive Models: Training and Validation

We begin by collecting a training dataset, as detailed in Section III-A. The control input is the remote controller signal to the actuators of the machine shown in Fig. 1 over controller area network (CAN) connections operating above 50 Hz. The data is downsampled to 10 Hz and z-score normalized [14] to facilitate GP training. The input to the models comprise up to 5 time-delayed samples. The total time of data collected for each actuator is approximately 20 minutes, with equally split training and validation data. Validation data is obtained through steadily-increasing and decreasing inputs, and the outputs are smoothed using a SG-filter (acausal) that operates over a window of 51 samples and fits a polynomial of order 11. A sample of the ramp-input experiment for validation is exemplified in Fig. 3.

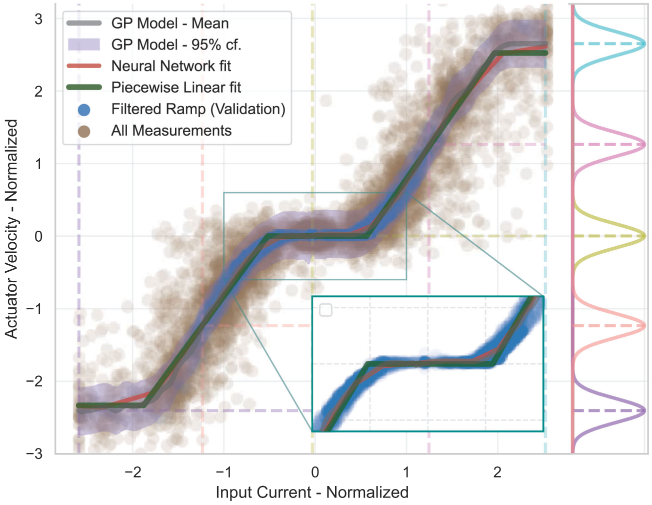

Fig. 5. Characteristic curves of three optimized surrogate models (GP, NN, Piecewise Linear) for a real hydraulic actuator ④. Gaussian distributions of outputs corresponding to selected inputs are illustrated on the right. The curve is non-linear, asymmetric, off-centered, and contains deadband and saturation. Ramp experiments validate the accuracy of trained models.

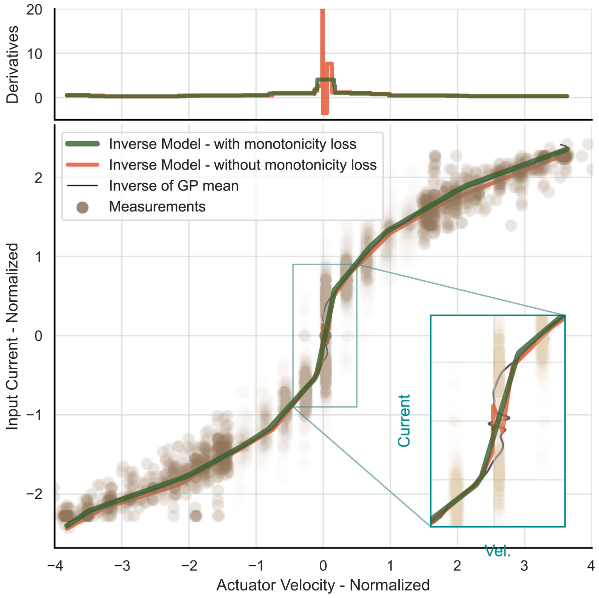

Fig. 6. Characteristic curve of the NN inverse model trained on the dual-GP model for the slew actuator ① and the effect of the monotonicity loss on trained models. The derivatives of functions are illustrated at the top.

TABLE I
Performance of Trained Models - Training and Validation

Model	$\mathcal{L}_{MSE}$ (Training)	$\mathcal{L}_{MSE}$ (Validation)	# of Parameters
Gaussian Process	$\mathbf{2.592}\times10^{-2}$	$\mathbf{1.081}\times10^{-3}$	non-parametric
Neural Network	7.061×10^{-2}	7.726×10^{-3}	385
Piecewise Linear	7.265×10^{-2}	10.237×10^{-3}	6

This is a case where state measurements are only accurate to 0.1 degrees, hence causal filtering techniques that work in real time (online), such as the low-pass filter in Fig. 3 cannot yield a correct estimate of the velocity. The carefully collected and filtered ramp data (as used in hydraulic systems identification) provide an estimate of the actuator velocity ground truth, and are better suited for validation of the models than using part of the discrete measurements as validation data (as in most ML workflows). However, a drawback is that the ramp experiments can be carried out for only a limited range of inputs (in Fig. 3, a little over 40%) before the actuator state reaches its physical bounds. As a result, we are only able to validate the models in the regions within this limit. The training dataset, on the other hand, includes all arbitrary transitions and is not limited to ramp inputs. One instance is shown in Fig. 4, where the spool valve is commanded with step inputs up to 100%. The actuator in this case is for the prismatic joint in Fig. 1, accurate to 1 mm in measuring the displacement of the piston rod. However, the noise in raw measurements varies, and is influenced by vibrations in the links at higher speeds. Objectively, it is possible to design the filters to better smooth the data, yet the process would require cumbersome filter design and tuning based on data, and will introduce additional lag in the response, which for the purpose of model training and control is undesirable.

Projecting the end-to-end dataset of training, along with the predictive distribution of the GPs, we get the characteristic curve of the spool valve shown in Fig. 5. The curve is well constructed, though the measurements are sporadic due to sensor noise, vibrations, transient dynamics of pressure, and other uncertain factors. We also provide a comparison between our model and two widely used deterministic models, one having a piecewise linear structure as in [1], and the other an NN with two hidden layers of 16 nodes each with ReLU activation. All optimizations for the GP and parametric models are carried out in PyTorch [20] using Adam optimizer [29], with the same dataset and learning rate ($\eta = 10^{-2}$) that has worked best across all models. To address randomized initializations and training, the model with the best final loss out of 10 runs is reported for the analysis. The overall training for each actuator model is complete within only a few minutes using GPU hardware [15]. The predictions of the NN and piecewise function are shown in Fig. 5, and detailed values are compared in Table I. The models are validated via the MSE loss between the predictions and filtered data from ramp experiments which are depicted in blue in Fig. 5. As can be inferred from Table I and Fig. 5, the piecewise linear approach [1] is oversimplistic and presents the highest MSE loss, overestimates the deadband, and underestimates the saturation on the right half plane of the characteristic curve. The NN model has more capacity to capture the characteristics of deadband, where more data is available, but underestimates the values at saturation, and is outperformed by GPs by a factor of over ×7 on the validation set.

Overall, it can be concluded that there are a handful of properties that render GPs more suitable towards modeling the spool valves over the parametric models. Firstly, the noise and uncertainty in measurements are explicitly taken into account via the log marginal likelihood objective (4), while both parametric models try to minimize the naive MSE loss that not only leans over the regions where more data is available, but also treat the outputs as deterministic points with no regards to noise

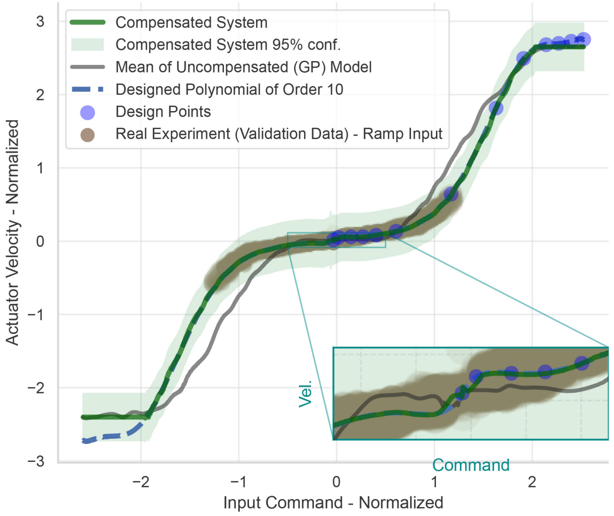

Fig. 7. The characteristic curve of a compensated system. An NN controller is trained on the surrogate model to make the dynamics follow the designed curve within dynamical constraints. Performance of the new system is validated by applying ramp commands to the compensator for actuator ④.

or uncertainty. They are also prone to getting stuck in locally optimal solutions, as the comparison of training losses in Table I delineates. The GPs are also better at using the neighboring measurements for more accurate predictions on a point, owing to the covariance function (1), an extremely helpful property for constructing the characteristic curve over all admissible controls from limited data as sufficient measurements are not available at every input. Lastly, the parametric models shown here perform relatively well in the case of a sensor with 1 mm accuracy, but poorly on actuators with low-resolution measurements.

B. Computing Inverse Models

This section follows the method outlined in Section III-C and Fig. 2 for training an inverse model. The results are shown in Fig. 6 for an actuator with discrete measurements limited to 0.1 degrees. The effect of the proposed monotonicity loss (7) on the inverse is demonstrated by comparing it against training using only the MSE loss (6), which exhibits undesirable high gradient jumps in the deadband. As explained in Section III-C, our approach preserves the flexibility of the GP models, the inverse solution is ensured to remain within the defined plausible set, the training for each actuator model takes less than a minute [15], and can be adopted for a variety of different tasks.

C. Compensating for Nonlinear Dynamics

In addition to training inverse models, the surrogate models can be utilized to train a controller for desired purposes directly. In this section we will train a compensator to reshape the dynamic characteristics of the actuator shown in Fig. 5 into a desired smooth, no deadband, centered and symmetric behavior shown in Fig. 7. We propose a desired curve outlined in Fig. 7, by fitting a polynomial to a set of waypoints. For symmetry, the points are chosen on the right-half plane only and the curve is mirrored. Using the same structure for NN and the loss objectives as before, we train a controller that, when commanded with an input u_{cmd}, maps the command to an input current that will reshape the output of the dual-GP model (and consequently the real system) to match the desired output determined by the polynomial curve. A comparison between the uncompensated (GP) models and the new, compensated system of NN+GPs and the corresponding confidence bounds are depicted in Fig. 7, and the results of the real experiment verify the integrity of the controller. This compensation has significant impacts and use cases in assistant functions and operator controlled machines, as it reduces the burden of adapting to the (unknown) deadband, and makes the function symmetric. The compensated system is also less sensitive to the change in inputs around the center, allowing a broader tolerance where precision control is desired.

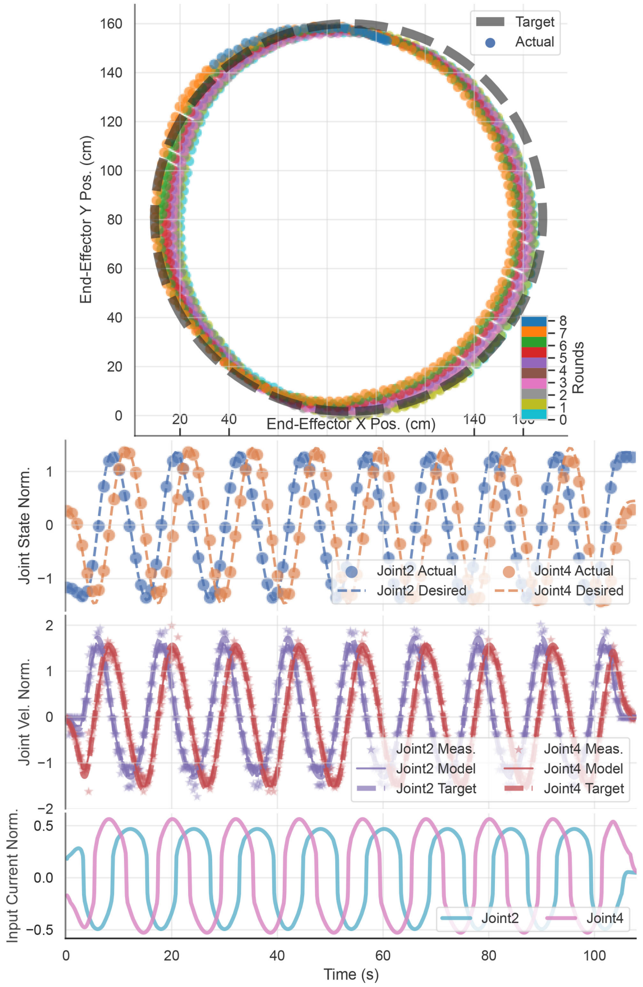

Fig. 8. Open-loop velocity tracking on the real machine following a circular trajectory using actuators ②④. The inverse model is able to handle the deadband and nonlinearities very delicately to follow the desired profile, so that the real velocity measurements match the model predictions and drift is small. The root-mean-square-error of end effector position is 6.85cm.

D. Multi-Joint Open-Loop Trajectory Tracking

This section presents our final experiment on the effectiveness of the constructed GP models in control. Given a pre-generated time-sequence of desired velocities $\tau = \{\dot{x}_{t_0}, \dot{x}_{t_1}, \ldots, \dot{x}_{t_f}\}$

(e.g., from a high-level planner) for two actuators, we passively execute the corresponding input currents computed by the inverse model $\{\hat{u}_{t_0}, \hat{u}_{t_1}, \ldots, \hat{u}_{t_f}\}$ (Section IV-B) and evaluate the tracking performance of the system. The results are summarized in Fig. 8.

V. Conclusion

System identification and control of hydraulic actuators faces several challenges when it comes to real machines: Complex dynamics, noise and uncertainty in measurements, and having a handful of data to train with, among other complications that need to be accounted for. To this end, we proposed a robust framework of machine learning methods to learn accurate models of actuator dynamics while explicitly accounting for the uncertainties, and outlined the use of these models in training physically-plausible inverse models and feedforward controllers for real-world applications. As demonstrated through detailed experimental results, the proposed approach is applicable to a variety of hydraulic actuators and sensor resolutions. Moreover, the dual-GP architecture outperforms the neural networks and piecewise models, and enables rapid simulations for training well-performing neural network controllers in a very short time.

V. Acknowledgments

The authors gratefully acknowledge S. Siren, B. Ur Rehman, S. Fodor, and H. Lyngbäck (HIAB), J. Vihonen and M. Aref (Cargotec), for all the insightful discussions regarding hydraulics, actuation, data collection, filtering, and dynamics modelling. They would also like to thank CSC IT Center for Science in Finland for computational resources.

References

[1] S. Aranovskiy, “Modeling and identification of spool dynamics in an industrial electro-hydraulic valve,” in *Proc. 21st Mediterranean Conf. Control Automat.*, 2013, pp. 82–87.

[2] J. Nurmi and J. Mattila, “Automated feed-forward learning for pressure-compensated mobile hydraulic valves with significant dead-zone,” *Fluid Power Syst. Technol.*, vol. 58332, 2017, Art. no. V001T01A027.

[3] M. Lee, H. Choi, C. Kim, J. H. Moon, D. Kim, and D. Lee, “Precision motion control of robotized industrial hydraulic excavators via data-driven model inversion,” *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 1912–1919, Apr. 2022.

[4] S. Dadhich, F. Sandin, U. Bodin, U. Andersson, and T. Martinsson, “Field test of neural-network based automatic bucket-filling algorithm for wheel-loaders,” *Automat. Construction*, vol. 97, pp. 1–12, 2019.

[5] W. Yang *et al.*, “Neural network controller for autonomous pile loading revised,” in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 2198–2204.

[6] E. Halbach, J. Kämäräinen, and R. Ghabcheloo, “Neural network pile loading controller trained by demonstration,” in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 980–986.

[7] S. Dadhich, F. Sandin, and U. Bodin, “Predicting bucket-filling control actions of a wheel-loader operator using a neural network ensemble,” in *Proc. Int. Joint Conf. Neural Netw.*, 2018, pp. 1–6.

[8] G. Gregorcic and G. Lightbody, “Gaussian processes for modelling of dynamic non-linear systems,” in *Proc. Ir. Signals Syst. Conf.*, 2002, pp. 141–147.

[9] P. Tamburrano, A. R. Plummer, E. Distaso, and R. Amirante, “A review of direct drive proportional electrohydraulic spool valves: Industrial state-of-the-art and research advancements,” *J. Dynamic Syst. Measur., Control*, vol. 141, no. 2, 2019, Art. no. 020801.

[10] J.-H. Lee, Y.-W. Yun, H.-W. Hong, and M.-K. Park, “Control of spool position of on/off solenoid operated hydraulic valve by sliding-mode controller,” *J. Mech. Sci. Technol.*, vol. 29, no. 12, pp. 5395–5408, 2015.

[11] P. Egli and M. Hutter, “Towards RL-based hydraulic excavator automation,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 2692–2697.

[12] J. Andersson, K. Bodin, D. Lindmark, M. Servin, and E. Wallin, “Reinforcement learning control of a forestry crane manipulator,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 2121–2126.

[13] S. Backman, D. Lindmark, K. Bodin, M. Servin, J. Mörk, and H. Löfgren, “Continuous control of an underground loader using deep reinforcement learning,” *Machines*, vol. 9, no. 10, 2021, Art. no. 216.

[14] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, vol. 2. Cambridge, MA, USA: MIT Press, 2006.

[15] A. Taheri, J. Pajarinen, and R. Ghabcheloo, “GPU-accelerated policy optimization via batch automatic differentiation of Gaussian processes for real-world control,” in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 10557–10563, doi: 10.1109/ICRA46639.2022.9811876.

[16] K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, and J.-B. Mouret, “A survey on policy search algorithms for learning robot controllers in a handful of trials,” *IEEE Trans. Robot.*, vol. 36, no. 2, pp. 328–347, Apr. 2020.

[17] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, “Gaussian processes for data-efficient learning in robotics and control,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 2, pp. 408–423, Feb. 2015.

[18] G. Pleiss, J. Gardner, K. Weinberger, and A. G. Wilson, “Constant-time predictive distributions for Gaussian processes,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4114–4123.

[19] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, “GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7587–7597.

[20] A. Paszke *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.

[21] P. Egli and M. Hutter, “A general approach for the automation of hydraulic excavator arms using reinforcement learning,” *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 5679–5686, Apr. 2022.

[22] J. Park, B. Lee, S. Kang, P. Y. Kim, and H. J. Kim, “Online learning control of hydraulic excavators based on echo-state networks,” *IEEE Trans. Automat. Sci. Eng.*, vol. 14, no. 1, pp. 249–259, Jan. 2017.

[23] A. Dalla Libera, D. Romeres, D. K. Jha, B. Yerazunis, and D. Nikovski, “Model-based reinforcement learning for physical systems without velocity and acceleration measurements,” *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 3548–3555, Apr. 2020.

[24] G. Gregorčič and G. Lightbody, “Gaussian process approach for modelling of nonlinear systems,” *Eng. Appl. Artif. Intell.*, vol. 22, no. 4–5, pp. 522–533, 2009.

[25] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, “Dynamic systems identification with Gaussian processes,” *Math. Comput. Modelling Dynamical Syst.*, vol. 11, no. 4, pp. 411–424, 2005.

[26] G. Casiez, N. Roussel, and D. Vogel, “1€ filter: A simple speed-based low-pass filter for noisy input in interactive systems,” in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2012, pp. 2527–2530.

[27] P. A. Gorry, “General least-squares smoothing and differentiation of nonuniformly spaced data by the convolution method,” *Anal. Chem.*, vol. 63, no. 5, pp. 534–536, 1991.

[28] A. Savitzky and M. J. Golay, “Smoothing and differentiation of data by simplified least squares procedures,” *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, 1964.

[29] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2015, *arXiv:1412.6980*.

[30] A. F. L. Lopera, “Gaussian process modelling under inequality constraints,” Ph.D. dissertation, Université de Lyon, Lyon, France, 2019.

[31] J. Riihimäki and A. Vehtari, “Gaussian processes with monotonicity information,” in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 645–652.