



Bridging Reinforcement Learning and Iterative Learning Control: Autonomous Motion Learning for Unknown, Nonlinear Dynamics

Michael Meindl^{1,2*}, Dustin Lehmann³ and Thomas Seel²

¹Embedded Mechatronics Laboratory, Hochschule Karlsruhe, Karlsruhe, Germany, ²Department Artificial Intelligence in Biomedical Engineering, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany, ³Control Systems Group, Technische Universität Berlin, Berlin, Germany

This work addresses the problem of reference tracking in autonomously learning robots with unknown, nonlinear dynamics. Existing solutions require model information or extensive parameter tuning, and have rarely been validated in real-world experiments. We propose a learning control scheme that learns to approximate the unknown dynamics by a Gaussian Process (GP), which is used to optimize and apply a feedforward control input on each trial. Unlike existing approaches, the proposed method neither requires knowledge of the system states and their dynamics nor knowledge of an effective feedback control structure. All algorithm parameters are chosen automatically, i.e. the learning method works plug and play. The proposed method is validated in extensive simulations and real-world experiments. In contrast to most existing work, we study learning dynamics for more than one motion task as well as the robustness of performance across a large range of learning parameters. The method's plug and play applicability is demonstrated by experiments with a balancing robot, in which the proposed method rapidly learns to track the desired output. Due to its model-agnostic and plug and play properties, the proposed method is expected to have high potential for application to a large class of reference tracking problems in systems with unknown, nonlinear dynamics.

Keywords: autonomous systems, Gaussian processes (GP), iterative learning control, nonlinear systems, reinforcement learning, robot learning

1 INTRODUCTION

Recent developments in robotic technology remarkably contribute to the quality of human live: Hazardous tasks on rescue missions are handled by mobile robots that rifle through wreckage to locate people in need of help (Murphy, 2004). Advances in medical robotics strive for minimizing complications during surgery (Coulson et al., 2008). And the combination of exoskeletons and control algorithms aims for a future in which people struck by disability can walk again (Harib et al., 2018). The way to such accomplishments is paved by control techniques that enable robots to precisely perform agile and dynamic motions.

For example, model predictive control can achieve accurate motion if a precise model of the dynamics is available (Apgar et al., 2018; Hehn and D'Andrea, 2012; Feng et al., 2014). Requirements regarding the model's precision can be relaxed by robust or adaptive control techniques if the uncertainties comply with preset assumptions (Dong and Kuhnert, 2005; Dydek et al., 2013; Golovin

OPEN ACCESS

Edited by:

Bojan Nemeč,
Institut Jožef Stefan (IJS), Slovenia

Reviewed by:

Miha Deniša,
Institut Jožef Stefan (IJS), Slovenia
João Silvério,
German Aerospace Center (DLR),
Germany

*Correspondence:

Michael Meindl
meindlmichael@web.de

Specialty section:

This article was submitted to
Robot Learning and Evolution,
a section of the journal
Frontiers in Robotics and AI

Received: 12 October 2021

Accepted: 20 May 2022

Published: 12 July 2022

Citation:

Meindl M, Lehmann D and Seel T
(2022) Bridging Reinforcement
Learning and Iterative Learning
Control: Autonomous Motion Learning
for Unknown, Nonlinear Dynamics.
Front. Robot. AI 9:793512.
doi: 10.3389/frobt.2022.793512

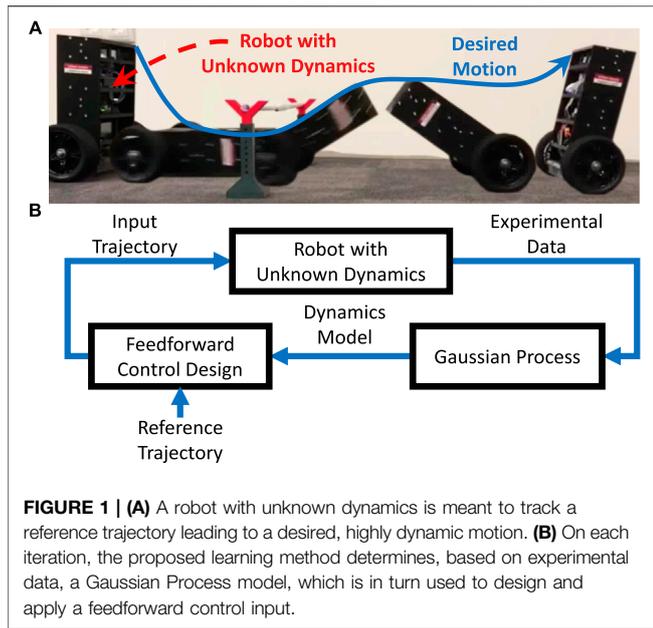
and Palis, 2019). Under similar conditions, Iterative Learning Control (ILC) can overcome model uncertainties and unknown disturbances by learning from errors of previous trials (Muller et al., 2012; Seel et al., 2016). However, all of these control approaches require system-specific prior knowledge to craft a suited model, controller, or learning configuration. In contrast, autonomy requires a methodology that self-reliantly learns a solution to the control problem without requiring any system-specific prior knowledge. In particular, Reinforcement Learning (RL) techniques have been employed to solve complex motion tasks without requiring any prior information. However, RL solutions typically suffer from two major drawbacks: First, the vast majority of the results were obtained in simulated environments (Heess et al., 2017; Tassa et al., 2018; Tsounis et al., 2020). Second, the few results obtained in real-world environments required at least multiple hours of learning, and the resulting controllers can be prone to failure (Schuitema, 2012; Kalashnikov et al., 2018; Ha et al., 2020; Zeng et al., 2020). The only exception from this statement is given by RL methods that exploit task-specific knowledge in the form of good initial policies and require 60–300 trials for local policy optimization (Kober and Peters, 2008; Peters and Schaal, 2008; Kormushev et al., 2013). A major breakthrough with respect to robustness and data-efficiency was achieved by hybrid techniques that learn parameter-free models, namely Gaussian Processes (GP), but also employ system-specific information such as knowledge of a state vector and an effective state feedback structure (Deisenroth, 2010). In the prominent example of *PILCO* (Deisenroth and Rasmussen, 2011a), experimental data are used to approximate the unknown dynamics by a GP, which is used to determine the optimal parameters of a state feedback controller. By this approach, an inverted pendulum on a cart could be swung up and stabilized in 12 s of system interaction (Deisenroth and Rasmussen, 2011a). However, in the context of autonomous motion learning, GP-based learning methods still suffer from two drawbacks: First, previously proposed methods only solve set-point stabilization tasks, which do not enable robots to perform challenging, dynamic maneuvers. These require reference tracking. Second, GP-based learning techniques still require system-specific prior knowledge such as the configuration of cost functions, a state vector that fully describes the system dynamics, and a control structure that is effective with respect to the problem at hand. Hence, the methods are not suited for plug and play learning of highly dynamic robotic motions.

The present contribution proposes a GP-based learning method for autonomously solving highly dynamic reference tracking tasks in systems with unknown, nonlinear, single-input/single-output dynamics. The proposed method autonomously determines all of its necessary parameters such that plug and play application becomes feasible. The method's capability to rapidly learn solutions to various reference tracking tasks while not requiring any system-specific prior knowledge is validated by extensive simulations and real-world experiments using a two-wheeled inverted pendulum robot, see **Figure 1**.

1.1 Related Work

Learning for control has been considered in a large body of literature that can be categorized by 1) the considered control problem respectively control strategy, 2) necessary system-specific prior knowledge, and 3) speed of learning. Reinforcement Learning (RL) techniques typically do not require any model and only few learning parameters such as step sizes or weights in cost functions. Furthermore, general RL approaches such as genetic algorithms (Moriarty and Miikkulainen, 2007) or policy gradient approaches (Peters and Schaal, 2006) can be applied to arbitrary control problems with unknown, nonlinear dynamics, but in turn require comparatively long periods of learning (Deisenroth and Rasmussen, 2011a). The speed of learning can be significantly increased if the technique is targeted towards a specific control problem and strategy such as stabilization by state feedback control, see e.g. (Lewis and Vrabie, 2009; Lewis and Vamvoudakis, 2011). A particularly data-efficient approach are so called model-based techniques that model the unknown, nonlinear dynamics by a GP, which is then used to design a state feedback controller (Deisenroth, 2011). Some successful applications to real-world examples are the control of a single inverted pendulum (Deisenroth and Rasmussen, 2011a), double inverted pendulum (Hesse et al., 2018), and robotic manipulator (Deisenroth et al., 2012). The concept of GP-based learning control has been further investigated in a variety of contributions. Stability of feedback-controlled GPs has been analyzed (Vinogradska et al., 2017, 2016), the problem of computational and data requirements has been investigated (Nguyen-Tuong and Peters, 2008; Capone et al., 2020), and solutions for safely improving an existing feedback controller have been proposed (Berkenkamp and Schoellig, 2015; Hewing et al., 2017; Umlauf et al., 2020). In a similar fashion, a type of lazy learning methods constructs locally weighted models based on experimental data to design feedback controllers for set-point stabilization, see (Atkeson et al., 1997), but require problem-specific knowledge like, e.g., the configuration of cost functions. While all of these works consider the challenging problem of efficiently learning control solutions for unknown, respectively uncertain, nonlinear dynamics, they have focused on the problem of set-point stabilization of systems, for which an effective feedback control structure is known. If the control tasks consists in performing a highly-dynamic motion, the achievable performance of time-domain feedback control is inherently limited by phenomena such as unknown delays, measurement noise, or non-minimum phase dynamics. To overcome the performance limitations of feedback control, a feedforward control component is required, see **Figure 2**.

In contrast to GP-based learning techniques, Iterative Learning Control (ILC) has focused on reference tracking tasks solved by feedforward control (Ahn et al., 2007; Arimoto et al., 1984). Model-based techniques like norm-optimal or \mathcal{H}_∞ ILC automatically determine the learning parameters, but require a model of the linear plant dynamics (Tayebi and Zaremba, 2002; Gunnarsson and Norrlöf, 2001; Amann et al., 1996). Model-free approaches like PD-ILC do not require a model but learning parameters that are typically tuned in experiment (Bristow et al., 2005). The concepts of PD-type (Shen et al., 2016) and norm-optimal (Lu et al., 2018) ILC have been extended to the case of nonlinear dynamics, but assume



the dynamics to be known. To relax requirements with respect to available model information, recent research has focused on so called data-driven ILC (DD-ILC) (Hou and Wang, 2013), which does not require a model of the plant. In the case of nonlinear, unknown dynamics, DD-ILC methods typically employ dynamic linearization of the plant dynamics and estimate the gradient of said linearization (Chi et al., 2015a,b; Ai et al., 2020). Alternatively, neural networks (NN) have been employed in DD-ILC to model the unknown dynamics (Ma et al., 2020; Yu et al., 2020). In a similar fashion, (Petric et al., 2018) propose basis functions for computing the input trajectory to track a desired reference, whereby the weights of the basis functions are learned by weighted linear regression, but the learning requires multiple hand-tuned parameters including the number of basis functions, step-sizes, and damping constants. In summary, while existing DD-ILC and similar methods can solve reference tracking tasks without requiring a plant model, some system-specific prior knowledge is required as, e.g., the signs of the dynamic linearization (Chi et al., 2015a,b; Ai et al., 2020), the layout of a suited neural network (Ma et al., 2020; Yu et al., 2020), or weights and step-sizes in update laws (Petric et al., 2018).

In summary, we conclude that reference tracking tasks in systems with unknown, nonlinear dynamics can be solved by DD-ILC methods, which, however, require system-specific prior knowledge such that autonomous plug and play application is generally not possible. In contrast, set-point stabilization problems can be solved by GP-based learning methods that assume comparatively little system-specific prior knowledge. However, in the context of reference tracking tasks, GP-based learning methods suffer from the inherent limitations of feedback control. To the best of our knowledge, there exists no learning method that autonomously solves reference tracking tasks for unknown, nonlinear systems, employs feedforward control to overcome the limitations of feedback control, and does not

require system-specific prior knowledge such that autonomous plug and play application is enabled.

1.2 Contributions

In this contribution, a GP-based ILC scheme is proposed that autonomously solves reference tracking tasks in systems with unknown, nonlinear, single-input/single-output dynamics. The proposed method includes a procedure to autonomously determine necessary parameters and enable plug and play application. Since the method directly models the input/output dynamics, only the output variable, instead of an entire state vector, has to be known and measured. To overcome the inherent limitations of feedback control, the proposed method employs feedforward control.

The proposed method is first validated by extensive simulations of a two-wheeled inverted pendulum robot (TWIPR), in which precise tracking is achieved after a small number of trials. Unlike existing approaches, the proposed method is not only verified for a single, well-chosen parameter configuration but for a wide range of parameter combinations such that robustness with respect to the autonomously determined parameters is ensured. In contrast to a variety of contributions, in which validation was restricted to simulated environments, the proposed method’s capability of solving real-world reference tracking tasks in a plug and play manor is validated by experiments on a TWIPR, see **Figure 1**.

2 PROBLEM FORMULATION

Consider an autonomous system that can repeatedly attempt a reference tracking task, as, e.g., a robot trying to perform a desired maneuver. We assume that the system’s output, e.g., a joint angle or position, can be influenced by an input signal, e.g., a motor torque, and that the relation of these variables is deterministic, causal, and time-invariant. However, we do *not* assume that a model of the dynamics is available and we *do* assume the general case of nonlinear dynamics.

Formally, consider a discrete-time, single-input, single-output, repetitive system with a finite trial duration of $N \in \mathbb{N}$ samples, and, on trial $j \in \mathbb{N}_{\geq 0}$ and sample $n \in [1, N]$, output variable $y_j(n) \in \mathbb{R}$, respectively input variable $u_j(n) \in \mathbb{R}$. The samples are collected in the so called output trajectory $\mathbf{y}_j \in \mathbb{R}^N$, respectively input trajectory $\mathbf{u}_j \in \mathbb{R}^N$, i.e., $\forall j \in \mathbb{N}_{\geq 0}$,

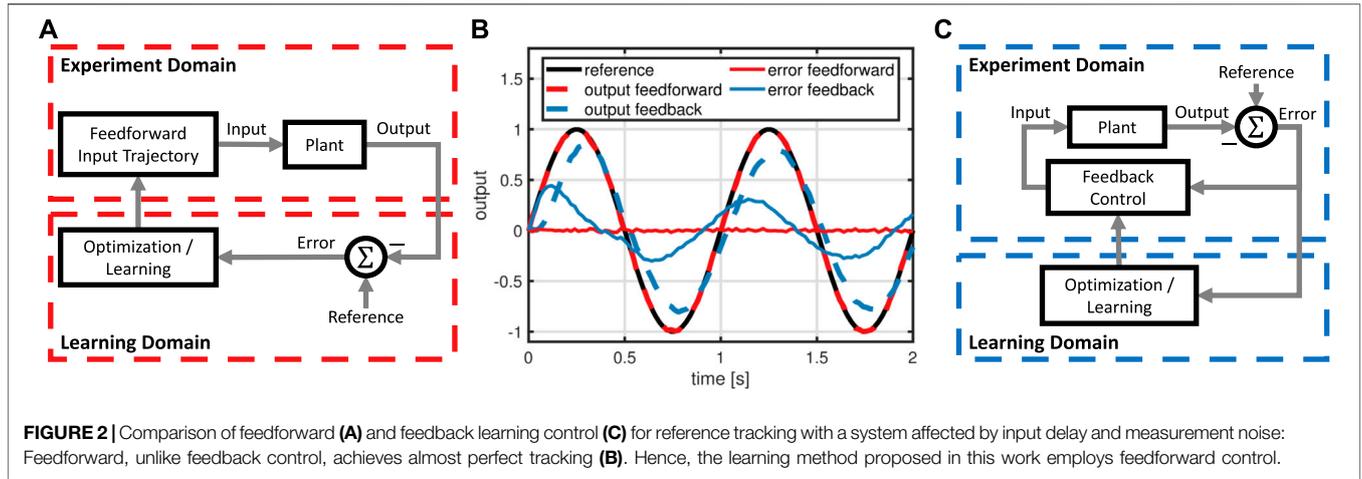
$$\mathbf{y}_j := [y_j(1) \ y_j(2) \ \dots \ y_j(N)]^T \tag{1}$$

$$\mathbf{u}_j := [u_j(1) \ u_j(2) \ \dots \ u_j(N)]^T. \tag{2}$$

Without loss of generality, the dynamics can be written in the lifted form

$$\forall j \in \mathbb{N}_{\geq 0}, \quad \mathbf{y}_j = \mathbf{p}(\mathbf{u}_j), \tag{3}$$

where \mathbf{p} is the unknown, trial-invariant, nonlinear dynamics. The task consists of updating the input \mathbf{u}_j from trial to trial such that the output \mathbf{y}_j converges to the desired reference trajectory $\mathbf{r} \in \mathbb{R}^N$. Tracking performance is measured by the error trajectory



$$\forall j \in \mathbb{N}_{\geq 0}, \quad \mathbf{e}_j := \mathbf{r} - \mathbf{y}_j \quad (4)$$

and root-mean-squared error (RMSE)

$$\forall j \in \mathbb{N}, \quad e_j^{\text{RMS}} := \sqrt{\sum_{i=1}^N \frac{[e_i]_j^2}{N}}. \quad (5)$$

The problem considered in this work consists in developing a learning method that updates the input trajectory on each trial such that the RMSE decreases. Learning performance is judged based on the progression of the RMSE through trials, and the RMSE shall decline quickly and monotonically. The learning method must not require any a priori model information on the plant dynamics. To support plug and play application, the method must autonomously determine necessary parameters. Furthermore, the method must provide a fair degree of robustness with respect to autonomously determined parameters.

3 PROPOSED LEARNING METHOD

We address the proposed problem by an iterative learning scheme, in which each iteration consists of three steps. First, a parameter-free model of the plant dynamics is identified using the experimental data of previous trials, see Section 3.1. To accommodate for possibly nonlinear dynamics, a generic GP

model is employed, which predicts the output trajectory for a given input trajectory. Second, the updated input trajectory is determined by solving an optimal feedforward control problem based on the GP model, see Section 3.2. Third, the updated input trajectory is applied to the plant and resulting data is in turn used to refine the GP model. The structure of the proposed learning scheme is depicted in Figure 3. To enable plug and play application, the proposed method autonomously determines necessary parameters, see Section 3.3.

3.1 Gaussian Process Model

We propose a Gaussian Process (GP) model, formally a function $m: \mathbb{R}^N \mapsto \mathbb{R}^N$, that predicts the plant's output trajectory $\hat{\mathbf{y}} \in \mathbb{R}^N$ based on an input trajectory $\mathbf{u} \in \mathbb{R}^N$, where the trial index is omitted for sake of notational simplicity.

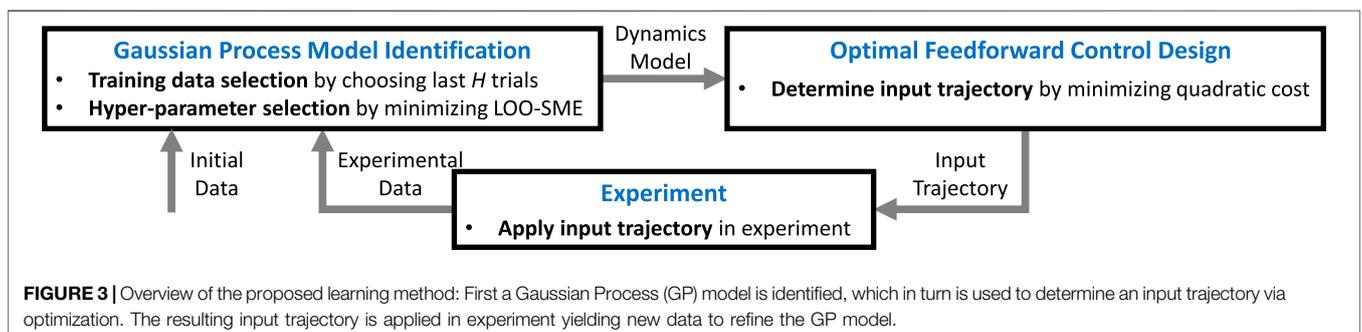
Let $f(\mathbf{v}): \mathbb{R}^D \mapsto \mathbb{R}$ denote the unknown target function that depends on the regression vector $\mathbf{v} \in \mathbb{R}^D$. Predictions are based on $K \in \mathbb{N}$ observations $z_k \in \mathbb{R}$ stemming from:

$$\forall k \in [1, K], \quad z_k = f(\mathbf{v}_k) + w_k \quad | \quad w_k \sim \mathcal{N}(0, \sigma_w^2). \quad (6)$$

The K observation pairs (z_k, \mathbf{v}_k) are collected in the observation training vector $\bar{\mathbf{z}} \in \mathbb{R}^K$ and regression training matrix $\bar{\mathbf{V}} \in \mathbb{R}^{D \times K}$, i.e.,

$$\bar{\mathbf{z}} := [z_1 \ z_2 \ \dots \ z_K]^T, \quad (7)$$

$$\bar{\mathbf{V}} := [\mathbf{v}_1 \ \mathbf{v}_2 \ \dots \ \mathbf{v}_K]. \quad (8)$$



The kernel function of two regression vectors $\mathbf{v} \in \mathbb{R}^D$ and $\hat{\mathbf{v}} \in \mathbb{R}^D$ is denoted by $k_{\mathbf{v}\hat{\mathbf{v}}} \in \mathbb{R}$. The kernel matrix of two regression matrices $\mathbf{V} \in \mathbb{R}^{D \times K}$, $\hat{\mathbf{V}} \in \mathbb{R}^{D \times \hat{K}}$, which are assembled according to (8), is denoted by $\mathbf{K}_{\mathbf{V}\hat{\mathbf{V}}} \in \mathbb{R}^{K \times \hat{K}}$ and has entries $[\mathbf{K}_{\mathbf{V}\hat{\mathbf{V}}}]_{ij} = k_{\mathbf{v}_i\hat{\mathbf{v}}_j}$.

Given $F \in \mathbb{N}$ test regression vectors assembled in the regression matrix $\mathbf{V} \in \mathbb{R}^{D \times F}$, the predicted mean $\boldsymbol{\mu} \in \mathbb{R}^F$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{F \times F}$ are given by:

$$\boldsymbol{\mu} = \mathbf{K}_{\mathbf{V}\hat{\mathbf{V}}} [\mathbf{K}_{\hat{\mathbf{V}}\hat{\mathbf{V}}} + \sigma_w^2 \mathbf{I}]^{-1} \hat{\mathbf{z}} \quad (9)$$

$$\boldsymbol{\Sigma} = \mathbf{K}_{\mathbf{V}\mathbf{V}} - \mathbf{K}_{\mathbf{V}\hat{\mathbf{V}}} [\mathbf{K}_{\hat{\mathbf{V}}\hat{\mathbf{V}}} + \sigma_w^2 \mathbf{I}]^{-1} \mathbf{K}_{\hat{\mathbf{V}}\mathbf{V}} \quad (10)$$

The general GP framework can be employed in different ways to model the unknown dynamics (Eq. 3), where the model characteristics are determined by the definition of observation variable z , regression vector \mathbf{v} , and kernel function k . First, we exploit the dynamics' time-invariance by employing a single GP for predicting each output sample. Hence, the observation variable and regression vector are time dependent, i.e., $\forall n \in [1, N]$, z_n, \mathbf{v}_n . Now, the model can be chosen to be one of three types, namely finite impulse response (FIR), infinite impulse response (IIR), or state space (SS), which are outlined in the following. A FIR model is obtained when the regression vector consists of the current and all previous input samples, i.e.,

$$\forall n \in [1, N], \quad \mathbf{v}_n := [u(n) \dots u(1) \dots 0 \dots 0]^T. \quad (11)$$

An IIR model is obtained when the regression vector consists of the current input and the $P \in \mathbb{N}$ previous output samples, i.e.,

$$\forall n \in [1, N], \quad \mathbf{v}_n := [u(n) \ y(n-1) \dots y(n-P) \ 0 \dots 0]^T. \quad (12)$$

A SS model is obtained when the regression vector consists of the current input and the previous state sample, which is denoted by $\mathbf{x} \in \mathbb{R}^O$, i.e.,

$$\forall n \in [1, N], \quad \mathbf{v}_n := [u(n) \ \mathbf{x}^T(n-1)]^T. \quad (13)$$

The SS model requires multiple GPs with each predicting the progression of a single state variable (Deisenroth and Rasmussen, 2011b), which not only increases computational complexity, but also requires measurements of the full state vector. Furthermore, IIR and SS models require so called *roll-out predictions*, meaning that the predictions of previous samples are required for predicting the current sample (Deisenroth and Rasmussen, 2011b), and, hence, the matrix $\mathbf{K}_{\mathbf{V}\hat{\mathbf{V}}}$ has to be recomputed for each sample in the output trajectory, which increases the model's complexity and computational demands. In contrast, the FIR model only requires a single prediction according to Eq. 9 and only a single computation of the matrix $\mathbf{K}_{\mathbf{V}\hat{\mathbf{V}}}$. We, hence, employ a FIR model, and the regression vector is defined according to Eq. 11 in order to reduce the computational demands of the learning method.

We further choose difference-predictions, i.e.,

$$\forall n \in [1, N], \quad z_n := y(n) - y(n-1) \quad | \quad y(-1) = 0, \quad (14)$$

which, compared to absolute predictions, increase the model's capability of extrapolation, see (Deisenroth and Rasmussen, 2011b).

As kernel function, we employ a squared-exponential kernel (SEK)

$$k(\mathbf{v}, \hat{\mathbf{v}}) = \exp\left(-\frac{1}{2l^2}(\mathbf{v} - \hat{\mathbf{v}})^T(\mathbf{v} - \hat{\mathbf{v}})\right), \quad (15)$$

where $l \in \mathbb{R}$ is a so called length scale.

Remark 1: SEKs allow a GP to model arbitrary target functions. In the context of dynamic systems, a SEK leads to a nonlinear, time-invariant (NTI) model. Using a squared kernel instead, as e.g.,

$$k_{\mathbf{v}\hat{\mathbf{v}}} := \mathbf{v}^T \hat{\mathbf{v}}, \quad (16)$$

results in a linear, time-invariant (LTI) model. If the plant dynamics are linear, one may employ a squared kernel to decrease computational complexity in comparison to a NTI model.

To predict an output trajectory $\hat{\mathbf{y}}$ for an arbitrary input trajectory \mathbf{u} , the latter is used to determine N regression vectors \mathbf{v}_n , $n \in [1, N]$, according to Eq. 11, which are assembled in a regression matrix \mathbf{V} according to Eq. 8. The predicted mean vector $\boldsymbol{\mu}$ follows from Eq. 9. By Eq. 14, $\boldsymbol{\mu}$ contains difference predictions such that the components of $\hat{\mathbf{y}}$ follow from the cumulative sum of $\boldsymbol{\mu}$, i.e.

$$\forall n \in [1, N], \quad [\hat{\mathbf{y}}]_n = \sum_{i=1}^n [\boldsymbol{\mu}]_i. \quad (17)$$

Mean and covariance predictions require the measurement variance σ_w^2 and length-scale l , which are so called hyper-parameters. Typically, hyper-parameters are determined based on training data, and numerous approaches have been detailed in the literature (Rasmussen and Williams, 2005). We propose selecting hyper-parameters by minimizing the so called leave-one-out squared-mean-error (LOO-SME). For each of the $k \in [1, K]$ available observations z_k , the remaining observation pairs are used to predict z_k . The LOO-SME follows from summing the squared difference between the K leave-one-out predictions and respective observations z_k . Formally, the k th LOO-prediction $\hat{\mu}_k$ is given by

$$\hat{\mu}_k = z_k - \frac{[\mathbf{K}_{\hat{\mathbf{V}}\hat{\mathbf{V}}} + \sigma_w^2 \mathbf{I}]^{-1} \hat{\mathbf{z}}_k}{[\mathbf{K}_{\hat{\mathbf{V}}\hat{\mathbf{V}}} + \sigma_w^2 \mathbf{I}]^{-1}}_{kk} \quad (18)$$

leading to the LOO-SME e_{LOO}

$$e_{\text{LOO}} = \sum_{k=1}^K (z_k - \hat{\mu}_k)^2 \quad (19)$$

and the hyper-parameters $\boldsymbol{\theta} := [\sigma_w^2 \ l]^T$ follow from

$$\boldsymbol{\theta} = \underset{\hat{\boldsymbol{\theta}}}{\text{argmin}} \ e_{\text{LOO}}. \quad (20)$$

The optimization problem (Eq. 20) can be solved efficiently, because analytic expressions of the gradients are available, see (Rasmussen and Williams, 2005).

Remark 2: Determining hyper-parameters by LOO-SME minimization is a rather uncommon choice because the

variance of the predictions is not taken into account, see (Rasmussen and Williams, 2005). However, we compared LOO-SME minimization with the state-of-the-art method *evidence maximization*, as described in Rasmussen and Williams (2005), and we found that performance is superior when using LOO-SME minimization. We assume that this is due to the proposed learning scheme solely relying on the GP's mean prediction.

GP predictions are known to become computational expensive with increasing amounts of training data (Snelson and Ghahramani, 2006). To overcome this limitation, various data selection approaches have been proposed to reduce training data to a tractable amount, see, e.g., (Seeger et al., 2003; Snelson and Ghahramani, 2006). In the present work, we simply propose limiting the training data to the last $H \in \mathbb{N}$ trials.

3.2 Optimal Feedforward Control

After the GP model has been identified, it is used to determine an input trajectory that leads to a smaller difference between reference and output trajectory than the input trajectories of previous trials. We propose an optimal control design, where the input is chosen to minimize a quadratic cost criterion. The latter not only considers the predicted tracking error, but also the change of the input trajectory to avoid model inversion and, hence, increase robustness with respect to the uncertainty of the current trial's model. Formally, the cost criterion is given by

$$\forall j \in \mathbb{N}_{\geq 0}, \quad J(\mathbf{u}_{j+1}) = q \|\mathbf{r} - \hat{\mathbf{y}}(\mathbf{u}_{j+1})\|_2^2 + s \|\mathbf{u}_{j+1} - \mathbf{u}_j\|_2^2, \quad (21)$$

where $q, s \in \mathbb{R}_{>0}$ are scalar weights. On each trial, the updated input trajectory \mathbf{u}_{j+1} is chosen to minimize the cost criterion, i.e.,

$$\forall j \in \mathbb{N}_{\geq 0}, \quad \mathbf{u}_{j+1} = \underset{\bar{\mathbf{u}}}{\operatorname{argmin}} J(\mathbf{u}_{j+1}) \quad (22)$$

The optimization problem (Eq. 22) can be solved efficiently since analytic expressions of the cost's gradient with respect to the input variable can be obtained (Deisenroth and Rasmussen, 2011a).

Remark 3: (Learning to Track Multiple Reference Trajectories). In this paper, we have only considered learning to track a single reference trajectory. This is particularly relevant to applications, in which a robotic system has to repetitively solve a single task, as for example in manufacturing. However, in some applications, a robot has to track a variety of different reference trajectories. In order to accelerate learning in such *multi-reference problems*, the proposed method can be extended to train a GP model based on all previous input/output trajectory pairs, which is then used to determine an optimal initial input trajectory \mathbf{u}_0 for any new reference.

3.3 Autonomous Parameterization

Ideally, autonomous learning methods should require neither a priori model information nor manual tuning of parameters. In contrary to previous contributions, the proposed method automatically determines necessary parameters by the

procedure outlined in this section, and, as a result, plug and play application is enabled, see **Figure 4**.

First, we consider the choice of the initial data set \mathcal{I} that is used to determine the first GP model and consists of $I \in \mathbb{N}$ trajectory pairs $(\mathbf{y}_i, \mathbf{u}_i)$, i.e.,

$$\mathcal{I} := \{(\mathbf{y}_i, \mathbf{u}_i) \mid i \in [1, I]\}. \quad (23)$$

For this purpose, we first determine the largest significant frequency f_O of the reference trajectory. The frequency f_O is used to design a zero-phase low-pass filter f_{LP} . The low-pass filter f_{LP} is applied to a zero mean normal distribution with covariance $\sigma_1^2 \mathbf{I}$, and the initial input trajectories are drawn from the resulting distribution, i.e.,

$$\forall i \in [1, I], \quad \mathbf{u}_i \sim f_{LP}(\mathcal{N}(\mathbf{0}, \sigma_1^2 \mathbf{I})). \quad (24)$$

The input variance σ_1^2 is iteratively increased until an input trajectory drawn according to **Eq. 24** leads to an output trajectory, whose maximum roughly equals the maximum of the reference, i.e.,

$$\mathbf{u} \sim f_{LP}(\mathcal{N}(\mathbf{0}, \sigma_1^2 \mathbf{I})) \quad \text{such that} \quad e_1 := \|\mathbf{r}\|_\infty - \|\mathbf{p}(\mathbf{u})\|_\infty \approx 0. \quad (25)$$

In autonomous parameterization, the number of initial trials is chosen as one, i.e., $I = 1$, to decrease the number of total trials required for the learning. Note that a larger number of initial trials reduces the variance of the convergence speed, i.e., in safety-critical applications a larger number of initial trials may be recommendable. However, using only one initial trial, the proposed method already provides a remarkably safe convergence, as will be demonstrated in **Section 4.3**. Hence, the autonomous parameterization employs $I = 1$.

Once the parameters f_O , σ_1^2 , and I have been determined, the initial trials are performed, and the weights q and s are chosen based on the experimental data. The scalar q , which weights the error trajectory, is without loss of generality chosen as unity, i.e.,

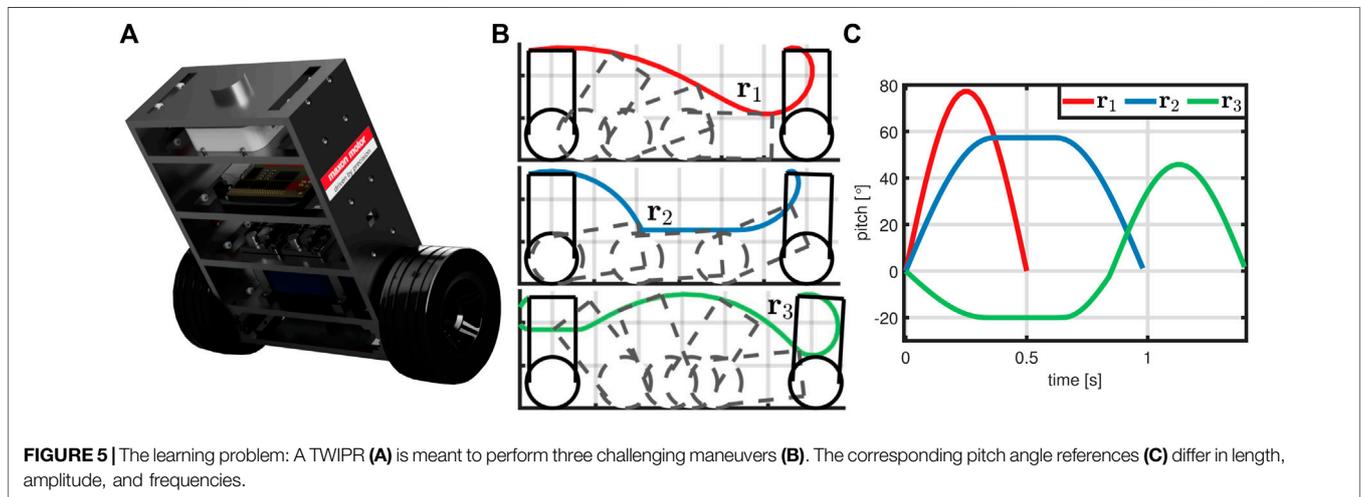
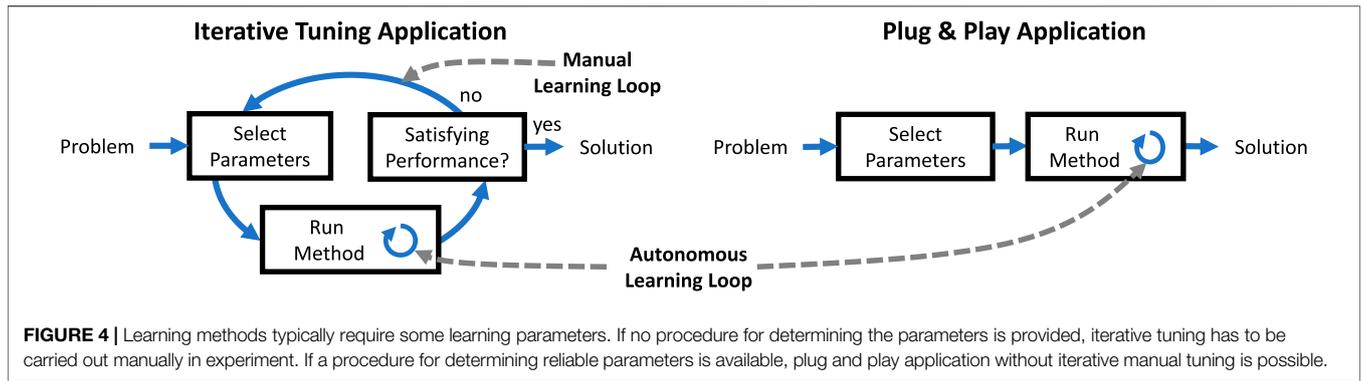
$$q = 1. \quad (26)$$

The scalar s , which weighs the change in input variable, is chosen as the average squared ratio of output to input maxima over the I initial trials, i.e.,

$$s = \frac{1}{I} \sum_{i=1}^I \frac{\|\mathbf{y}_i\|_\infty^2}{\|\mathbf{u}_i\|_\infty^2}. \quad (27)$$

The purpose of the weight selection in **Eqs 26–27** is to normalize the cost function (**Eq. 21**), i.e., we would like the weighted change in the input trajectory to have an impact on the cost function that is equal to impact to the weighted next-trial error trajectory. To achieve this normalization, we employ the squared ratio of the initial trials' input and output trajectories as described in **Eq. 27**. Note that the normalization of the cost function only depends on the ratio of the weights q and s but does *not* depend on their absolute values, hence the choice $q = 1$ is arbitrary and without loss of generality.

The procedure described in this section automatically determines all the necessary parameters without requiring any



a priori information on the plant. The following simulations are going to demonstrate that the automatically determined parameters lead to the desired learning performance and that the method provides a fair degree of robustness with respect to the automatically determined parameters.

Remark 4: Note that the proposed autonomous parameterization method aims at successful learning for unknown, nonlinear dynamics and different reference trajectories without requiring any manual adjustment of the parameters. Beyond this aspect, the method may be extended to automatically determine parameters that yield optimal performance in some to-be-defined sense.

4 VALIDATION BY SIMULATION

In this section, the proposed learning method is validated by simulation of a two-wheeled inverted pendulum robot (TWIPR) that is meant to perform challenging maneuvers, see Figure 5. The TWIPR and automatic determination of learning parameters are presented in Section 4.1. Afterwards, the learning performance for three representative references is investigated in Section 4.2, and the proposed method’s robustness with

respect to learning parameters is verified in Section 4.3. Lastly, the effect of the weight s on the learning characteristics is studied in Section 4.4.

4.1 The Learning Problem

Consider the TWIPR and three desired maneuvers depicted in Figure 5. The corresponding pitch angle reference trajectories are denoted by $\mathbf{r}_1 \in \mathbb{R}^{25}$, $\mathbf{r}_2 \in \mathbb{R}^{50}$, and $\mathbf{r}_3 \in \mathbb{R}^{71}$ and formal definitions are given in Supplementary Appendix S1.2. The robot consists of a pendulum body housing main electronics including a microcomputer, inertial measurement units, motors and accumulator. Wheels are mounted onto the motors such that the robot can drive while balancing its chassis. In order to validate the proposed method via simulations, a detailed, nonlinear model of the TWIPR dynamics, see Kim and Kwon, 2015, is implemented. However, the simulation model is completely unknown to the learning method, which can only interact with the simulation by applying an input trajectory and receiving the corresponding output trajectory. Only an approximate, linear model of the dynamics at the upright equilibrium has been obtained, which merely suffices to design a stabilizing feedback controller, see Supplementary Appendix S1.3. Due to the imprecise model, the feedback controller can not track the references precisely, and we instead employ the proposed learning method.

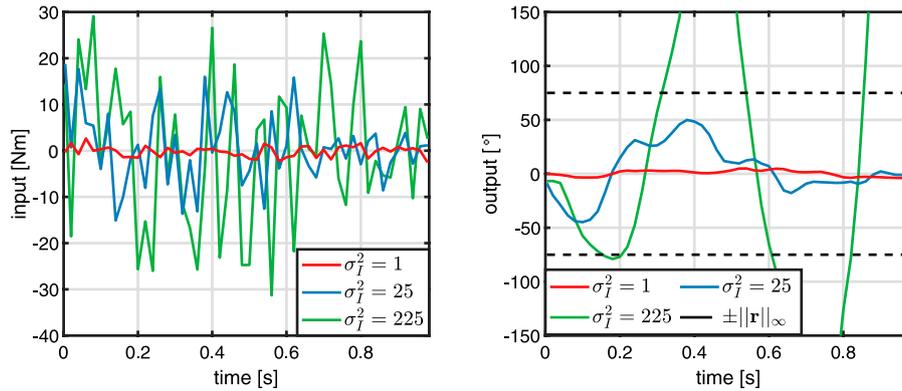


FIGURE 6 | Determination of the input variance σ_I^2 : Five different values, of which only three are presented, are used to draw random input trajectories that are applied to the plant. The input variance $\sigma_I^2 = 1$ hardly excites the system. In contrary, the input variance $\sigma_I^2 = 225$ leads to an output trajectory that significantly exceeds the reference's maximum. The input variance $\sigma_I^2 = 25$ is selected, because the corresponding output trajectory has the same order of magnitude as the reference.

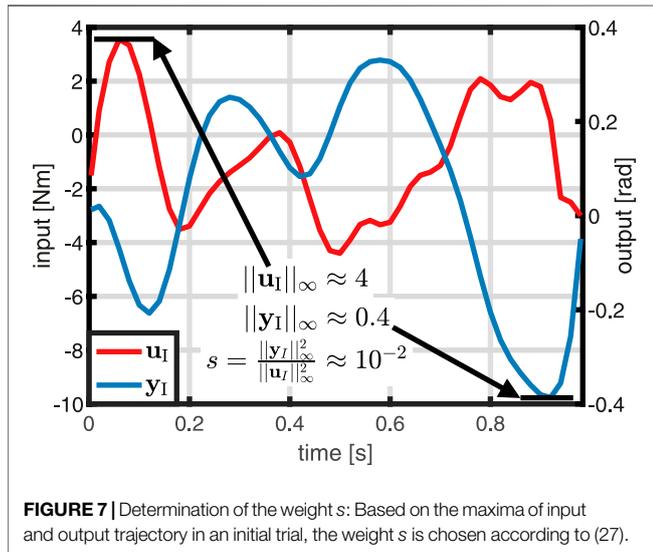


FIGURE 7 | Determination of the weight s : Based on the maxima of input and output trajectory in an initial trial, the weight s is chosen according to (27).

Instead of a state vector, the learning method only requires knowledge of the output variable, which is given by the pitch angle, i.e.,

$$\forall n \in \mathbb{N}_{\geq 0}, \quad y(n) := \Theta(n). \quad (28)$$

The input variable is given by the motor torque, $\forall n \in [1, N]$, $u_L(n) \in \mathbb{R}$.

Application of the proposed learning method requires learning parameters that are automatically determined by the procedure outlined in Section 3.3. We aim at tracking pitch trajectories with a maximum of approximately 75° and spectral content roughly below 5 Hz, i.e.,

$$\|\mathbf{r}\|_\infty \approx 75^\circ \quad f_0 \approx 5 \text{ Hz}. \quad (29)$$

Based on the frequency f_0 , a forward-backward, second order Butterworth filter f_{LP} is designed, which is used for drawing initial input trajectories, see Eq. 24. To determine the input variance σ_I^2 , the method automatically applies test input trajectories with

successively increasing amplitudes to the nonlinear system as described in Section 3.3. A coarse grid $\sigma_I^2 \in \{0.005, 0.05, 1, 25, 225\}$ is chosen, and the algorithm obtains $e_1 = [75, 75, 71, 20, -280]$ and thus selects $\sigma_I^2 = 25$, see Eq. 25 and Figure 6. Note that an even larger and finer grid as well as more sophisticated selection methods could be used, but this simple approach is sufficient because the proposed algorithm exhibits great robustness with respect to the choice of the input variance, as detailed in Section 4.3.

Next, the weights s and q of the cost function are determined. According to Eq. 26, $q = 1$ is selected. To determine the weight s , one initial trial with the previously determined input variance is performed. As detailed in Figure 7, the value of s directly results from Eq. 27 and the experimental data, i.e.,

$$s = \frac{\|y_I\|_\infty^2}{\|u_I\|_\infty^2} \approx \frac{0.4^2}{4^2} = 10^{-2}. \quad (30)$$

To demonstrate the data-efficiency of the proposed learning method, the training data are limited to the last five trials, i.e., $H = 5$.

4.2 Learning Performance

First, learning performance for the desired references $\mathbf{r}_1, \mathbf{r}_2$, and \mathbf{r}_3 is investigated. The parameters are chosen according to the previous section and only one initial trial $I = 1$ is used. In Figure 8, progressions of the output trajectories and error norms over the trials are depicted. For all three references, the proposed method achieves precise tracking after roughly 15 trials. The respective RMSEs rapidly decline over the first trials and converge to a small value close to zero. In case of the references \mathbf{r}_2 and \mathbf{r}_3 , the RMSE decreases monotonically. The simulations demonstrate that, by using the automatically determined parameters, the proposed method rapidly learns to track three different reference trajectories without requiring any system-specific prior knowledge.

Lastly, Figure 8 also depicts the performance of a pure reinforcement learning method, namely a policy-gradient scheme, applied to the same learning task to serve as a baseline for comparison. Details of the policy-gradient

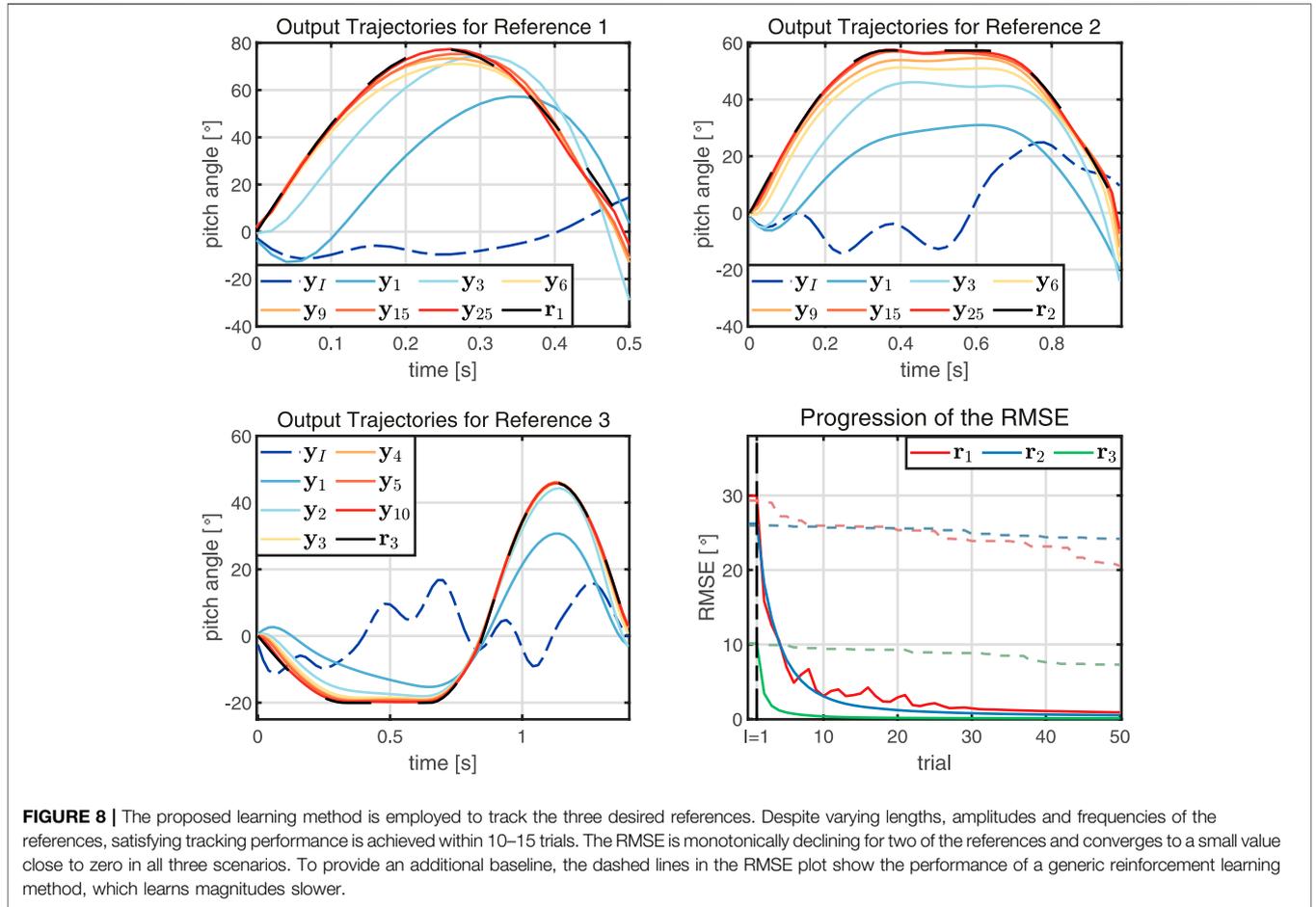


FIGURE 8 | The proposed learning method is employed to track the three desired references. Despite varying lengths, amplitudes and frequencies of the references, satisfying tracking performance is achieved within 10–15 trials. The RMSE is monotonically declining for two of the references and converges to a small value close to zero in all three scenarios. To provide an additional baseline, the dashed lines in the RMSE plot show the performance of a generic reinforcement learning method, which learns magnitudes slower.

algorithm are presented in **Supplementary Appendix S1.4** and (Peters and Schaal, 2006). Here, we see that the RMSE declines at a rate that is by magnitudes slower compared to the learning speed of the proposed method. These results are of little surprise because the policy-gradient algorithm is a generic scheme that is not tailored towards the specific task of learning an input trajectory to track a desired reference trajectory. In contrast, the proposed approach leverages the fact that in reference tracking tasks the input/output dynamics of a nonlinear system can be effectively modelled by a FIR GP-model.

4.3 Robustness Analysis

The previous simulations have validated the method’s capability of achieving satisfying tracking performance when using the automatically determined parameters. However, as discussed above, presenting results for a single parameterization is of little value. Instead, a learning control method should, ideally, not only achieve satisfying performance for a single parameter configuration, but for a wide parameter space. This is a crucial prerequisite for a method that performs well on different systems for different reference trajectories without any manual adjustments.

To address this question, the proposed method’s robustness with respect to the automatically determined parameters is validated in the following study, where we aim at tracking reference r_1 . We

consider two different scenarios, namely, the greedy case of one initial trial, $I = 1$, and the conservative case of five initial trials, $I = 5$. Recall the two parameters s and σ_1^2 , which are the weight in the optimal control problem and the initial input variance.

In the case of $I = 1$, the weight s is chosen from the set \mathcal{S}_1 that consists of ten logarithmically spaced values and the initial input variance σ_1^2 is chosen from the set \mathcal{V}_1 that consists of ten quadratically spaced values, i.e., for $I = 1$.

$$\mathcal{S}_1 = \{10^{-1}, \dots, 10^{-3}\} \quad |\mathcal{S}_1| = 10, \quad (31)$$

$$\mathcal{V}_1 = \{3^2, \dots, 6^2\} \quad |\mathcal{V}_1| = 10, \quad (32)$$

$$(s, \sigma_1^2) \in \mathcal{P}_1 := \mathcal{S}_1 \times \mathcal{V}_1 \quad |\mathcal{P}_1| = 100. \quad (33)$$

For each of the 100 parameter pairs in \mathcal{P}_1 , 50 runs are performed. A run r consists of choosing a parameter pair $(s, \sigma_1^2)_k$ from \mathcal{P}_1 , producing I initial input trajectories, and executing the proposed learning method for an additional 50 trials such that a progression of the RMSE throughout trials is obtained, which we denote by

$$e_{j,k,r}^{\text{RMS}}, \quad (34)$$

where $j \in [0, I + N]$ is the trial index, $k \in [1, 100]$ is the parameter index, and $r \in [1, 50]$ is the run index.

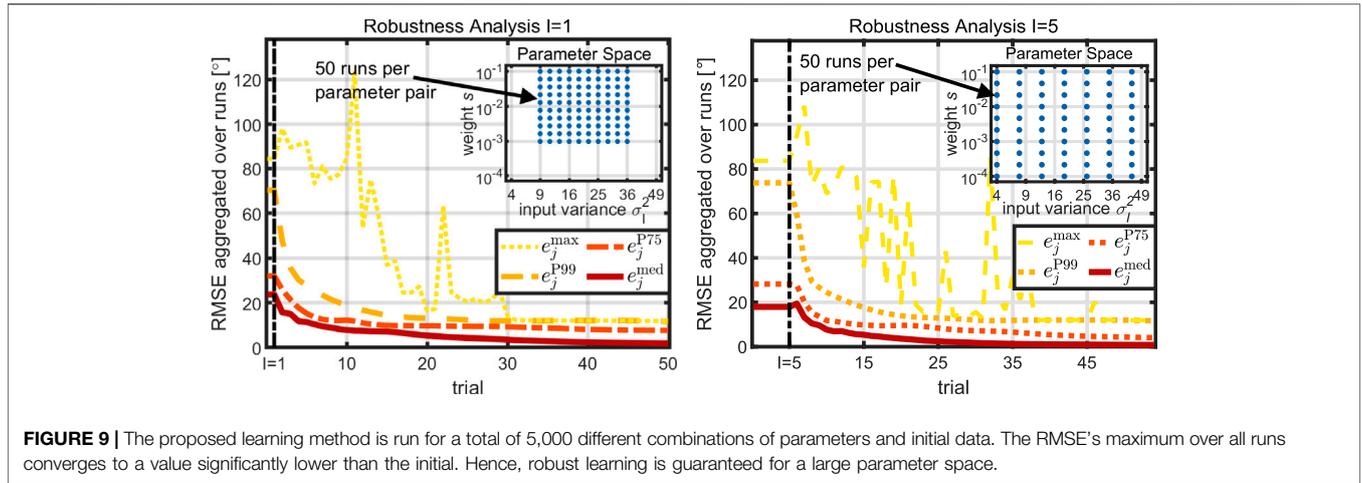


FIGURE 9 | The proposed learning method is run for a total of 5,000 different combinations of parameters and initial data. The RMSE's maximum over all runs converges to a value significantly lower than the initial. Hence, robust learning is guaranteed for a large parameter space.

The same procedure is applied in the case of $I = 5$, but the parameters are chosen from larger sets, i.e., for $I = 5$.

$$\mathcal{S}_5 = \{10^{-1}, \dots, 10^{-4}\} \quad |\mathcal{S}_5| = 10, \quad (35)$$

$$\mathcal{V}_5 = \{3^2, \dots, 7^2\} \quad |\mathcal{V}_5| = 10, \quad (36)$$

$$(s, \sigma_1^2) \in \mathcal{P}_1 := \mathcal{S}_1 \times \mathcal{V}_5 \quad |\mathcal{P}_1| = 100. \quad (37)$$

To evaluate performance, the maximum $e_j^{\max} \in \mathbb{R}$, 99th percentile $e_j^{\text{p99}} \in \mathbb{R}$, 75th percentile $e_j^{\text{p75}} \in \mathbb{R}$, and median $e_j^{\text{med}} \in \mathbb{R}$ of the RMSE over parameters and runs are considered. Formally,

$$\forall j \in \mathbb{N}_{\geq 0}, \quad e_j^{\max} := \max_{k \in [1, 100], r \in [1, 50]} (e_{j,k,r}^{\text{RMSE}}), \quad (38)$$

and e_j^{p99} , e_j^{p75} , e_j^{med} are defined accordingly. Results depicted in **Figure 9** show that, for both $I = 1$ and $I = 5$, the maximum of the RMSE converges to a value that is a roughly ten times smaller than the initial value such that the method's robustness is validated. The RMSE's 99th percentile is monotonically decreasing, which implies that, besides single outliers, the method achieves the desired form of convergence as defined in **Section 2**. This means that the proposed method yields desirable performance for a large range of values of the weight s and the initial input variance σ_1^2 . Furthermore, the RMSE's median declines below a value of five degrees within 25 trials meaning that satisfying tracking performance is achieved. Lastly, it should be noted that a wider parameter space could be considered in the case of $I = 5$ meaning that, by increasing the amount of initial data, the robustness of the method can be further increased. We, hence, conclude that the simple approach proposed to autonomously determine the weight s and the input variance σ_1^2 is more than sufficient and can be expected to suffice for more complex systems.

4.4 Effects of Weights

The previous analysis has shown that the method rapidly learns to track a desired reference while also being robust with respect to the automatically determined parameters. Next to the use case of automatic plug and play application, the method can also be

tuned to meet the needs of a specific application. Hence, we next investigate how the choice of the weight s affects learning characteristics, namely the rate of convergence and robustness with respect to initial data. For this purpose, we consider the weights

$$s \in \{10^0, 10^{-2}, 10^{-4}, 10^{-6}\}. \quad (39)$$

The remaining learning parameters are chosen as one initial trial $I = 1$ and an initial input variance $\sigma_1^2 = 25$. For each weight, 50 runs with differing initial data are performed and performance is judged based on the RMSE's 90th percentile and median over the 50 runs.

Results depicted in **Figure 10** show that for a comparatively large value of $s = 10^0$, the RMSE monotonically declines at a slow pace. Furthermore, there is hardly any difference between median

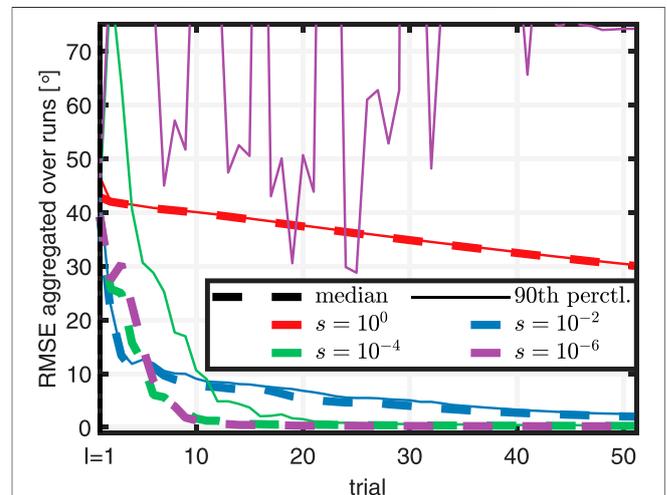
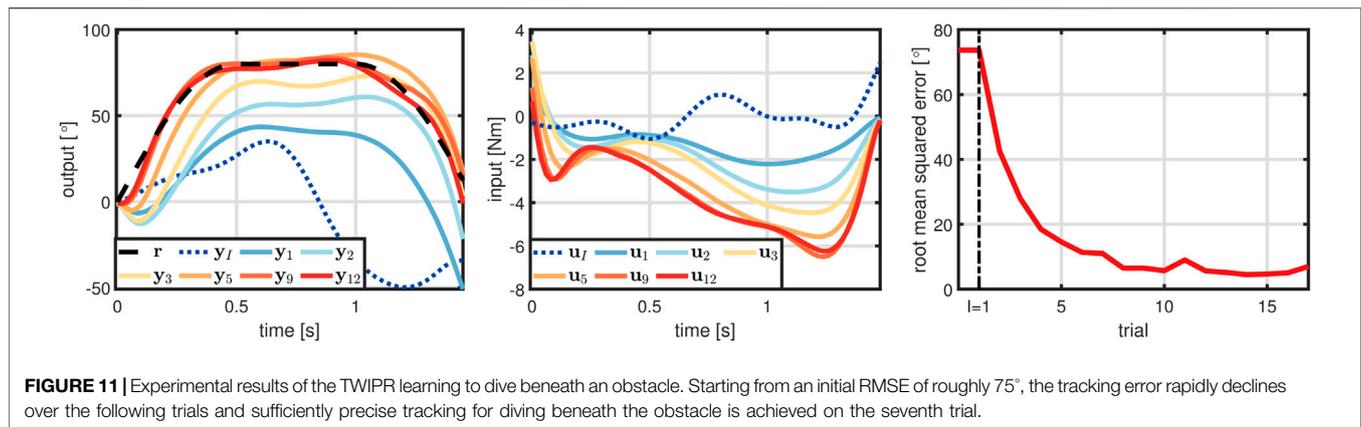


FIGURE 10 | Investigation of the effect of weight s on the learning characteristics: Large values of s lead to slow learning with small performance variance. Increasing the value leads to faster learning but also a larger variance in performance. Excessively small values of s may lead to a RMSE that diverges for some initial data.



and 90th percentile performance. Decreasing the value to $s = 10^{-2}$ leads to a significant increase in speed of convergence. Speed of median convergence can be further increased by lowering the value of the weight to $s = 10^{-4}$, which, however, comes at the price of larger 90th percentile RMSEs, which imply an increase in performance variance. If the weight is lowered to an even smaller value, $s = 10^{-6}$, median performance is not further increased, but the 90th percentile RMSE does no longer converge meaning that learning fails in a significant portion of runs. In summary, the study indicates that the weight s may be used to tune learning behavior, whereby comparatively large values of s lead to slow learning that is robust with respect to initial data. Decreasing the value of s can increase the speed of learning, but may come at the price of sensitivity with respect to initial data.

5 VALIDATION BY EXPERIMENT

To demonstrate the plug and play applicability of the proposed learning method, it is applied to a real-world TWIPR, which has been previously used to validate learning control methods (Meindl et al., 2020). The robot is meant to dive beneath an obstacle as depicted in **Figure 2** with the corresponding reference trajectory $\mathbf{r} \in \mathbb{R}^{75}$, $\forall n \in [1, 75]$,

$$[\mathbf{r}]_n = \begin{cases} 80 \sin(\pi T n) & n \leq 25 \\ 80 & 25 < n \leq 50 \text{ [°]}. \\ 80 \sin(\pi T (n - 25)) & 50 < n \end{cases} \quad (40)$$

First, the proposed method determines the learning parameters yielding $I = 1$, $\sigma_1^2 = 2$, and $s = 0.1$. The initial input trajectory is drawn according to (Eq. 24) and applied to the TWIPR. The corresponding output trajectory significantly differs from the reference with a RMSE of roughly 75°, see **Figure 11**. From here onwards, the method iteratively determines a GP model, updates the input trajectory, and performs an experimental trial. Once learning begins, the RMSE rapidly declines, the RMSE drops below 20° on the fourth trial, and a RMSE of less than 10° is reached on the eighth trial. Sufficiently precise tracking precision for diving beneath the obstacle is achieved on the seventh trial

and a RMSE close to zero is achieved on the 10th trial. Note, that the RMSE slightly increases on some of the trials, which is likely due to the initial conditions varying from trial to trial.

In summary, the experiments validate that the proposed method enables a real-world robot with unknown, nonlinear dynamics to learn a challenging maneuver. Not only did learning require a small number of trials (≈ 10) but the method could also be applied in a plug and play manor without iterative tuning of parameters.

6 DISCUSSION AND CONCLUSION

In this work, a GP-based learning control scheme has been proposed that autonomously solves reference tracking tasks in systems with unknown, nonlinear dynamics. On each iteration, the unknown dynamics are approximated by a Gaussian Process (GP), which is then used to determine and apply an optimal feedforward control input. The method is completely plug and play, since all required algorithms parameters are determined automatically and manual tuning is avoided. The effectiveness and efficiency of the method were demonstrated by simulations and experiments using the example of a two-wheeled inverted pendulum robot that rapidly learns to perform several challenging maneuvers without any manual tuning or system-specific prior knowledge.

In contrast to previous GP-based learning control approaches, the proposed method overcomes the inherent limitations of time-domain feedback control; it neither assumes knowledge of an effective feedback control structure, nor does it assume the entire state vector to be known. Instead, the proposed method directly adjusts the input based on the measured output. It is therefore as model-agnostic and independent of system-specific prior knowledge as pure reinforcement learning schemes.

While reinforcement learning approaches typically require hundreds of trials for convergence and are therefore unsuitable for experimental validation, the proposed learning control method solves reference tracking problems in a small two-digit number of trials and was successfully validated in real-world experiments.

While the vast majority of previous contributions either validate methods only in simulations or provide only a single result for one carefully chosen parameterization and one specific motion, the present validation has proven effectiveness of the proposed method for several different motions and a large range of algorithm parameterizations. We thereby demonstrated robustness with respect to the automatically determined parameters, and we further investigated the effect of the learning weights on the trade-off between speed of learning and robustness.

We believe that the proposed method is highly suitable for use in kinematic systems that must perform challenging, highly dynamic maneuvers. Beyond the use case of rigid robotics, we expect the proposed method to have a major impact on the development of soft robotics, exoskeletons, and neuroprosthetics, and will therefore contribute to the evolution of autonomous robotic systems that rapidly learn to perform complex, dynamic motions under unknown conditions.

Despite these achievements, the proposed method is subject to remaining limitations. First, only learning to track a single reference trajectory was considered, and future research is going to extend the method to multi-reference tracking tasks. This can either be achieved by the concept outlined in Remark 3 or by combining the proposed method with model-free feedback controller learning to handle trial-varying references and disturbances. Second, the proposed method requires a feasible reference trajectory which might not be directly available in some applications. If the reference is only specified at a subset of the trial's samples, the proposed method might be extended by well-established *P2P-ILC* concepts, see (Freeman and Tan, 2013; Janssens et al., 2013; Huo et al., 2020). And in cases in which the motion task is formulated only via goal states and constraints, a prior planning step might be required. Third, future research is going to extend the method to be applicable to multi-input/multi-output systems by implementing multiple GP models with each predicting the progression of one of the respective output

variables. This may require adjustments like, e.g., sparse GP models, see (Rasmussen and Williams, 2005), in order to reduce the computational demands to a tractable amount. Fourth, the proposed method was validated on one real-world application, and future research will be concerned with applying the method to other, complex real-world applications.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

MM developed the proposed learning method and carried out the simulations and experiments under the guidance and advice of TS. MM wrote the first draft of the manuscript, and all authors contributed to manuscript writing and revision.

FUNDING

This work was supported in part by the Deutsche Forschungsgemeinschaft (DFG), German Research Foundation, through Germany's Excellence Strategy-EXC 2002/1 "Science of Intelligence," under Project 390523135, and in part by the Verbund der Stifter through the Project "Robotic Zoo."

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: <https://www.frontiersin.org/articles/10.3389/frobt.2022.793512/full#supplementary-material>

REFERENCES

- Ahn, H.-S., Chen, Y., and Moore, K. L. (2007). Iterative Learning Control: Brief Survey and Categorization. *IEEE Trans. Syst. Man. Cybern. C* 37, 1099–1121. doi:10.1109/tsmcc.2007.905759
- Ai, Q., Ke, D., Zuo, J., Meng, W., Liu, Q., Zhang, Z., et al. (2020). High-Order Model-free Adaptive Iterative Learning Control of Pneumatic Artificial Muscle with Enhanced Convergence. *IEEE Trans. Ind. Electron.* 67, 9548–9559. doi:10.1109/TIE.2019.2952810
- Amann, N., Owens, D. H., and Rogers, E. (1996). Iterative Learning Control for Discrete-Time Systems with Exponential Rate of Convergence. *IEE Proc. Control Theory Appl.* 143, 217–224. doi:10.1049/ip-cta:19960244
- Apgar, T., Clary, P., Green, K., Fern, A., and Hurst, J. (2018). "Fast Online Trajectory Optimization for the Bipedal Robot Cassie," in *Robotics: Science And Systems XIV (Robotics: Science and Systems Foundation)*. doi:10.15607/rss.2018.xiv.054
- Arimoto, S., Kawamura, S., and Miyazaki, F. (1984). Bettering Operation of Robots by Learning. *J. Robot. Syst.* 1, 123–140. doi:10.1002/rob.4620010203
- Atkeson, C. G., Moore, A. W., and Schaal, S. (1997). Lo Cally Weighted Learning for Control. *Artif. Intell. Rev.* 11, 75–113. doi:10.1023/a:1006511328852
- Berkenkamp, F., and Schoellig, A. P. (2015). Safe and Robust Learning Control with Gaussian Processes. 2015 *Eur. Control Conf. ECC 2015*, 2496–2501. doi:10.1109/ECC.2015.7330913
- Bristow, D. A., Tharayil, M., Alleyne, A. G., and Han, Z. Z. (2005). A Survey of Iterative Learning Control. *Kongzhi yu Juece/Control Decis.* 20, 961–966. doi:10.1109/mcs.2006.1636313
- Capone, A., Noske, G., Umlauf, J., Beckers, T., Lederer, A., and Hirche, S. (2020). Localized Active Learning of Gaussian Process State Space Models. *arXiv* 120, 1–10.
- Chi, R., Hou, Z., Huang, B., and Jin, S. (2015a). A Unified Data-Driven Design Framework of Optimality-Based Generalized Iterative Learning Control. *Comput. Chem. Eng.* 77, 10–23. doi:10.1016/j.compchemeng.2015.03.003
- Chi, R., Hou, Z., Huang, B., and Jin, S. (2015b). A Unified Data-Driven Design Framework of Optimality-Based Generalized Iterative Learning Control. *Comput. Chem. Eng.* 77, 10–23. doi:10.1016/j.compchemeng.2015.03.003
- Coulson, C. J., Taylor, R. P., Reid, A. P., Griffiths, M. V., Proops, D. W., and Brett, P. N. (2008). An Autonomous Surgical Robot for Drilling a Cochleostomy: Preliminary Porcine Trial. *Clin. Otolaryngol.* 33, 343–347. doi:10.1111/j.1749-4486.2008.01703.x
- Deisenroth, M. P. (2011). A Survey on Policy Search for Robotics. *FNT Robotics* 2, 1–142. doi:10.1561/23000000021
- Deisenroth, M. P. (2010). *Efficient Reinforcement Learning Using Gaussian Processes*, 9. KIT Scientific Publishing.
- Deisenroth, M. P., and Rasmussen, C. E. (2011a2011). PILCO: A Model-Based and Data-Efficient Approach to Policy Search. *Proc. 28th Int. Conf. Mach. Learn. ICML*, 465–472.

- Deisenroth, M. P., and Rasmussen, C. E. (2011b/2011). PILCO: A Model-Based and Data-Efficient Approach to Policy Search. *Proc. 28th Int. Conf. Mach. Learn. ICML*, 465–472.
- Deisenroth, M., Rasmussen, C., and Fox, D. (2012). Learning to Control a Low-Cost Manipulator Using Data-Efficient Reinforcement Learning. *Robotics Sci. Syst.* 7, 57–64. doi:10.15607/rss.2011.vii.008
- Duy Nguyen-Tuong, D., and Peters, J. (2008). “Local Gaussian Process Regression for Real-Time Model-Based Robot Control,” in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Manhattan, New York: IEEE), 380–385. doi:10.1109/IROS.2008.4650850
- Dydek, Z. T., Annaswamy, A. M., and Lavretsky, E. (2013). Adaptive Control of Quadrotor UAVs: A Design Trade Study with Flight Evaluations. *IEEE Trans. Contr. Syst. Technol.* 21, 1400–1406. doi:10.1109/tcst.2012.2200104
- Feng, S., Whitman, E., Xinjilefu, X., and Atkeson, C. G. (2014). “Optimization Based Full Body Control for the Atlas Robot,” in *2014 IEEE-RAS International Conference on Humanoid Robots (IEEE)*. doi:10.1109/humanoids.2014.7041347 *Optimization Based Full Body Control for the Atlas Robot*
- Freeman, C. T., and Tan, Y. (2013). Iterative Learning Control with Mixed Constraints for Point-to-point Tracking. *IEEE Trans. Contr. Syst. Technol.* 21, 604–616. doi:10.1109/TCST.2012.2187787
- Golovin, I., and Palis, S. (2019). Robust Control for Active Damping of Elastic Gantry Crane Vibrations. *Mech. Syst. Signal Process.* 121, 264–278. doi:10.1016/j.ymssp.2018.11.005
- Gunnarsson, S., and Norrlöf, M. (2001). On the Design of ILC Algorithms Using Optimization. *Automatica* 37, 2011–2016. doi:10.1016/S0005-1098(01)00154-6
- Ha, S., Xu, P., Tan, Z., Levine, S., and Tan, J. (2020). *Learning to Walk in the Real World with Minimal Human Effort*. arXiv preprint arXiv:2002.08550.
- Harib, O., Hereid, A., Agrawal, A., Gurriet, T., Finet, S., Boeris, G., et al. (2018). Feedback Control of an Exoskeleton for Paraplegics: Toward Robustly Stable, Hands-free Dynamic Walking. *IEEE Control Syst.* 38, 61–87. doi:10.1109/mcs.2018.2866604
- Heess, N., Tb, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., et al. (2017). *Emergence of Locomotion Behaviours in Rich Environments*. arXiv preprint arXiv:1707.02286.
- Hehn, M., and D’Andrea, R. (2012). “Real-time Trajectory Generation for Interception Maneuvers with Quadcopters,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE)*, 4979–4984. doi:10.1109/iros.2012.6386093
- Hesse, M., Timmermann, J., Hüllermeier, E., and Trächtler, A. (2018). A Reinforcement Learning Strategy for the Swing-Up of the Double Pendulum on a Cart. *Procedia Manuf.* 24, 15–20. doi:10.1016/j.promfg.2018.06.004
- Hewing, L., Kabzan, J., and Zeilinger, M. N. (2017). Cautious Model Predictive Control Using Gaussian Process Regression. arXiv 28, 2736–2743.
- Hou, Z.-S., and Wang, Z. (2013). From Model-Based Control to Data-Driven Control: Survey, Classification and Perspective. *Inf. Sci.* 235, 3–35. doi:10.1016/j.ins.2012.07.014
- Huo, B., Freeman, C. T., and Liu, Y. (2020). Data-driven Gradient-Based Point-to-point Iterative Learning Control for Nonlinear Systems. *Nonlinear Dyn.* 102, 269–283. doi:10.1007/s11071-020-05941-8
- Janssens, P., Pipeleers, G., and Swevers, J. (2013). A Data-Driven Constrained Norm-Optimal Iterative Learning Control Framework for Lti Systems. *IEEE Trans. Contr. Syst. Technol.* 21, 546–551. doi:10.1109/TCST.2012.2185699
- Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., et al. (2018). Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation (PMLR). *Proc. Mach. Learn. Res.* 87, 651–673.
- Kim, S., and Kwon, S. (2015). Dynamic Modeling of a Two-Wheeled Inverted Pendulum Balancing Mobile Robot. *Int. J. Control Autom. Syst.* 13, 926–933. doi:10.1007/s12555-014-0564-8
- Kober, J., and Peters, J. (2008). “Policy Search for Motor Primitives in Robotics,” in *Advances in Neural Information Processing Systems*. Editors D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (Red Hook: Curran Associates, Inc.), 21.
- Kormushev, P., Calinon, S., and Caldwell, D. (2013). Reinforcement Learning in Robotics: Applications and Real-World Challenges. *Robotics* 2, 122–148. doi:10.3390/robotics2030122
- Lewis, F. L., and Vamvoudakis, K. G. (2011). Reinforcement Learning for Partially Observable Dynamic Processes: Adaptive Dynamic Programming Using Measured Output Data. *IEEE Trans. Syst. Man. Cybern. B* 41, 14–25. doi:10.1109/TSMCB.2010.2043839
- Lewis, F. L., and Vrabie, D. (2009/2009). “Adaptive Dynamic Programming for Feedback Control,” in *Proceedings of 2009 7th Asian Control Conference* (Manhattan, New York: IEEE), 1402–1409.
- Lewis, F. L., Vrabie, D., and Syrmos, V. L. (2012). *Optimal Control*. John Wiley & Sons.
- Lu, J., Cao, Z., Zhang, R., and Gao, F. (2018). Nonlinear Monotonically Convergent Iterative Learning Control for Batch Processes. *IEEE Trans. Ind. Electron.* 65, 5826–5836. doi:10.1109/TIE.2017.2782201
- Ma, L., Liu, X., Kong, X., and Lee, K. Y. (2021). Iterative Learning Model Predictive Control Based on Iterative Data-Driven Modeling. *IEEE Trans. Neural Netw. Learn. Syst.* 32, 3377–3390. doi:10.1109/TNNLS.2020.3016295
- Meindl, M., Molinari, F., Raisch, J., and Seel, T. (2020). *Overcoming Output Constraints in Iterative Learning Control Systems by Reference Adaptation*. arXiv preprint arXiv:2002.00662.
- Moriarty, D. E., and Miikkulainen, R. (2007). Efficient Reinforcement Learning through Symbiotic Evolution, 11, 32. doi:10.1007/978-0-585-33656-5_3
- Muller, F. L., Schoellig, A., and D’Andrea, R. (2012). “Iterative Learning of Feed-Forward Corrections for High-Performance Tracking,” in *Proc. Of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (Manhattan, New York: IEEE), 3276–3281.
- Murphy, R. R. (2004). Activities of the Rescue Robots at the World Trade Center from 11-21 September 2001. *IEEE Robot. Autom. Mag.* 11, 50–61. doi:10.1109/mra.2004.1337826
- Peters, J., and Schaal, S. (2006/2006). Policy Gradient Methods for Robotics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE)*. doi:10.1109/iros.2006.282564
- Peters, J., and Schaal, S. (2008). Reinforcement Learning of Motor Skills with Policy Gradients. *Neural Netw. Neurosci.* 21, 682–697. doi:10.1016/j.neunet.2008.02.003
- Petric, T., Gams, A., Colasanto, L., Ijspeert, A. J., and Ude, A. (2018). Accelerated Sensorimotor Learning of Compliant Movement Primitives. *IEEE Trans. Robot.* 34, 1636–1642. doi:10.1109/tro.2018.2861921
- Rasmussen, C. E., and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.
- Schuitema, E. (2012). *Reinforcement Learning on Autonomous Humanoid Robots*. Ph.D. thesis. Delft, Netherlands: Delft University of Technology. doi:10.4233/UUID:986EA1C5-9E30-4AAC-AB66-4F3B6B6CA002
- Seeger, M., Williams, C. K. I., and Lawrence, N. D. (2003). “Fast Forward Selection to Speed up Sparse Gaussian Process Regression,” in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*. Editors C. M. Bishop and B. J. Frey (Key West, FL).
- Seel, T., Werner, C., Raisch, J., and Schauer, T. (2016). Iterative Learning Control of a Drop Foot Neuroprosthesis - Generating Physiological Foot Motion in Paretic Gait by Automatic Feedback Control. *Control Eng. Pract.* 48, 87–97. doi:10.1016/j.conengprac.2015.11.007
- Shen, D., Zhang, W., and Xu, J.-X. (2016). Iterative Learning Control for Discrete Nonlinear Systems with Randomly Iteration Varying Lengths. *Syst. Control Lett.* 96, 81–87. doi:10.1016/j.sysconle.2016.07.004
- Snelson, E., and Ghahramani, Z. (2006). “Sparse Gaussian Processes Using Pseudo-inputs,” in *Advances in Neural Information Processing Systems*, 1257–1264.
- Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., de Las Casas, D., et al. (2018). *Deepmind Control suite* CoRR Abs/1801, 00690.
- Tayebi, A., and Zaremba, M. B. (2002). Robust ILC Design Is Straightforward for Uncertain LTI Systems Satisfying the Robust Performance Condition. *IFAC Proc. Vol.* 35, 445–450. doi:10.3182/20020721-6-es-1901.01060
- Tsounis, V., Alge, M., Lee, J., Farshidian, F., and Hutter, M. (2020). Deepgait: Planning and Control of Quadrupedal Gaits Using Deep Reinforcement Learning. *IEEE Robot. Autom. Lett.* 5, 3699–3706. doi:10.1109/LRA.2020.2979660
- Umlauf, J., Beckers, T., Capone, A., Lederer, A., and Hirche, S. (2020). *Smart Forgetting for Safe Online Learning with Gaussian Processes*. Learning for Dynamics & Control. Proceedings in Machine Learning Research (PMLR), 1–10.
- Vinogradskaja, J., Bischo, B., Nguyen-Tuong, D., and Peters, J. (2017). Stability of Controllers for Gaussian Process Dynamics. *J. Mach. Learn. Res.* 18, 1–37.
- Vinogradskaja, J., Bischoff, B., Nguyen-Tuong, D., Schmidt, H., Romer, A., and Peters, J. (2016/2016). Stability of Controllers for Gaussian Process Forward Models. *33rd Int. Conf. Mach. Learn. ICML* 2, 819–828.

- Wenjie Dong, W., and Kuhnert, K.-D. (2005). Robust Adaptive Control of Nonholonomic Mobile Robot with Parameter and Nonparameter Uncertainties. *IEEE Trans. Robot.* 21, 261–266. doi:10.1109/TRO.2004.837236
- Yu, Q., Hou, Z., Bu, X., and Yu, Q. (2020). RBFNN-based Data-Driven Predictive Iterative Learning Control for Nonaffine Nonlinear Systems. *IEEE Trans. Neural Netw. Learn. Syst.* 31, 1170–1182. doi:10.1109/tnnls.2019.2919441
- Zeng, A., Song, S., Lee, J., Rodriguez, A., and Funkhouser, T. (2020). Tossingbot: Learning to Throw Arbitrary Objects with Residual Physics. *IEEE Trans. Robot.* 36, 1307–1319. doi:10.1109/TRO.2020.2988642

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Copyright © 2022 Meindl, Lehmann and Seel. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.