



Delft University of Technology

## Guided Bayesian Optimization Data-Efficient Controller Tuning With Digital Twin

Nobar, Mahdi; Keller, Jurg; Rupenyan, Alisa; Khosravi, Mohammad; Lygeros, John

DOI

[10.1109/TASE.2024.3454176](https://doi.org/10.1109/TASE.2024.3454176)

Publication date

2025

Document Version

Final published version

Published in

IEEE Transactions on Automation Science and Engineering

### Citation (APA)

Nobar, M., Keller, J., Rupenyan, A., Khosravi, M., & Lygeros, J. (2025). Guided Bayesian Optimization: Data-Efficient Controller Tuning With Digital Twin. *IEEE Transactions on Automation Science and Engineering*, 22, 11304-11317. <https://doi.org/10.1109/TASE.2024.3454176>

### Important note

To cite this publication, please use the final published version (if applicable).  
Please check the document version above.

### Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

### Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.  
We will remove access to the work immediately and investigate your claim.

***Green Open Access added to TU Delft Institutional Repository***

***'You share, we take care!' - Taverne project***

**<https://www.openaccess.nl/en/you-share-we-take-care>**

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

# Guided Bayesian Optimization: Data-Efficient Controller Tuning With Digital Twin

Mahdi Nobar<sup>ID</sup>, Jürg Keller, *Senior Member, IEEE*, Alisa Rupenyan, *Member, IEEE*,  
 Mohammad Khosravi<sup>ID</sup>, and John Lygeros<sup>ID</sup>, *Fellow, IEEE*

**Abstract**—This article presents the guided Bayesian optimization (BO) algorithm as an efficient data-driven method for iteratively tuning closed-loop controller parameters using a digital twin of the system. The digital twin is built using closed-loop data acquired during standard BO iterations, and activated when the uncertainty in the Gaussian Process model of the optimization objective on the real system is high. We define a controller tuning framework independent of the controller or the plant structure. Our proposed methodology is model-free, making it suitable for nonlinear and unmodelled plants with measurement noise. The objective function consists of performance metrics modeled by Gaussian processes. We utilize the available information in the closed-loop system to progressively maintain a digital twin that guides the optimizer, improving the data efficiency of our method. Switching the digital twin on and off is triggered by our data-driven criteria related to the digital twin's uncertainty estimations in the BO tuning framework. Effectively, it replaces much of the exploration of the real system with exploration performed on the digital twin. We analyze the properties of our method in simulation and demonstrate its performance on two real closed-loop systems with different plant and controller structures. The experimental results show that our method requires fewer experiments on the physical plant than Bayesian optimization to find the optimal controller parameters.

**Note to Practitioners**—Industrial applications typically are difficult to model due to disturbances. Bayesian optimization is a data-efficient iterative tuning method for a black box system in which the performance can only be measured given the control parameters. Iterative measurements involve operational costs. We propose a guided Bayesian optimization method that uses all information flow in a system to define a simplified digital twin of the system using out-of-the-box methods. It is continuously updated with data from the system. We use the digital twin instead of the real system to perform experiments

and to find optimal controller parameters while we monitor the uncertainty of the resulting predictions. When the uncertainty exceeds a tolerance threshold, the real system is measured, and the digital twin is updated. This results in fewer experiments on the real system only when needed. We then demonstrate the improved data efficiency of the guided Bayesian optimization on real-time linear and rotary motor hardware. These common industrial plants need to be controlled rigorously in a closed-loop system. Our method requires 57% and 46% fewer experiments on the hardware than Bayesian optimization to tune the control parameters of the linear and rotary motor systems. Our generic approach is not limited to the controller parameters but also can optimize the parameters of a manufacturing process.

**Index Terms**—Learning control systems, iterative methods, database systems, control systems, optimization methods.

## I. INTRODUCTION

**M**ECHATRONIC systems require periodic retuning to deal with the uncertainties caused by the system's operation. The controller parameters are typically conservative in handling the unknown and time-varying characteristics of the industrial systems. Auto-tuning methods for adjusting the controller parameters replace the awkward manual controller tuning [1], [2]. Classical model-based controllers derive a model for the system by identification or first-principle methods. The symmetric optimum tuning methods use a performance metric such as phase margin or the closed-loop system's bandwidth to tune the controller [3], [4]. Global optimization algorithms typically require a precise plant model or many trials. For example, given a performance-based objective function in a fixed boundary, particle swarm optimization [5] determines appropriate controller gains [6]. The genetic algorithm used to tune PID controller parameters dealing with complex cost functions where the gradient-based methods struggle [7], [8].

One can directly incorporate available plant data in the control parameter tuning and the control algorithm which can relieve the modeling task of complicated systems. Convolution-based data-driven method [9] simulates the feedback system without building any model of it that accounts for the controller nonlinearities. This method can control rotary actuators by accurately estimating the control input and system response [10], but it holds a strong assumption that the plant is a linear time-invariant system. Reachable zone-based pure data-driven controller design uses only noisy input-output data of a trajectory of the system [11] independent from the

Manuscript received 12 July 2024; accepted 13 August 2024. Date of publication 5 September 2024; date of current version 8 April 2025. This article was recommended for publication by Associate Editor S.-L. Chen and Editor B. Vogel-Heuser upon evaluation of the reviewers' comments. This work was supported by the Swiss National Science Foundation through the National Centre of Competence in Research (NCCR) Automation under Grant 180545. (*Corresponding author: Mahdi Nobar.*)

Mahdi Nobar and John Lygeros are with the Automatic Control Laboratory, ETH Zürich, 8092 Zürich, Switzerland (e-mail: mnobar@ethz.ch; jlygeros@ethz.ch).

Jürg Keller is with the Automation Institute, FHNW, 5210 Windisch, Switzerland (e-mail: juerg.keller1@fhnw.ch).

Alisa Rupenyan is with the ZHAW Centre for AI, Zurich University of Applied Sciences, Zürich, 8401 Winterthur, Switzerland (e-mail: alisa.rupenyan@zhaw.ch).

Mohammad Khosravi is with Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: mohammad.khosravi@tudelft.nl).

Digital Object Identifier 10.1109/TASE.2024.3454176

system's model for controller tuning. Nevertheless, such direct methods are usually less data-efficient than their model-based counterparts, especially when the controller needs to learn about complex real systems with limited data [12].

Integrating the measurements with simulation behaviors allows the data-driven controller to adapt effectively to the real world [13]. Learning from measurement can improve the tracking accuracy of an inverse compensator for a fiber-based piezoelectric actuator [14]. Furthermore, the iterative learning controller combines the process measurements with the optimization framework to provide robustness in the presence of repetitive disturbances and plant-model mismatch [15], [16], [17]. A digital twin (DT) is a virtual replica of a real-world system that continuously updates with measurements from its physical counterpart. DT enables online monitoring and optimization of complex manufacturing processes. For example, [18] explains how a liquid-level control system can be monitored by a DT that encompasses virtual twins of all subsystems. These components are then synchronized with the physical counterpart through a proper bridge between the subsystems such that one can extend the application to control the real process variables [19], [20]. However, there is a gap between the "twinning" property of DTs in industrial systems which limits the DT accuracy. To mitigate the system-twin gap in the controller tuning, [21] incorporates a generic learning algorithm into the DT that enables active braking control of a race car. Reference [22] challenges the sample efficiency of the learning-based algorithms in comparison with the local gradient-based black-box optimization methods. Reference [23] uses the model-based policy iteration to ensure regional optimality by locally approximating the system's behavior.

Industrial closed-loop systems are required to satisfy high precision and performance standards. The controllers in such systems typically contain a set of parameters. One can also measure the performance as a cost function containing predefined metrics based on the system response to a specific signal. Model predictive control (MPC) relies on a model of a system to tune controller parameters that optimize a cost function. For instance, to train MPC parameters on a complex system, [24] defines multiple models to predict a nonlinear system's output using several sampling times. However, a reliable plant model rarely exists, and identifying a potentially nonlinear industrial system in a closed loop is often not feasible without adequate system excitation. To control a hard-to-model industrial process, [25] uses the process input and output data to train a controller with a Kalman filter, reducing the measurement noise effect. Reference [26] tunes such control parameters purely by data acquired while the system operates with repetitive tasks. However, the high operation cost of such industrial systems demands data-efficient controller tuning approaches.

Bayesian optimization (BO) is a data-driven technique that iteratively tunes the controller parameters in a limited number of experiments using a *surrogate model* of the cost function [27]. In complex industrial applications, conducting each of the iterative experiments can involve time-consuming measurements [28]. To increase the BO controller tuning

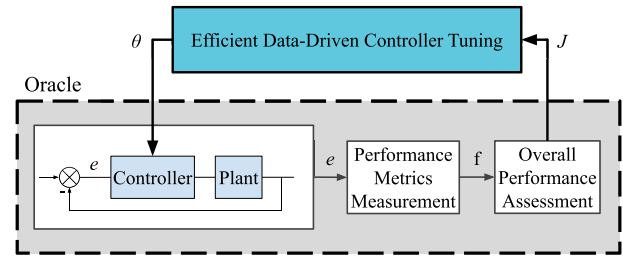


Fig. 1. Efficient data-driven controller tuning scheme based on performance assessment.

efficiency, one can integrate all available process information into the optimizer [29]. For instance, [30] uses coarse physical knowledge about the Plasma Spray Process to encapsulate a linear dependence between the process inputs in the prior mean distribution. In [31] the process information is incorporated in BO to save experimental time in a tedious additive manufacturing application. Furthermore, [32] uses BO to tune the parameters of a goal-oriented model predictive controller that sets the reference input to another cascaded control system to reduce the energy consumption of a thermal management system. This way, the predictive controller's state-space model serves as prior knowledge for BO, ruling out the search space that contradicts the plant's physics. In previous examples, BO is guided by prior information about the process.

This work proposes *Guided Bayesian Optimization* (Guided BO) method that incorporates available closed-loop system information into the optimization. Our data-efficient method reduces the required real system operation to tune the industrial feedback controller parameters iteratively. In our approach, automated controller tuning seeks a global optimum of a cost function that estimates the desired closed-loop system behavior. We use the available data without additional system operation to create a DT. We only take measurements on the real system when the BO surrogate model has sufficiently low uncertainty. Otherwise, we use our DT to estimate the system performance and guide the optimizer. This way, we tune the controller parameters with fewer experiments on the real system, reducing the operating cost. We demonstrate the enhanced performance of the guided BO algorithm in simulation and two real industrial feedback control systems.

Section II introduces the data-driven model-free controller tuning problem. Then, in Section III, we review the BO method and present our guided BO algorithm. Section IV provides preliminary information about the system and controller structure, the performance metrics, and the overall cost function used for the numerical studies in V. Finally, we demonstrate our guided BO method on two experimental setups in VI.

## II. DATA-DRIVEN CONTROLLER TUNING PROBLEM

We consider a feedback control system with possibly a nonlinear plant. Fig. 1 visualizes the components of our controller tuning problem. We have no assumption on the parametric controller structure. Let  $\theta \in \Theta$  be the *controller parameter vector* inside a feasible set  $\Theta \subseteq \mathbb{R}^{n_\theta}$ . We assume the feasible set  $\Theta$  is known (see IV-B and VI-B). For the convergence properties of the closed-loop system, we choose

the set  $\Theta$  such that the step response of the system is bounded in a fixed period.

Given the controller parameter vector  $\theta$ , we denote the cost function  $\hat{J}(\theta)$ , where  $\hat{J} : \Theta \rightarrow \mathbb{R}$  is a scalar metric reflecting the closed-loop performance. There is no assumption on the convexity of the cost function. So  $\hat{J}(\theta)$  may have multiple local minima. We want to simultaneously optimize multiple performance metrics denoted by  $f_i : \Theta \rightarrow \mathbb{R}$  for  $i = 1, \dots, n_f$  and  $n_f > 1$ . These metrics are measured based on the tracking error signal  $e$ , the difference between the reference and output signal. While one approach would be to optimize all the metrics separately determining the Pareto frontier [33], we adopt the classical weighted aggregation method to convert our multi-objective problem to a single-objective problem [34]. So the overall performance cost function  $\hat{J}(\theta)$  is defined as

$$\hat{J}(\theta) := \mathbf{w}^T \mathbf{f}(\theta) = \sum_{i=1}^{n_f} w_i f_i(\theta), \quad (1)$$

where  $\mathbf{f}(\theta) := [f_1(\theta), \dots, f_{n_f}(\theta)]$  and  $\mathbf{w} := [w_1, \dots, w_{n_f}]$ . Since the data collecting process contains uncertainty, we assume that  $J(\theta) := \hat{J}(\theta) + \epsilon$  where  $J(\theta)$  is the value of  $\hat{J}(\theta)$  computed from the noisy data and  $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$  is the measurement noise realization drawn from a normal distribution with zero mean and variance of  $\sigma_\epsilon^2$ . The weights vector  $\mathbf{w}$  adjusts our metrics' priority and scale such that each metric properly contributes to the overall performance.

For practical reasons, we assume that system identification is not feasible to obtain a realistic plant model. For instance, data acquisition from an industrial plant is typically expensive due to operational time and costs. Consequently, we cannot calculate the cost function  $\hat{J}(\theta)$  analytically. So we suppose a *black-box oracle* for our closed loop system that provides the cost value  $\hat{J}$  given the controller parameter vector  $\theta$ . Now our *data-driven controller tuning problem* is to obtain the optimum controller parameter vector  $\theta^*$  by solving

$$\theta^* := \underset{\theta \in \Theta}{\operatorname{argmin}} \hat{J}(\theta). \quad (2)$$

The objective function  $\hat{J}(\theta)$  is available as expensive to evaluate the black box oracle on the real plant.

### III. GUIDED BAYESIAN OPTIMIZATION

The BO-based methods introduced in the literature inefficiently compress the data of each experiment to a single overall performance value. For example, [35], [36], [37] maps only the output tracking error signal of the closed-loop system to a cost function, ignoring explicitly utilizing all the other internal signals. However, intermediary data, such as control signals and output measurements, might be available and usually not explicitly exploited. In this Section, we propose our novel *guided BO* method shown in Fig. 2. Our method utilizes the information flow in the closed-loop system to define a DT, improving the optimization and reducing operational costs.

#### A. Bayesian Optimization-Based Controller Tuning

Bayesian inference is a statistical inference method that uses Bayes' theorem to update the probability of a

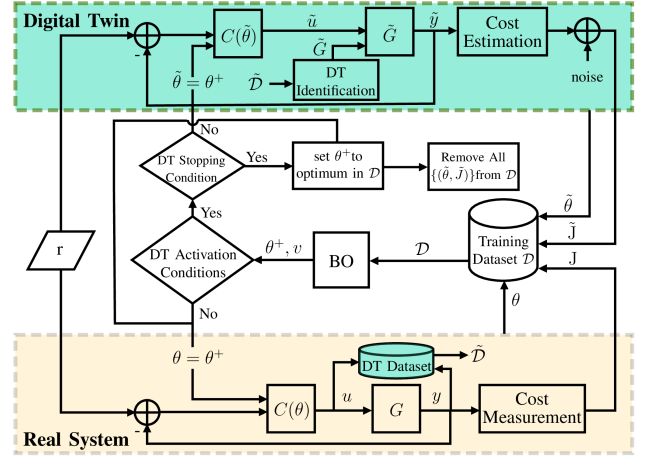


Fig. 2. Guided BO schematic representation. The digital twin of the closed-loop system is built using available data without additional operations on the real system.

hypothesis as more data becomes available [27]. BO can optimize complex “black-box” objective functions using sparse observations [38], [39]. However, the computational disadvantage of the Bayesian inference approach is that it typically solves an intractable integration problem [40]. Gaussian Process (GP) regression trains a stochastic model of data that provides predictive uncertainty and a tractable surrogate model for Bayesian inference.

To build a prior distribution of the mean and covariance functions, BO requires an initial collection of control parameters  $\Theta_{\text{init}} := \{\theta_j \in \Theta \mid j = 1, 2, \dots, N_0\}$ , where  $N_0 \geq 1$  is the number of parameters. We use the Latin hypercube introduced by [41] as a space-filling sampling method to build  $\Theta_{\text{init}}$ . So, each dimension of the feasible set  $\Theta$  is divided into equally probable intervals where random control parameters are drawn. This method selects the control parameters of the initial training data set independently and uniformly in each dimension of the feasible set.

The BO algorithm maintains a pool of parameters and their corresponding performance values as a *training data set*

$$\mathcal{D} := \{(\theta_j, J(\theta_j)) \mid \theta_j \in \Theta ; j = 1, \dots, n; n \geq N_0\}, \quad (3)$$

in which we assume it initially contains  $N_0$  data points. Then, at every BO iteration, we estimate the cost function using Gaussian Process (GP) regression denoted by  $\hat{J}$  based on  $\mathcal{D}$ . More precisely, let  $\mathcal{GP}(\mu, k)$  be the prior Gaussian process function with the prior mean and kernel functions  $\mu : \Theta \rightarrow \mathbb{R}$  and  $k : \Theta \times \Theta \rightarrow \mathbb{R}_{\geq 0}$ . Consider parameter vectors  $\mathbf{X}_n := [\theta_1, \dots, \theta_n]^T$  and performance values  $\mathbf{y}_n := [J(\theta_1), \dots, J(\theta_n)]^T$ , we write

$$\forall \theta \in \Theta : \hat{J}(\theta) \sim \mathcal{N}(\mu_n(\theta), v_n(\theta)), \quad (4)$$

with mean and variance

$$\mu_n(\theta) := \mu(\theta) + \mathbf{k}_n(\theta)^T (\mathbf{K}_n + \sigma_\epsilon^2 \mathbb{I})^{-1} (\mathbf{y}_n - \mu(\mathbf{X}_n)), \quad (5)$$

$$v_n(\theta) := k(\theta, \theta) - \mathbf{k}_n(\theta)^T (\mathbf{K}_n + \sigma_\epsilon^2 \mathbb{I})^{-1} \mathbf{k}_n(\theta), \quad (6)$$

where  $\mathbf{K}_n$  is defined as  $[k(\theta_{\xi_1}, \theta_{\xi_2})]_{\xi_1, \xi_2=1}^n \in \mathbb{R}^{n \times n}$ ,  $\sigma_\epsilon$  is the standard deviation of the additive noise to the predictive performance uncertainty,  $\mathbb{I}$  is the identity matrix, vector



$\mu(X_n) = [\mu(\theta_1), \dots, \mu(\theta_n)]^\top \in \mathbb{R}^n$ , and the vector  $k_n(\theta) \in \mathbb{R}^n$  is equal to  $[k(\theta, \theta_1), \dots, k(\theta, \theta_n)]^\top$ .

The kernel function determines the relationship between neighboring points modeled by the GP. Here, we utilize a Matérn kernel 5/2 that has demonstrated exemplary performance on noisy differentiable functions [40]. We use a constant zero prior mean function without loss of generality. We use available observations to optimize the prior mean and kernel by log marginal likelihood maximization.

To select query parameter vector  $\theta_{n+1}$ , we use an auxiliary *acquisition function* to ensure a trade-off between exploring regions with high uncertainty and exploiting regions with a higher probability of finding the optimum gains. We choose Expected Improvement (EI) [42] acquisition function with a closed-form formulation because it improves the optimization accuracy with a limited number of observations [30], [43], [44]. Let  $a_{EI,n}$  be the expected improvement acquisition function corresponding to the  $n$ -th iteration, defined as

$$a_{EI,n}(\theta) := v_n(\theta)^{\frac{1}{2}} (z_n(\theta) \Phi(z_n(\theta)) + \varphi(z_n(\theta))), \quad (7)$$

with  $z_n(\theta) := (\mu_n(\theta) - J_n^+) v_n(\theta)^{-\frac{1}{2}}$  where  $\mu_n(\theta)$  and  $v_n(\theta)$  are the predictive mean and variance of the GP model,  $J_n^+$  is the best-observed performance among the  $n$  experiments in training data set  $D$ ,  $\Phi(\cdot)$  is standard normal cumulative distribution function, and  $\varphi(\cdot)$  is the standard normal probability density function.

To optimize the acquisition function for the next parameter  $\theta_{n+1}$ , we exhaustively grid search with a fixed number of data within the feasible set  $\Theta$ :

$$\theta_{n+1} := \operatorname{argmax}_{\theta \in \Theta} a_{EI,n}(\theta). \quad (8)$$

Accordingly, we denote the next query point  $\theta^+ := \theta_{n+1}$ , measure the next performance value  $J(\theta^+)$ , and update the *BO training data set* as

$$D = D \cup \{(\theta^+, J(\theta^+))\}. \quad (9)$$

The computed sequence of  $\theta_1, \theta_2, \theta_3, \dots$  by this iterative optimization converges to the proper solution of (1) [45], [46].

The computational complexity of BO controller tuning at iteration  $m$  is heavily influenced by inversion of the covariance matrix  $K_n$  in (6) and is approximated by  $O(m.n^3)$  where  $n$  is the size of training data set  $D$ .

## B. Digital Twin

In the BO-based controller tuning method, we already acquire informative data in each experiment so one could constantly learn a digital counterpart of the closed loop system. Starting from an initial training data set with size  $N_0$ , we build the *digital-twin data set*  $\tilde{D}$  at experiment  $n$  as

$$\tilde{D} := \bigcup_{j=1}^n \bigcup_{i=1}^{\tilde{n}} \{(u_{i,j}, y_{i,j})\}, \quad (10)$$

where  $\tilde{n}$  is the sampling length per each experiment,  $u$  and  $y$  are the real plant input and output, respectively. For simplicity, we assume all the experiments have the same length  $\tilde{n}$ .

We estimate an approximate model of the real plant called *DT plant* denoted by  $\tilde{G}$  based on  $\tilde{D}$ . Assume  $\tilde{G}$  is a linear time-invariant transfer function estimated by the refined instrumental variable method [47]. This DT plant model is computationally cheap and does not require fast processing. So our DT is the same closed-loop system but with a DT plant instead of the real plant.

The DT plant must have sufficient fidelity (see Section V-A). The fidelity of the DT plant model is estimated by Root Mean Square Error (RMSE)

$$e_{\text{RMSE}} := \left[ \frac{1}{n \times \tilde{n}} \sum_{k=1}^n \sum_{j=1}^{\tilde{n}} (y_j - \tilde{y}_j)^2 \right]^{\frac{1}{2}}, \quad (11)$$

where  $\tilde{y}$  is the output predicted by the DT plant.

## C. Guided Bayesian Optimization

Algorithm 1 summarizes our guided BO method. At each BO iteration using the physical plant  $G$  and given the query control parameters  $\theta^+$ , we collect the control signal  $u$  and the plant output  $y$  to build DT data set  $\tilde{D}$  and identify a DT plant model  $\tilde{G}$ . The DT plant is updated each time the performance of the closed-loop control system with the physical plant is measured. Based on the GP model uncertainty, the guided BO algorithm gains more information from the updated DT. More specifically, we activate the DT when two conditions

$$2\sqrt{v_n(\theta^+)} > \eta_1, \quad (12a)$$

$$e_{\text{RMSE}} < \eta_2, \quad (12b)$$

are satisfied where  $\eta_1 \geq 0$  is the DT activation threshold,  $v_n$  is the variance of the BO's posterior GP model calculated at the next query parameter vector  $\theta^+$ ,  $e_{\text{RMSE}}$  is RMSE defined in (11), and  $\eta_2$  is the RMSE threshold and is chosen based on the noise standard deviation as  $\eta_2 := 3\sigma_\epsilon$ . We estimate  $\sigma_\epsilon^2$  by measuring the performance metrics in five experiments with identical controller gains in the feasible set. When the DT is activated, we use the DT instead of the actual system to estimate the cost value denoted by  $\tilde{J}$  at  $\tilde{\theta} := \theta^+$ . We add a Gaussian noise with zero mean and  $\sigma_\epsilon^2$  variance to the estimated performance to obtain  $\tilde{J} := \hat{J} + \epsilon$  where  $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ . This estimated performance  $\tilde{J}$  is then added to the BO training data set in Line 11 of Algorithm 1. In Section V-C, we numerically study the DT activation threshold  $\eta_1$ .

We continuously monitor the DT quality to maintain optimum controller parameters when the real plant changes. Each time we measure the performance on the real system  $J$ , we compare it with its estimated value by DT denoted by  $\tilde{J}$ . Suppose that the system behavior has considerably changed. In that case, we discard all previous data from the DT dataset. More specifically, if the normalized difference between the measured and estimated costs is considerable, i.e.,

$$\frac{|J - \tilde{J}|}{J} > \tilde{\delta}, \quad (13)$$

where  $\tilde{\delta}$  is DT re-initialization threshold, then we reinitialize DT dataset with the recent measurements as in Line 24 of Algorithm 24. If the system significantly changes, a new safe

**Algorithm 1** Guided BO Controller Tuning Algorithm

---

```

1: Input: Set feasible set  $\Theta$ , initial control parameters  $\Theta_{\text{init}}$ ,
   weight vector  $w$ , DT sampling length  $\tilde{n}$ , DT activation
   parameters  $\eta_1$  and  $\eta_2$ , DT re-initialization parameter  $\tilde{\delta}$ ,
   maximum experiments  $n_{\text{max}}$ , DT stopping threshold and
   consecutive iteration parameters  $\eta_3$  and  $\tilde{n}_{\text{EI}}$ 
2: Initialize  $\mathcal{D}$  and  $\tilde{\mathcal{D}}$  using  $G$  to measure  $u$ ,  $y$ , and
    $J(\theta)$ ,  $\forall \theta \in \Theta_{\text{init}}$ 
3: while  $n_{\text{max}}$  experiments on  $G$  is not reached do
4:   Optimize  $\mathcal{GP}(\mu, k)$  prior mean and kernel hyperpa-
   rameters by minimizing negative log marginal likelihood
5:   Estimate cost  $\hat{J}$  as in (4)
6:   Derive next query point  $\theta^+$  solving (8)
7:   if DT activation criteria in (12) is met then
8:     Update DT plant model  $\tilde{G}$ 
9:     while DT stopping criterion in (14) is not met do
10:      Use  $\tilde{G}$  to estimate cost  $\tilde{J}$  at  $\tilde{\theta} = \theta^+$ 
11:      Update BO training data:  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\tilde{\theta}, \tilde{J})\}$ 
12:      Optimize  $\mathcal{GP}(\mu, k)$  prior hyperparameters
13:      Estimate cost  $\hat{J}$  as in (4)
14:      Derive next query point  $\theta^+$  solving (8)
15:    end while
16:    Set  $\theta^+ = \theta_m \in \mathcal{D}$  s.t.  $\forall \theta \in \mathcal{D} : J(\theta_m) < J(\theta)$ 
17:    Remove all DT data  $(\tilde{\theta}, \tilde{J})$  from  $\mathcal{D}$  in 3
18:  end if
19:  Use  $G$  to measure performance  $J$  at  $\theta = \theta^+$ 
20:  Update BO training data:  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\theta, J)\}$ 
21:  Use  $\tilde{G}$  to estimate cost  $\tilde{J}$  at  $\tilde{\theta} = \theta^+$ 
22:  Update DT data:  $\tilde{\mathcal{D}} \leftarrow \tilde{\mathcal{D}} \cup \left\{ \bigcup_{i=1}^{\tilde{n}} (u_i, y_i) \right\}$ 
23:  if DT re-initialization criterion in (13) is met then
24:    Remove previous DT data:  $\tilde{\mathcal{D}} \leftarrow \left\{ \bigcup_{i=1}^{\tilde{n}} (u_i, y_i) \right\}$ 
25:  end if
26: end while

```

---

boundary for the optimization parameters must be determined, as demonstrated in [48], [49], for example.

An appropriate stopping criterion for the optimization process is necessary for the practical implementation of the algorithm. We terminate the BO after a fixed number of iterations on the real system denoted by  $n_{\text{max}}$ . When using DT, we need a stopping condition that ensures prompt termination of the optimization. We use the stopping criterion that depends on the maximal expected improvement over previous iterations and the current expected improvement proposed in [35]. Once we have activated the DT plant to predict the closed-loop response, we stop the BO iterations on DT when the maximum number of iterations on the DT is reached or depending on the ratio between the last expected improvement and the maximal expected improvement realized so far. This way, we avoid unnecessary iterations using the DT plant where we anticipate no further improvement. Notably, we stop BO on DT when for  $\tilde{n}_{\text{EI}}$  consecutive iterations on DT we meet one of the conditions

$$2a_{\text{EI},i}(\theta) \leq \eta_3 \max_{i-\tilde{n}_{\text{EI}} \leq j \leq i-1} a_{\text{EI},j}(\theta), \quad (14a)$$

$$i > n_{\text{max}}, \quad (14b)$$

where  $i$  is the number of BO iterations on DT *after* last time DT is activated,  $a_{\text{EI},i}(\theta) = a_{\text{EI}}(\theta^+)$  and  $\eta_3 > 0$  is a predefined threshold to stop BO iteration on DT as in [50]. If the expected improvement does not significantly improve compared with the improvements in the last  $\tilde{n}_{\text{EI}}$  iterations, then (14a) halts the optimization with DT.

As soon as we stop using DT, we update the query parameter vector  $\theta^+$  with the optimum observed parameter vector in the BO training data set  $\mathcal{D}$  as defined in line 16 of Algorithm 1. Then, we remove all the DT data  $(\tilde{\theta}, \tilde{J})$  from  $\mathcal{D}$ . We eventually proceed with the actual system to measure the overall performance at  $\theta^+$ .

The training data set size  $n$  changes when we remove the DT data from  $\mathcal{D}$  (see line 17 of Algorithm 1). Furthermore, at each iteration  $m$  on the real plant (where  $0 \leq m \leq n_{\text{max}}$ ), the DT can be activated and  $i$  number of BO iterations can be performed on the DT where  $i \leq n_{\text{max}}$ . So the computational complexity of the guided BO algorithm is bounded and dominated by a polynomial that depends on  $i$ ,  $m$ , and  $n$ .

#### IV. SYSTEM MODELLING

Our guided BO is independent of the plant or parametric controller structure. In this Section, we first specify the controller and the plant structure to numerically evaluate the guided BO algorithm in Section V. Then, we explain our chosen performance metrics used to assess the system's overall performance in (1).

We have a DC rotary motor with speed encoders equipped with a current converter and a cascaded speed controller. As the two controllers operate on different time scales, we only consider the speed controller and aim to optimize its proportional and integral gains. The DC motor with its encoder and the current converter is denoted by  $G$ .

##### A. System Structure and Identification

For the numerical studies in Section V, the real plant  $G$  is an identified model of the DC motor system around an operation point as a linear time-invariant (LTI) system. We have a speed encoder with LabVIEW-based real-time interface [51]. The nominal encoder speed  $v$  is the number of pulses counted with sampling time  $t = 2$  ms. We consider low and high reference encoder nominal speed inputs equal to  $v = 1600$  and  $v = 2000$ , respectively. Our encoder can count geometrically  $N = 2048$  pulses per revolution. So the angular speed in radian per second can be calculated by  $\omega = \frac{2\pi \cdot v}{N \cdot t}$ .

We measure the plant response to a given sinusoidal signal with various frequencies, a fixed amplitude, and a single DC value of  $\bar{\Omega} = 5522.40$  rad/s. Considering several excitation frequencies, the excitation signal has a fixed amplitude of 613.60 rad/s. We measure the system output with a high sampling rate of 0.5 ms and draw the bode diagram of the system response, which allows us to fit a second-order LTI transfer function visually. We repeat each measurement two times to alleviate the measurement noise. For A/D conversion, we sample 128 points evenly distributed in every period, introducing a constant phase shift in our bode diagram. So, because we have 12 different frequencies for the sinusoidal excitation

TABLE I

IDENTIFICATION PARAMETERS OF THE LINEAR TIME-INVARIANT PLANT MODEL AROUND A SPECIFIED OPERATING POINT

Identified Plant Parameters		Identification Specifications	
Parameter	Value	Parameter	Value
$\tau_1$	4.145	number of poles	2
$\tau_2$	4.199	number of frequencies	12
$L$	9.544	excitation amplitude [rad/s]	613.60
$l$	0.002	sampling rate [Hz]	10-2000

signal, the sampling rate varies (see Table I). Consequently, an input time delay of 2 ms abstracts the time shift introduced by A/D conversion. We address this delay by shifting the phase in the bode diagram to a fixed offset corresponding to the delay. Finally,  $G$  is written as  $G(s) := \frac{L}{s^2 + \tau_1 s + \tau_2} e^{-sl}$ , where Table I summarizes its identified parameters.

### B. Controller Structure and Nominal Tuning

A PI speed controller

$$C(s) := K_p(1 + K_i \frac{1}{s}), \quad (15)$$

with  $K_p$  and  $K_i$  gains is defined. For the controller parameter vector  $\theta := [K_p, K_i]$ , we define a *feasible set*

$$\Theta := \left\{ [K_p, K_i] \mid K_p \in [K_{p_{\min}}, K_{p_{\max}}], K_i \in [K_{i_{\min}}, K_{i_{\max}}] \right\}. \quad (16)$$

We set the boundaries  $K_{p_{\min}}, K_{p_{\max}}, K_{i_{\min}}, K_{i_{\max}}$  equal to 0.11, 1.10, 0.87, 2.08, respectively, defining a rectangle around stable nominal controller parameters where the overshoot is bounded.

Classical tuning methods for the controller in (15) involve numerical or graphical approaches using the Bode diagram [52]. We design a *nominal controller* based on the real closed-loop system's phase margin and gain margin (PGM) [53]. We analytically calculate the controller gains given PGM specifications with the simple approximations introduced by [54] as follows

$$\omega_p := \frac{A_m \Phi_m + \frac{1}{2} \pi A_m (A_m - 1)}{(A_m^2 - 1)l}, \quad (17)$$

$$K_p := \frac{\omega_p \tau}{A_m L}, \quad (18)$$

$$K_i := 2\omega_p - \frac{4\omega_p^2 l}{\pi} + \frac{1}{\tau}, \quad (19)$$

where  $\omega_p$  is the approximated phase crossover frequency. We choose two value pairs for phase and gain margins to estimate nominal controllers that assure the under-damped behavior of the closed loop system. Table II lists the resulting nominal controller gains and their PGM.

### C. Performance Metrics

Following [50], we define four performance indicators namely percentage overshoot ( $\zeta$ ), settling time ( $T_s$ ), rise time ( $T_r$ ) and the integral of time-weighted absolute error ( $e_{ITAE}$ ).

TABLE II

SPECIFICATIONS OF NOMINAL CONTROLLERS

Controller name	$\Phi_m$	$A_m$	$K_p$	$K_i$
First Nominal PGM	60°	46 dB	0.85	1.07
Second Nominal PGM	75°	46 dB	0.86	0.89

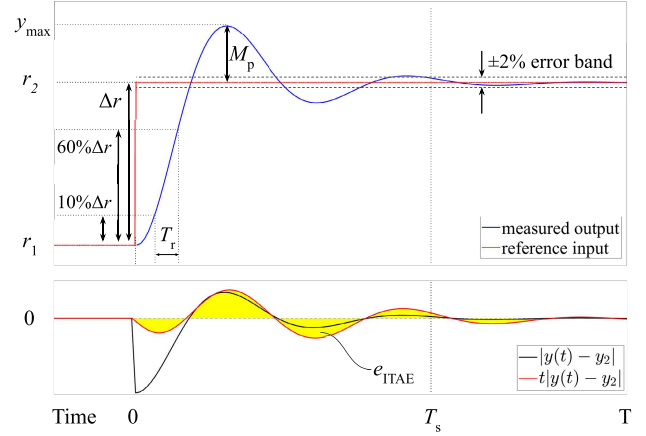


Fig. 3. Details of the performance metrics used to define the overall cost function.

1) *Percentage Overshoot*: If  $y(t)$  is the step response of a continuous-time system, the percentage overshoot at each transition from low initial state  $r_1$  to the high final reference input  $r_2$  is defined such that

$$\zeta := 100 \times \max(0, \frac{M_p}{\Delta r}), \quad (20)$$

where  $y_{\max}$  is the maximum response in the transition period,  $\Delta r := r_2 - r_1$  is the step height,  $M_p := y_{\max} - y(T)$  is the peak overshoot, and  $T$  is the final time. Percentage overshoot presents the stability degree and damping in the system [55].

2) *Settling Time*: This is the time that the system response takes to converge and stay within an error band, i.e., 2% of the step height, from the final value [56]. Settling time indicates a closed-loop system's convergence time to reach its steady state. The settling time  $T_s$  is defined as

$$\forall t \geq T_s : |y(t) - r_2| \leq \frac{2}{100} \times \Delta r. \quad (21)$$

3) *Rise Time*:  $T_r$  indicates the time for the step response to rise from 10% to 60% of the way from  $r_1$  to  $r_2$  (see Fig. 3). So, a shorter rise time indicates a rapid response to the changes in the input signal.

4) *Integral of Time-Weighted Absolute Error*: Integral of time-weighted absolute error (ITAE) is the absolute time-weighted area of the step response error from the reference signal. This performance index reduces system overshoot and oscillations [57], allowing a smoother response. We write ITAE as

$$e_{ITAE} := \int_0^T t |y(t) - r_2| dt, \quad (22)$$

where the final time  $T \geq T_s$  is a fixed period. Fig. 3 visualizes details of our selected performance metrics.

According to the overall cost function in (1), a weighted sum of four performance metrics is defined. The system performs



well if its step response rises fast enough with minimum overshoot and fluctuations to a stable bandwidth. We can write

$$J(\theta) = w_1\zeta + w_2T_s + w_3T_r + w_4e_{ITAE}, \quad (23)$$

such that  $\sum_{i=1}^4 w_i = 1$  and  $w_i > 0$  where  $w_i$  are the normalized weights of our selected individual performance metrics. Different weights can be used to attain different trade-offs between the performance metrics; in the experiments below, we measure each performance metric using 5 controller gains evenly spanning the entire feasible set  $\Theta$ , then take the average of each metric as a normalization weight.

We adjust the weights based on the relative significance of each metric. For instance, we put more relevant gravity on minimizing overshoot related to performance and stability. The normalized weights  $w_1, w_2, w_3, w_4$  are respectively selected as 0.44, 0.22, 0.22, 0.11, summing up to unit value.

The following Section will analyze our guided BO properties on the identified plant G.

## V. NUMERICAL ANALYSIS

We numerically study the guided BO properties using the controller (15) and plant G comparing with the nominal controller tuning. Computations are performed by the Euler high-performance cloud-based cluster of ETH Zurich [58]. We use 8 AMD EPYC 7742 CPUs cores with a clock speed of 2.25 GHz nominal and 3.4 GHz peak and 3072 MB of memory.

The reference input  $r$  of the closed-loop system equals the angular encoder speed  $\omega$  in radian per second. We consider low and high reference encoder speed inputs equal to  $r_1 = 245$  rad/s and  $r_2 = 306$  rad/s corresponding to the nominal encoder speeds of  $v = 1600$  and  $v = 2000$ , respectively. We model the output measurement noise by an additive Gaussian noise with zero mean and standard deviation  $\sigma_\epsilon = 0.03$ .

We calculate the ground truth speed controller parameters  $\theta^*$  with an exhaustive grid search in the feasible set  $\Theta$  such that the cost in (23) is minimized. The *optimality ratio*  $\phi$  is the ratio between the cost and the ground truth optimum cost value  $J(\theta^*)$ . We calculate once the  $\theta^* \in \Theta$  with a dense grid search where the minimum cost on the numerical system is retrieved (see Table VI). The optimality ratios of our nominal controllers in Table II are  $\phi = 1.71$  and  $\phi = 1.49$ , respectively.

We will analyze the DT plant fidelity, the proportion of BO iterations on DT, and the number of data in the initial training data set  $\mathcal{D}$  on the identified plant. To build the DT data set in (10), for simplicity, we fix the DT re-initialization threshold  $\tilde{\delta} = 2$  and the sampling length  $\tilde{n} = 100$  in a fixed period  $T$  for all experiments. We summarize the parameter values used in our guided BO algorithm in Table III.

### A. Digital Twin Fidelity

We identify two DT plants using second-order and fifth-order LTI models to assess the DT fidelity on the guided BO performance, as described in Section III-B. We sample uniform data from the actual system's step response over a fixed period  $T = 5$  s, which is longer than the nominal closed-loop settling time. We compare the fidelity of these two DT plant models

TABLE III

PARAMETER VALUES OF OUR GUIDED BO USED IN CORRESPONDING SECTIONS, INCLUDING THE RANGE FOR VARIED PARAMETERS

Section	System	$\eta_1$	$\eta_2$	$\eta_3$	$N_0$	$\tilde{n}_{EI}$	$n_{\max}$
V-A	numerical	0- $\infty$	0.09-2.5	$\infty$	1	5-40	50
V-B	numerical	0- $\infty$	0.09	$\infty$	1-10	5	50
V-C	numerical	$10^{-6}$ -20	0.09	0.2	1	3	50
V-D	numerical	3	0.09	0.2	1	3	35
VI-A	rotary motor	3	0.09	0.2	1	3	40
VI-B	linear motor	3	0.003	0.2	1	3	25

TABLE IV

MONTE CARLO RESULTS WITH 100 BATCHES OF 50 EXPERIMENTS ON THE REAL PLANT, EACH STARTING FROM A DIFFERENT INITIAL DATA SET  $\mathcal{D}$ . COLUMNS 3 – 5 REPORT THE ROUNDED AVERAGE NUMBER OF EXPERIMENTS OVER THE REAL PLANT REQUIRED TO OUTPERFORM THE NOMINAL OPTIMALITY RATIO  $\phi = 1.49$

$\tilde{n}_{EI}$	$N_G$	BO	Guided BO	
		no DT	high-fidelity DT	low-fidelity DT
5	1	11	7	>50
5	3	11	9	16
5	5	11	7	16
5	7	11	8	15
10	1	11	6	16
10	3	11	8	16
10	5	11	7	16
20	1	11	7	15
40	1	11	7	16

with RMSE in (11) at the beginning of optimization using the DT data set  $\tilde{\mathcal{D}}$ . Second order DT plant is a *high-fidelity* model with RMSE of  $e_{\text{RMSE}} = 0.01$ ; whereas the fifth-order model called *low-fidelity* DT plant has a higher RMSE equal to  $e_{\text{RMSE}} = 2.41$ .

We run a Monte Carlo analysis of the guided BO algorithm using both DTs with 100 batches of experiments. Each batch includes  $n_{\max} = 50$  experiments on the real plant starting from an initial data set with a single data  $N_0 = 1$ . The controller gains of the initial data set are randomly selected inside the feasible set. So, the initial data sets of the batches are different from each other. Starting from the initial training data set, we set  $\eta_1 = 0$  in (12a) and perform BO on the DT plant  $\tilde{n}_{EI}$  times. Each time we activate DT, we continue BO iterations on DT up to a fixed number of iterations  $\tilde{n}_{EI}$ . Next, we deactivate the DT by setting  $\eta_1 = \text{inf}$ , and continue  $N_G$  iterations on the real plant until we activate the DT plant by setting  $\eta_1 = 0$ . This way, we activate DT periodically after every  $N_G$  iteration on the real plant G. Here we set  $\eta_2 = 0.09$  and  $\eta_2 = 2.5$  when using high- and low-fidelity DT plants, respectively.

Results on Table IV show that with the same number and frequency of DT integration, the guided BO with high-fidelity DT converges to a desired optimality ratio sooner than BO. Thus, the DT directly affects the convergence of the guided BO method.

### B. Initial Training Set Size

We study how the number of initial data  $N_0$  measured on the real plant contributes to the performance. We perform Monte Carlo analysis with 100 batches of experiments. For each batch, we build the initial training data set  $\mathcal{D}$  by selecting

TABLE V

MONTE CARLO RESULTS ON 100 BATCHES OF EXPERIMENTS WITH DIFFERENT INITIAL DATA SET SIZE  $N_0$ . EACH ROW IS THE ROUNDED AVERAGE NUMBER OF ITERATIONS ON THE REAL PLANT REQUIRED TO OUTPERFORM THE NOMINAL CONTROLLER

$N_0$	$\phi = 1.71$		$\phi = 1.49$	
	Guided BO	BO	Guided BO	BO
1	5	9	13	17
5	4	5	14	14
10	5	6	17	19

$N_0$  number of controller vectors  $\theta$  using a Latin hyper-cube sampling method inside the feasible set  $\Theta$ . According to Table V, for a small  $N_0$ , the guided BO converges on average 3 experiments faster than BO. In this case, DT compensates for insufficient initial information. The DT benefit for the guided BO diminishes when the initial training set contains more data.

### C. Digital Twin Activation Threshold

We determine the DT activation threshold  $\eta_1$  in (12a). At each threshold  $\eta_1$ , we repeat the guided BO algorithm 100 times using different initial data sets. Fig. 4 shows that guided BO tunes the controller gains faster than BO when the DT is sufficiently activated, allowing the average optimality ratio of the measured performance to be low enough. More precisely, we observe in Fig. 4a that for guided BO, choosing a DT activation threshold in the range  $\eta_1 \in [2, 8]$  (corresponding to DT activation around 12 times on average in each batch) results in improved convergence of approximately 9 experiments to the nominal performance compared to the BO method. We also observe that with large  $\eta_1 = 20$ , the guided BO approaches the BO's performance indicated by dashed lines.

Fig. 4 indicates the correlation between the total DT activations and the average optimality ratio measured on the real plant using various  $\eta_1$  thresholds. For the DT activation threshold  $\eta_1 \geq 5$  in 12a, increasing  $\eta_1$  decreases the number of times DT guides the optimizer and increases the average optimality ratio measured on the real plant. However, the cost estimated by DT is inaccurate, and over-activating the DT results in lower performance. That is why using small  $\eta_1 = 1e-6$  results in the excessive activation of the DT, in which the average optimality ratio also increases to 2.2. Therefore, choosing the DT activation threshold in the range  $\eta_1 \in [2, 8]$  provides a better trade-off between the DT activation and the guided BO performance.

### D. Performance Evaluation

A Monte Carlo study is performed with 100 batches of experiments. Table III specifies the guided BO algorithm parameter values. According to V-C, the parameter  $\eta_1 = 3$  is chosen to provide us with an optimum trade-off between DT activation and the convergence speed of the algorithm.  $\eta_2 = 0.09$  is chosen concerning three times the system noise boundary compared to the expected fidelity of the DT through RMSE error. We set the stopping threshold  $\eta_3 = 0.2$  and

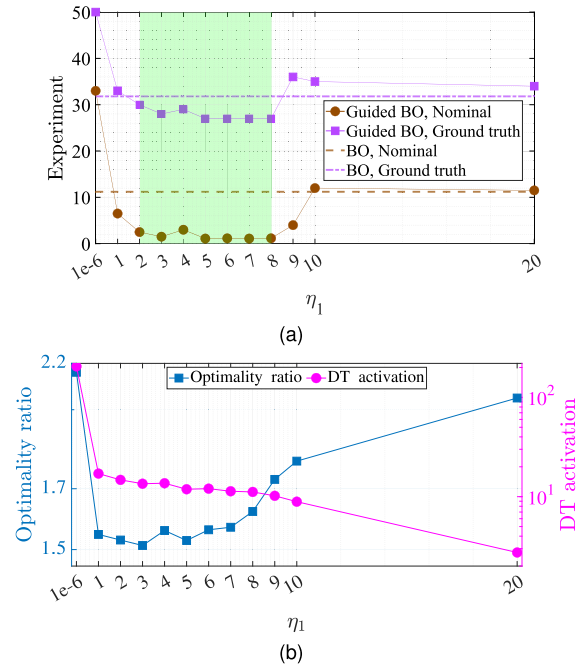


Fig. 4. Results for different activation threshold  $\eta_1$  of DT in guided BO method. Each data point averages over 100 batches, each including 50 experiments on the real plant. (top) The required number of experiments on the real system to outperform the nominal controller performance or to converge to the ground truth performance  $J(\theta^*)$ . The shaded area indicates  $\eta_1$  range where guided BO converges faster to the ground truth performance than BO. (bottom) The left axis is the average optimality ratio. The right axis shows the average number of DT activations in each batch.

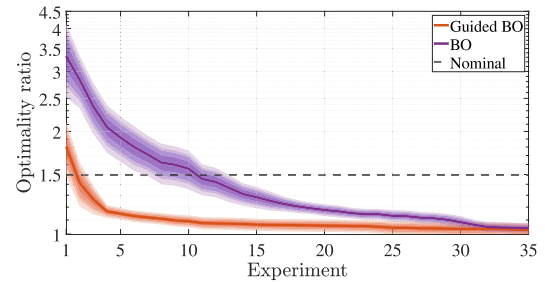


Fig. 5. Minimum observed optimality ratio up to number of BO experiments on the real plant. The thick line is the average over 100 batches, and the shaded area shows the 99%, 95%, 90%, and 68% confidence intervals.

$\tilde{n}_{EI} = 3$  to avoid unnecessary optimization on the DT. The minimum number of initial data  $N_0 = 1$  is chosen to provide the GP model with the least prior information when the guided BO is shown to improve the optimization further compared to BO as revealed in section V-B. The results are shown in Fig. 5. Guided BO with high-fidelity DT outperforms the nominal controller on average after approximately 2 experiments, whereas BO requires approximately 12 experiments.

We compare the evaluated controller parameters of guided BO and BO in a batch of experiments starting from the same initial controller parameters  $\Theta_{init}$ . In Fig. 6, we visualize one batch of our Monte Carlo analysis consisting of 35 control parameters proposed by each optimizer to measure the performance with the real plant. The true cost surface is presented with a colormap in Fig. 6 which is a non-convex function of the controller parameters. This figure shows that the BO (magenta squares) requires exploring the feasible set. Fig. 6 (bottom panel) depicts the cost estimated by DT after

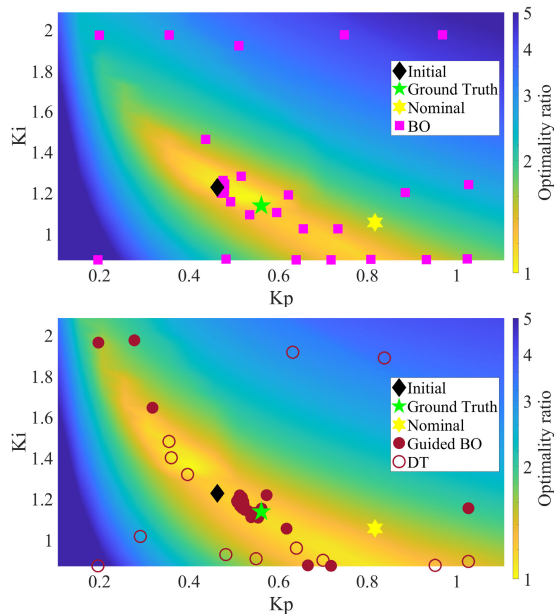


Fig. 6. Control parameters proposed by the acquisition function to measure performance in one batch of 35 iterations in guided BO and BO starting from the same initial data set. **(top)** Colormap shows *true cost* surface using the real plant. **(bottom)** Colormap shows the *estimated cost* by DT plant after 34 iterations on the numerical system. Hollow circles are the gains where DT estimates the performance.

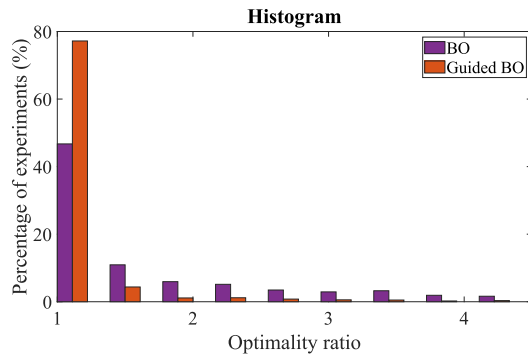


Fig. 7. Histogram of all measured performance values on the real plant in 100 batches of 35 experiments.

34 iterations on the real plant where the filled circles represent the measured parameters by guided BO on the real plant, and the hollow circles show the DT evaluations. We see that DT retrieves the true cost. Thus, the guided BO exploits the optimum region, avoiding the regions with higher costs.

The histogram in Fig. 7 shows the percentage of measurements on the real plant in each optimality ratio range during our Monte Carlo analysis. This histogram indicates a higher proportion of the guided BO experiments around the optimum, allowing the optimizer to exploit more around the optimum region. The BO requires more exploration with higher optimality ratios.

Table VI specifies the tuned controller parameters and corresponding cost values. The nominal controller is the PGM controller introduced in subsection IV-B with phase margin of  $60^\circ$ . The ground truth optimum controller gains are retrieved by dense grid search in the feasible set on the numerical system. According to this table, the tuned controller by our guided BO method has lower cost than BO and the nominal

TABLE VI

COMPARISON OF OPTIMUM COST AND CONTROLLER PARAMETERS TUNED BY GROUND TRUTH GRID SEARCH, NOMINAL, BO AND GUIDED BO METHODS AFTER  $n = 21$  ITERATIONS ON G

Controller	$K_p$	$K_i$	$\hat{J}([K_p, K_i])$
Ground truth grid search	0.54	1.16	0.79
Nominal ( $\Phi_m = 60^\circ$ )	0.85	1.07	1.05
BO	0.45	1.18	0.84
Guided BO	0.53	1.16	0.79

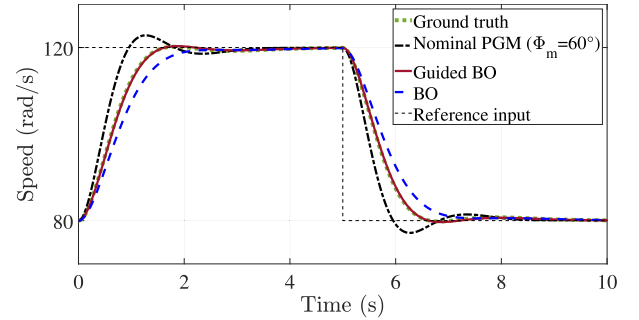


Fig. 8. Step response of the closed-loop numerical system with the speed controller tuned with different methods.

controller, and its tuned controller parameters are almost equal to the ground truth optimum gains.

Fig. 8 shows the step response of the closed-loop system with the guided BO controller on the real plant, corresponding to batch 4 and experiment 21 in Fig. 5. While BO step responses have over-damped behavior, the guided BO outperforms the other data-driven and nominal approaches. However, the nominal has larger overshoot resulting from its large proportional gain. The guided BO controller performance fits the ground truth performance with a slight overshoot, ITAE, and reasonably fast rise and settling time.

## VI. EXPERIMENTAL RESULTS

We demonstrate the performance of our proposed guided BO algorithm on two closed-loop real-time systems with different controller and system structures: direct current (DC) rotary motor and linear servomotor systems. The former system is time-variant, but the latter is time-invariant. We iteratively tune the speed controller's PI gains for the DC motor while we tune the position controller's PD gains for our servomotor system. At the end, each method's total controller tuning time is compared.

### A. Rotary Motor System

We have a DC rotary electrical motor setup presented in Fig. 9. This setup allows us to measure the system feedback in real time and modify the closed-loop system behavior by overwriting the controller gains. The controlled system comprises a DC rotary electrical motor with an angle encoder for speed measurements. We use a CompactPCI real-time engine PXI-7846R from National Instruments Corp to implement the PI speed controller. Given the reference input signal  $r$  and the PI controller gains  $K_p$  and  $K_i$  to the CompactPCI, a digital-to-analog data conversion with pulse-width modulation converts the digital speed control command to the analog voltage signal. Then, a current converter amplifies the voltage to its

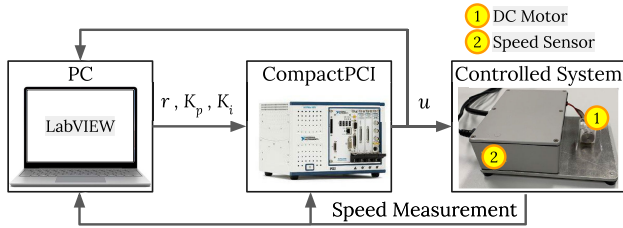


Fig. 9. Real-time DC rotary motor system structure for measurement and control.

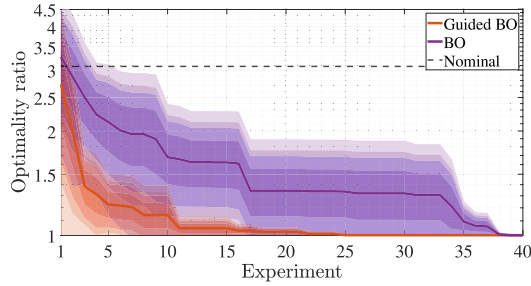


Fig. 10. Minimum observed optimality ratio up to each experiment on the DC rotary motor system. The thick line is the average over 100 batches, and the shaded area shows the 99%, 95%, 90%, and 68% confidence intervals.

proportional current  $u$  as an input to the DC motor. We use LabView software on our PC to interface with the hardware and implement the guided BO algorithm.

We want to tune the PI speed controller gains. We study the performance of our iterative tuning methods in the presence of unknown effects, such as friction and measurement noise. We focus on controlling the speed of the motor as it is relevant for its practical use. At the beginning of each experiment, we initialize the motor speed to reach a constant encoder angular speed equal to  $r_1 = 245$  rad/sec. This allows us to remove the uncertainty of starting from an arbitrary initial condition on this hardware. We build the initial data set  $\mathcal{D}$  with one gain pair ( $N_0 = 1$ ) and the respective measured objective according to (23). With a sampling rate of 20 Hz, we measure  $\tilde{n} = 100$  pairs of control commands  $u$  and physical plant's output  $y$ . In the guided BO algorithm, we use `tfest` function in MATLAB [59] with a second-order model to identify the DT of the nonlinear DC motor system based on refined instrumental variable method [47]. By measuring five times the performance metrics in (23) with a randomly selected  $\theta \in \Theta$ , we estimate the additive noise variance  $\sigma_\epsilon = 0.03$ . Table III summarizes the guided BO parameter values.

We study the measured performance of 100 experiment batches. Each batch starts with a different  $\Theta_{\text{init}}$  randomly selected inside the feasible set  $\Theta$  in (16), which is identical for both guided BO and BO methods. We define the optimum cost as the minimum cost measured among all batches of experiments. The optimality ratio is between the minimum observed cost in each batch and the optimum cost.

Fig. 10 demonstrates that guided BO converges on average with fewer experiments on the real DC rotary motor system and has a smaller variance between repetitions. Let's consider a threshold for the optimality ratio within 5% of the optimal cost. BO requires, on average, 38 experiments to converge to the optimality ratio of 1.05, whereas guided BO requires only

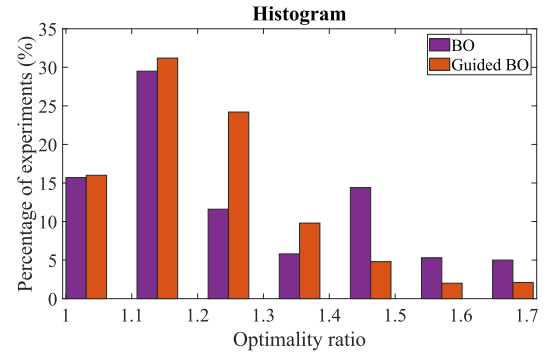


Fig. 11. Histogram of the experiments percentage vs. the optimality ratio of the measured performance among 100 batches of 40 experiments on the rotary motor system.

11 experiments to perform the same, a 71% improvement. Indeed, the best among the 100 batches of guided BO accomplishes this in 3 experiments (compared to 7 experiments required for BO, a 57% improvement), and the worst in 30 experiments (compared to 39 experiments required by BO, a 23% improvement). After 15 experiments, all 100 repetitions of guided BO attain optimality ratios better than 1.1, compared to 55% of the BO repetitions. In contrast, at this stage, the worst among the BO repetitions still has an optimality ratio of 2.85. The dashed black line in Fig. 10 depicts the cost value given the first nominal PGM controller designed based on the identified linear model of the system as described in Section IV-A. Both data-driven controller tuning approaches outperform the nominal GPM method in terms of the cost value at the expense of the number of iterations needed.

From the histogram of all measured performance values on Fig. 11, it can be seen that BO on the actual system requires performance measurements further away from the optimum, whereas the DT carries out the exploration allowing guided BO to test the controller gains on the real system at regions with smaller costs. Therefore, guided BO exploits more around the optimum cost, whereas BO explores further away from the optimum.

### B. Linear Servo Motor System

Our linear motor setup has a DM01 linear module from NTI AG company, displayed in Fig. 12. This module consists of a linear guide with an integrated LinMot P01 linear permanently actuated servo motor. We integrate a MS01-1/D-SSI non-contacting external position sensor with our module. This sensor has 5  $\mu\text{m}$  absolute resolution and 0.005 mm repeatability error. We build a unit feedback closed-loop system with a C1250-IP-XC-1S-C00 NTI AG axis controller and our external sensor to control the output position of the LinMot P01 linear motor with a sampling frequency of 1 kHz. This industrial position controller is a PDT1 controller implemented in C and defined as

$$C(s) = K_p + K_d \frac{s}{0.001s + 1}, \quad (24)$$

where  $K_p$  and  $K_d$  are proportional and derivative gains, respectively. This controller commands the linear motor plant as a current signal. A saturation mechanism clips the control signal once it exceeds the range  $[-7A, 7A]$ . We use OPC Unified Architecture in MATLAB to communicate with the



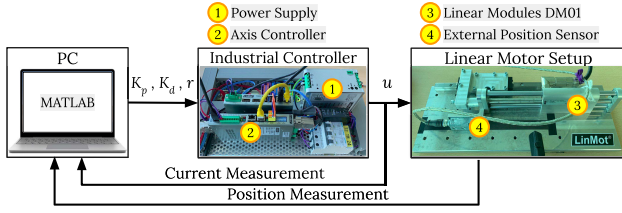


Fig. 12. LinMot P01 linear servo motor closed-loop control system and data-driven controller tuning framework.

hardware for the input and output measurements and controller parameter tuning. Given the current and linear motor position measurements, our algorithm sets the reference output  $r$  with certain industrial controller gains  $K_p$  and  $K_d$ .

We tune  $K_p$  and  $K_d$  gains of the position controller. To determine the RMSE threshold  $\eta_2$  in (12b), we estimate the noise standard deviation by measuring the performance metrics in 5 experiments with fixed identical controller gains. We normalize the standard deviation of each metric in  $f = [\zeta, T_s, T_r, e_{ITAE}]$  according to their mean measured value to retrieve the inverse of the signal-to-noise ratio (1/SNR) equal to  $[0.0001, 0.004, 0.002, 0.001]$ . Assuming weighted aggregation-based cost function in (23), we estimate the additive noise standard deviation as  $\sigma_\epsilon = 0.001$ . We choose the RMSE threshold  $\eta_2 = 3\sigma_\epsilon = 0.003$  to accept DT with sufficient fidelity considering the estimated noise level.

We assume that our linear motor generates linear motion for precise and automatic screw-driving applications. The speed of the step response is crucial, which can be represented by the rise time performance metric. One could also expect to avoid overshooting, whereas a slightly longer settling time could be permissible. We assume the ISO metric thread type M8.1 according to ISO 965 standard for the metric screw thread tolerances [60]. This thread type has a Pitch dimension tolerance of 0.112 mm. Therefore, considering our position controller's reference step height of 10 mm, the maximum overshoot must not exceed  $\frac{0.112}{10} \times 100 = 1.12\%$ .

We define the feasible set using a simplified and approximated model of the linear motor system retrieved from the system specifications provided by the LinMot company datasheet [61] where the phase margin is larger than 20 degrees. We thus redefine the feasible set for the control parameters as

$$\Theta = \left\{ [K_p, K_d] \mid K_p \in [K_{p_{\min}}, K_{p_{\max}}], K_d \in [K_{d_{\min}}, K_{d_{\max}}] \right\}, \quad (25)$$

where the boundaries of each control parameter  $K_{p_{\min}}, K_{p_{\max}}, K_{d_{\min}}, K_{d_{\max}}$  are equal to 5123.8, 6136.2, 40.1, 51.0, respectively.

We approximate the order of magnitude for each performance metric to define our overall performance function. We first measure the system's step response given four pairs of control gains in the rectangular feasible set vertices:  $(K_{p_{\min}}, K_{d_{\min}})$ ,  $(K_{p_{\min}}, K_{d_{\max}})$ ,  $(K_{p_{\max}}, K_{d_{\min}})$ , and  $(K_{p_{\max}}, K_{d_{\max}})$ . Considering SNR values, we also assume a priori relative importance weight to the metrics suitable for the screw driving application. Namely, we put 10 and 5 percent more

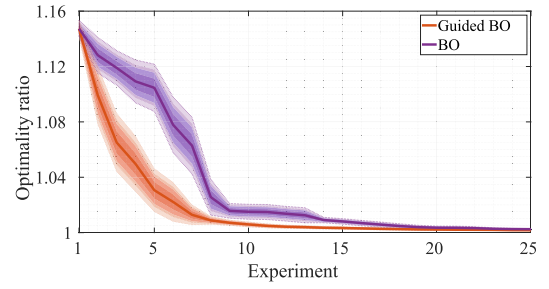


Fig. 13. Minimum observed optimality ratio on the linear motor system up to each experiment on the linear servo motor system, including 100 batches of 25 experiments where only the iterations on the real system are counted. The shaded area shows the 99%, 95%, 90%, and 68% confidence intervals. The thick line is the average of the batches.

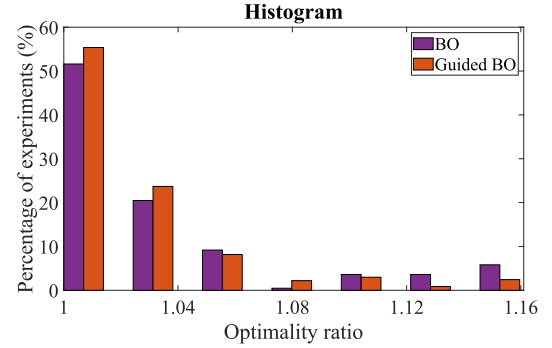


Fig. 14. Percentage of experiments on the real system vs. the optimality ratio for 100 batches in a fixed number (25) BO iterations.

weight on the maximum overshoot and rise time metrics. So the cost function weights  $w_1, w_2, w_3, w_4$  are equal to 0.10, 0.18, 0.69, 0.04, respectively, where  $\sum_{i=1}^4 w_i = 1$ .

We compare the performance of our proposed Guided BO with BO in 100 batches of experiments. Each batch consists of initial data with  $N_0$  and 25 performance measurements on the real linear motor system, whereas Guided BO internally may estimate the cost function with its DT model. To build the initial data set  $\mathcal{D}$  for the batches, we randomly select 100 pair of gains  $\theta = [K_p, K_d] \in \Theta$ . We use the same initial data set  $\mathcal{D}$  for each batch in both BO and guided BO methods. In guided BO, each time that we estimate the performance with an updated DT model  $\tilde{G}$ , we remove previous  $(\hat{\theta}, \tilde{\xi})$  data from the training data set  $\mathcal{D}$ . We perform an exhaustive grid search in  $\Theta$  to calculate the ground truth optimum performance  $J(\theta^*)$  required to calculate the optimality ratios.

Results on Fig. 13 prove that guided BO, on average, requires fewer experiments on the linear motor system and has a smaller variance between repetitions. Let's put a threshold on the optimality ratio equal to 1.01. BO requires, on average, 18 experiments on the real plant, excluding the initial data set experiment, to converge to the optimality ratio of 1.01.; in contrast, guided BO requires only 10 experiments to perform the same, a 44% improvement. Indeed, the best among the 100 batches of guided BO accomplishes this in 2 experiments (compared to 6 experiments required for BO, a 66% improvement), and the worst in 17 experiments, compared to 24 experiments required by BO, a 29% improvement). After 10 experiments, 86% of guided BO iterations attain optimality ratios better than 1.01, compared to only 14% of the BO repetitions.



TABLE VII  
CONTROLLER PARAMETERS FOR THE LINEAR MOTOR SETUP AT  
ITERATION  $i = 13$  OF A SINGLE BATCH OF  
EXPERIMENTS IN FIG. 13

Controller	$K_p$	$K_d$
Grid search (ground truth)	5341.25	49.68
Guided BO	5138.75	49.31
BO	6128.75	40.81

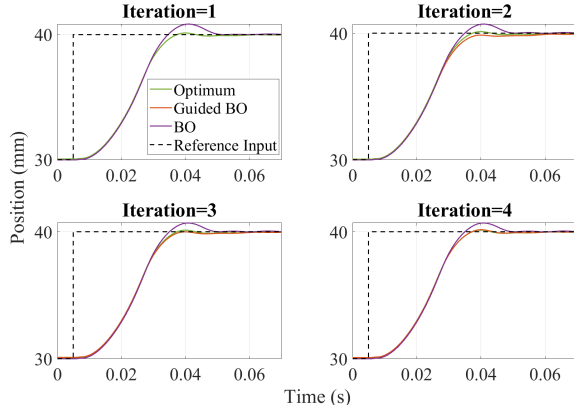


Fig. 15. Optimum controller performance at different iterations on the real system (linear motor closed-loop step response).

Furthermore, the confidence interval of the guided BO shrinks faster than BO and both eventually shrink inside the noise boundary. Looking at the corresponding histogram of all measured performance on the actual system in Fig. 14, guided BO does not explore the controller parameters with high-cost values due to the guidance of its DT. However, BO alone requires further exploration, eventually leading to its slower tuning performance.

We summarize the optimum controller gains of guided BO, standard BO, and the ground truth in Table VII. The guided BO controller's  $K_d$  is higher than the one obtained with BO, in line with the step responses shown in Fig. 15, causing smaller tracking error rate and a minor overshoot. The standard BO method has a larger  $K_p$  attempting to respond rapidly immediately after we change to a new reference signal. The optimum response has a maximum overshoot of 0.06%, lower than the required 1.12% by the metric thread standard.

Figure 15 visualizes the closed-loop response given the optimum controller gains of guided BO and BO methods after experiment  $i = 13$  of a batch of experiments. This figure indicates that the optimum step response tuned by guided BO converges faster than BO to the ground truth optimum response shown with the green line. We can rationalize the system's behavior based on the aggregated performance metrics. Recalling our cost in (23), we notice that the area under the curve of guided BO response is less than BO. While the settling and rise times are similar, we realize that the guided BO reduces the overshoot, resulting in a lower aggregate cost value closer to the optimum response.

### C. Controller Tuning Time

In each experiment on the physical systems, starting from an initial condition, we measure the step response during 10 s and 0.30 s for the rotary and linear motor systems, respectively. To decrease the noise level, we repeat it twice per experiment

TABLE VIII  
TIME REQUIRED BY EACH CONTROLLER TUNING METHOD

Method	Tuning Time [s]	
	Rotary system	Linear system
PGM	40.35	7.67
BO	785.19	16.13
Guided BO	503.54	14.23

for the DC motor setup and take the average estimated value per each performance metric.

Each batch of experiments consists of a single experiment conducted for the offline initial data set and specific experiments per each data-driven tuning, and on average, 121 experiments on DT in the guided BO method. The DT does not require a sophisticated model that would be time-consuming to calculate, and each BO iteration on the DT requires only 0.09 s of overdue computation time. So, the experiments on the physical system demand longer than the DT iterations.

For the PGM nominal controller, one needs to identify the plant model, which takes 7.66 s for the linear system and 40.35 s for the rotary system using the Labview-based real-time tool introduced in [51]. The PGM method takes 2 ms to calculate the controller parameters given the identified LTI model.

The results of the total tuning time are shown in Table VIII. Our guided BO is more time-efficient than the BO method because it requires fewer tedious experiments on the physical system and lower computational overdue by DT iterations.

## VII. CONCLUSION

In this paper, we guided the data-driven Bayesian optimization-based controller with a digital twin according to the uncertainty level of the GP model. The digital twin is approximated with available data during the system's operation without additional experiments. As long as the digital twin plant roughly captures the overall behavior of the system far away from the optimum region, it is going to be helpful to guide the Bayesian optimizer. We showed that the digital twin carries out the exploration duty of the optimizer whereas the exploitation is performed on the real system. We demonstrated that our guided BO approach considerably improves the controller tuning data efficiency, generalizes across the system and industrial controller structures, and converges faster to optimum values. Namely, we proved the guided BO's superior performance on a noisy linear servo motor and DC rotary motor real-time hardware.

A prospective extension of our method is to enhance it beyond motor systems to any complex plant with changing behavior. One can investigate how to develop a constrained version of the guided BO algorithm which can optimally update the feasible set while maintaining the plant's safety properties. Another future direction is to study estimation methods to obtain the DT plant model by further analyzing excitation signals or various closed-loop identification or regression methods. Our overall assessment can also be replaced with a multi-objective optimization using Pareto without emphasizing any single performance metric. Lastly,

online validation of the DT model can be integrated with our guided BO method to discard unnecessary information.

#### ACKNOWLEDGMENT

The authors would like to thank Matthias Geissmann for providing the LinMot equipment and useful discussions about its properties.

#### REFERENCES

- [1] R. P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar, "A review of PID control, tuning methods and applications," *Int. J. Dyn. Control*, vol. 9, no. 2, pp. 818–827, Jun. 2021.
- [2] G. Cavone, A. Bozza, R. Carli, and M. Dotoli, "MPC-based process control of deep drawing: An Industry 4.0 case study in automotive," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 1586–1598, Jul. 2022.
- [3] K. Astrom and R. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ, USA: Princeton Univ. Press, 2021.
- [4] Y. Tao, L. Li, H.-X. Li, and L. Zhu, "High-bandwidth tracking control of piezoactuated nanopositioning stages via active modal control," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 2998–3006, Oct. 2022.
- [5] R. E. J. Kennedy, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, Perth, WA, Australia, Nov. 1995, pp. 1942–1948.
- [6] Z. Qi, Q. Shi, and H. Zhang, "Tuning of digital PID controllers using particle swarm optimization algorithm for a CAN-based DC motor subject to stochastic delays," *IEEE Trans. Ind. Electron.*, vol. 67, no. 7, pp. 5637–5646, Jul. 2020.
- [7] K. Magkoutas, L. N. Rossato, M. Heim, and M. S. Daners, "Genetic algorithm-based optimization framework for control parameters of ventricular assist devices," *Biomed. Signal Process. Control*, vol. 85, Aug. 2023, Art. no. 104788.
- [8] M. J. Blondin, *Controller Tuning by Metaheuristics Optimization*. Cham, Switzerland: Springer, 2021, pp. 11–51.
- [9] N. Kameya, Y. Fujimoto, Y. Hosoyamada, and T. Suenaga, "Convolution-based data-driven simulation and controller design method," *IEEE Trans. Ind. Electron.*, vol. 71, no. 8, pp. 9541–9550, Aug. 2024.
- [10] N. Kameya, L. Bu, and Y. Fujimoto, "Application of data-driven simulation using CDDS approach to actuation systems," in *Proc. IEEE 18th Int. Conf. Adv. Motion Control (AMC)*, Feb. 2024, pp. 1–6.
- [11] A. Alanwar, Y. Stürz, and K. H. Johansson, "Robust data-driven predictive control using reachability analysis," *Eur. J. Control*, vol. 68, Nov. 2022, Art. no. 100666.
- [12] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," 2019, *arXiv:1904.12901*.
- [13] G. Schoettler, A. Nair, J. A. Ojea, S. Levine, and E. Solowjow, "Meta-reinforcement learning for robotic industrial insertion tasks," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Oct. 2020, pp. 9728–9735.
- [14] T. Chen, R. Lu, and Z. Ren, "Adaptive iterative learning control and inverse compensation of macro fiber composite system with hysteresis," *J. Sound Vibrat.*, vol. 569, Jan. 2024, Art. no. 118088.
- [15] Z. Afkhami, D. J. Hoelzle, and K. Barton, "Robust higher-order spatial iterative learning control for additive manufacturing systems," *IEEE Trans. Control Syst. Technol.*, vol. 31, no. 4, pp. 1692–1707, Jul. 2023.
- [16] M. Wang, K. Kang, C. Zhang, and L. Li, "Precise position control in air-bearing PMLSM system using an improved anticipatory fractional-order iterative learning control," *IEEE Trans. Ind. Electron.*, vol. 71, no. 6, pp. 6073–6083, Jun. 2024, doi: [10.1109/TIE.2023.3290251](https://doi.org/10.1109/TIE.2023.3290251).
- [17] D. Liao-McPherson, E. C. Balta, A. Rupenyan, and J. Lygeros, "On robustness in optimization-based constrained iterative learning control," *IEEE Control Syst. Lett.*, vol. 6, pp. 2846–2851, 2022.
- [18] Y. Jiang, S. Yin, K. Li, H. Luo, and O. Kaynak, "Industrial applications of digital twins," *Philos. Trans. Royal Soc. A*, vol. 379, no. 2207, 2021, Art. no. 20200360.
- [19] Y. Jiang, S. Yin, and O. Kaynak, "Performance supervised plant-wide process monitoring in Industry 4.0: A roadmap," *IEEE Open J. Ind. Electron. Soc.*, vol. 2, pp. 21–35, 2021.
- [20] W. Luo, T. Hu, Y. Ye, C. Zhang, and Y. Wei, "A hybrid predictive maintenance approach for CNC machine tool driven by digital twin," *Robot. Comput. Integr. Manuf.*, vol. 65, Oct. 2020, Art. no. 101974.
- [21] F. Dettù, S. Formentin, and S. M. Savaresi, "The twin-in-the-loop approach for vehicle dynamics control," *IEEE/ASME Trans. Mechatronics*, vol. 29, no. 2, pp. 1217–1228, Apr. 2024.
- [22] C. Cronrath and B. Lennartson, "How useful is learning in mitigating mismatch between digital twins and physical systems?" *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 1, pp. 758–770, Dec. 2004.
- [23] K. Wang, C. Mu, Z. Ni, and D. Liu, "Safe reinforcement learning and adaptive optimal control with applications to obstacle avoidance problem," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 3, pp. 4599–4612, Jul. 2024.
- [24] H. Han, S. Fu, H. Sun, and J. Qiao, "Data-driven multimodel predictive control for multirate sampled-data nonlinear systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 20, no. 3, pp. 2182–2194, Sep. 2004.
- [25] P. Zhou, S. Zhang, L. Wen, J. Fu, T. Chai, and H. Wang, "Kalman filter-based data-driven robust model-free adaptive predictive control of a complicated industrial process," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 2, pp. 788–803, Apr. 2022.
- [26] A. Rupenyan, M. Khosravi, and J. Lygeros, "Performance-based trajectory optimization for path following control using Bayesian optimization," in *Proc. 60th IEEE Conf. Decis. Control (CDC)*, Dec. 2021, pp. 2116–2121.
- [27] R. Garnett, *Bayesian Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2023.
- [28] A. Dey and N. Yodo, "A systematic survey of FDM process parameter optimization and their influence on part characteristics," *J. Manuf. Mater. Process.*, vol. 3, no. 3, p. 64, Jul. 2019.
- [29] M. Khosravi, A. Eichler, N. Schmid, R. S. Smith, and P. Heer, "Controller tuning by Bayesian optimization an application to a heat pump," in *Proc. 18th Eur. Control Conf. (ECC)*, Jun. 2019, pp. 1467–1472.
- [30] X. Guidetti, A. Rupenyan, L. Fassi, M. Nabavi, and J. Lygeros, "Plasma spray process parameters configuration using sample-efficient batch Bayesian optimization," in *Proc. IEEE 17th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2021, pp. 31–38.
- [31] X. Guidetti, A. Rupenyan, L. Fassi, M. Nabavi, and J. Lygeros, "Advanced manufacturing configuration by sample-efficient batch Bayesian optimization," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 11886–11893, Oct. 2022.
- [32] Y. Tao, J. Li, G. Gao, Z. Liu, and S. Rinderknecht, "Goal-oriented data-driven control for a holistic thermal management system of an electric vehicle," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 3, pp. 4893–4904, Jul. 2024.
- [33] M. Turchetta, A. Krause, and S. Trimpe, "Robust model-free reinforcement learning with multi-objective Bayesian optimization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 10702–10708.
- [34] P. Ngatchou, A. Zarei, and A. El-Sharkawi, "Pareto multi objective optimization," in *Proc. 13th Int. Conf. Intell. Syst. Appl. Power Syst.*, Nov. 2005, pp. 84–91.
- [35] M. Khosravi, C. König, M. Maier, R. S. Smith, J. Lygeros, and A. Rupenyan, "Safety-aware cascade controller tuning using constrained Bayesian optimization," *IEEE Trans. Ind. Electron.*, vol. 70, no. 2, pp. 2128–2138, Feb. 2023.
- [36] J. A. van Niekerk, J. D. le Roux, and I. K. Craig, "On-line automatic controller tuning of a multivariable grinding mill circuit using Bayesian optimisation," *J. Process Control*, vol. 128, Aug. 2023, Art. no. 103008.
- [37] W. Xu, C. N. Jones, B. Svetozarevic, C. R. Laughman, and A. Chakrabarty, "VABO: Violation-aware Bayesian optimization for closed-loop control performance optimization with unmodeled constraints," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2022, pp. 5288–5293.
- [38] X. Wang, Y. Jin, S. Schmitt, and M. Olhofer, "Recent advances in Bayesian optimization," *ACM Comput. Surv.*, vol. 55, no. 13s, pp. 1–36, Dec. 2023.
- [39] M. Maier, A. Rupenyan, C. Bobst, and K. Wegener, "Self-optimizing grinding machines using Gaussian process models and constrained Bayesian optimization," *Int. J. Adv. Manuf. Technol.*, vol. 108, nos. 1–2, pp. 539–552, May 2020.
- [40] C. E. Rasmussen, *Gaussian Processes in Machine Learning*. Berlin, Germany: Springer, 2004, pp. 63–71.
- [41] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 42, no. 1, p. 55, Feb. 2000.
- [42] M. A. Gelbart, J. Snoek, and R. P. Adams, "Bayesian optimization with unknown constraints," 2014, *arXiv:1403.5607*.
- [43] A. D. Bull, "Convergence rates of efficient global optimization algorithms," *J. Mach. Learn. Res.*, vol. 12, pp. 2879–2904, Jan. 2011.
- [44] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. Global Optim.*, vol. 13, no. 4, pp. 455–492, 1998.
- [45] P. I. Frazier, "Bayesian optimization," in *Recent Advances in Optimization and Modeling of Contemporary Problems*. Catonsville, MD, USA: Informa, 2018, pp. 255–278.
- [46] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, "Bayesian optimization with inequality constraints," in *Proc. ICML*, 2014, pp. 937–945.

- [47] M. Gilson, "What has instrumental variable method to offer for system identification?" *IFAC-PapersOnLine*, vol. 48, no. 1, pp. 354–359, 2015.
- [48] B. Sukhija, M. Turchetta, D. Lindner, A. Krause, S. Trimpe, and D. Baumann, "GoSafeOpt: Scalable safe exploration for global optimization of dynamical systems," *Artif. Intell.*, vol. 320, Jul. 2023, Art. no. 103922.
- [49] M. Zagorowska, E. C. Balta, V. Behrunani, A. Rupenyan, and J. Lygeros, "Efficient sample selection for safe learning," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 10107–10112, 2023.
- [50] M. Khosravi, V. N. Behrunani, P. Myszkowski, R. S. Smith, A. Rupenyan, and J. Lygeros, "Performance-driven cascade controller tuning with Bayesian optimization," *IEEE Trans. Ind. Electron.*, vol. 69, no. 1, pp. 1032–1042, Jan. 2022.
- [51] J. P. Keller, "Interactive control system design," *Control Eng. Pract.*, vol. 14, no. 2, pp. 177–184, Feb. 2006.
- [52] K. Ogata, *Modern Control Engineering* (Instrumentation and Controls Series). Upper Saddle River, NJ, USA: Prentice-Hall, 2010.
- [53] M. S. Fadali and A. Visioli, "Stability of digital control systems," in *Digital Control Engineering*, M. S. Fadali and A. Visioli, Eds., Boston, MA, USA: Academic, 2009, pp. 87–121.
- [54] W. K. Ho, C. C. Hang, and L. S. Cao, "Tuning of PID controllers based on gain and phase margin specifications," *Automatica*, vol. 31, no. 3, pp. 497–502, Mar. 1995.
- [55] E. Samur, *Performance Metrics for Haptic Interfaces*. Cham, Switzerland: Springer, 2012.
- [56] T.-T. Tay, I. Mareels, and J. B. Moore, *Off-Line Controller Design*. Boston, MA, USA: Birkhäuser, 1998, pp. 91–126.
- [57] D. Maiti, A. Acharya, M. Chakraborty, A. Konar, and R. Janarthanan, "Tuning PID and  $PI^{\lambda}D^{\delta}$  controllers using the integral time absolute error criterion," in *Proc. 4th Int. Conf. Inf. Autom. Sustainability*, Dec. 2008, pp. 457–462.
- [58] *Erweiterbarer, Umweltfreundlicher, Leistungsfähiger ETH-Rechner (Euler)*. Accessed: May 23, 2023. [Online]. Available: <https://scicomp.ethz.ch/wiki/Euler>
- [59] *MATLAB Version: 23.2.0.2380103 (r2023b) Update 1*, MathWorks, Inc, Natick, MA, USA, 2023.
- [60] *Standards Compendium 2022*, SNV, Swissmem, Swiss Association for Standardization (SNV), Sulzerallee, Switzerland, 2022.
- [61] *Linmot Linear Modules DM01*. Accessed: Feb. 28, 2024. [Online]. Available: <https://linmot.com/products/linear-guides-linear-modules/linear-modules-dm01/>



**Mahdi Nobar** received the master's degree in mechanical engineering with a minor in robotics from EPFL, Lausanne, Switzerland. He is currently pursuing the Ph.D. degree with the Automatic Control Laboratory, ETH Zürich, Zürich, Switzerland. He is also a Research Assistant with the Institut für Automation, University of Applied Sciences Northwestern Switzerland. Before that, he was a Researcher with the Robot Learning and Interaction Group, Idiap Research Institute. He is passionate about solving practical problems where he could

provide intelligence to autonomous systems. His research interests include data science, optimization, control theories, dynamical systems, and artificial intelligence.



**Jürg Keller** (Senior Member, IEEE) received the Engineering and Ph.D. degrees from ETH Zürich in 1983 and 1989, respectively. After his Post-Doctoral work with Australian National University (ANU), Canberra, he was with the Automation of Pharmaceutical Plants Department, Hoffman-La Roche, Basel, in 1991. His main task was to design modular and testable automation software. In 1995, he joined the startup of the University of Applied Sciences in Oensingen/Solothurn, where he had the opportunity to design a bachelor's program in automation and electronics and the automation master's program, in 2002. From 2005 to 2018, he was the President of Schweizerische Gesellschaft für Automatiken (SGA), the IFAC Member Organization of Switzerland. He is currently a Professor of control, machine vision, and machine learning with FHNW, University of Applied Sciences Northwestern Switzerland. His research interests include applications of modern control methods in industrial environments.



**Alisa Rupenyan** (Member, IEEE) received the B.Sc. degree in engineering physics and the M.Sc. degree in laser physics from the University of Sofia, Sofia, Bulgaria, in 2004 and 2005, respectively, and the Ph.D. degree from the Department of Physics and Astronomy, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands. She holds the Rieter endowed professorship of industrial AI with the ZHAW Centre for AI, Zurich University of Applied Sciences, Switzerland. From 2011 to 2014, she was a Post-Doctoral Fellow with ETH Zürich, Zürich, Switzerland, and from 2014 to 2018, she was a Lead Scientist in a robotic start-up. She was the Group Leader of Automation with Inspire AG, the technology transfer unit at ETH Zürich, and a Senior Scientist and the PI of the Automatic Control Laboratory, ETH Zürich, from 2018 to 2023. Her research interests include the intersection between machine learning, control, and optimization for industrial applications and robotics, especially in Bayesian methods applied for optimization and control, as well as learning-based control. She is an expert of the Swiss Innovation Agency (Innosuisse), a member of several technical committees at IEEE-CSS, IEEE-RAS, and IEEE-IES, and a member of the Executive Committee at the IFAC Industry Committee.



**Mohammad Khosravi** received the B.Sc. degree in electrical engineering and the B.Sc. degree in mathematical sciences from the Sharif University of Technology, Tehran, Iran, in 2011, the Postgraduate Diploma degree in mathematics from ICTP, Trieste, Italy, in 2012, the M.A.Sc. degree in electrical and computer engineering from Concordia University, Montreal, Canada, in 2016, and the Ph.D. degree from Swiss Federal Institute of Technology (ETH), Zürich, in 2022. He was a Junior Research Scientist with the Mathematical Biology Group, Institute for Research in Fundamental Sciences, Iran, from 2012 to 2014. He is currently an Assistant Professor with Delft Center for Systems and Control (DCSC), Delft University of Technology, The Netherlands. His research interests include data-driven and learning-based methods in modeling, model reduction, optimization, and control of dynamical systems and their applications in thermodynamics, buildings, energy, industry, and power systems. He has won several awards, including the ETH Medal, European Systems and Control Ph.D. Award, the Outstanding Student Paper Award in CDC 2020, the Outstanding Reviewer Award for IEEE Journal of Control Systems Letters, and the Gold Medal of the National Mathematics Olympiad.



**John Lygeros** (Fellow, IEEE) received the B.Eng. and M.Sc. degrees from Imperial College, London, U.K., in 1990 and 1991, respectively, and the Ph.D. degree from the University of California, Berkeley (UC Berkeley), in 1996. After research appointments with MIT, UC Berkeley, and SRI International, he joined the University of Cambridge, in 2000, as a Lecturer. From March 2003 to July 2006, he was an Assistant Professor with the Department of Electrical and Computer Engineering, University of Patras, Greece. In July 2006, he joined the Automatic Control Laboratory, ETH Zurich, where he is currently a Professor of computation and control and the Head of the Laboratory. His research interests include modeling, analyzing, and controlling large-scale systems with applications to biochemical networks, energy systems, transportation, and industrial processes. He is a member of IET and the Technical Chamber of Greece. Since 2013, he has been serving as the Vice-President Finances and a Council Member for the International Federation of Automatic Control, and the Director of the National Center of Competence in Research Dependable Ubiquitous Automation (NCCR Automation), since 2020.