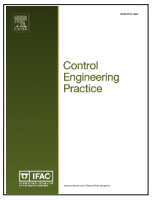
Control Engineering Practice 172 (2026) 106904

Contents lists available at ScienceDirect

Control Engineering Practice

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



Data-driven high-performance motion system optimization in practice

Alisa Rupenyan [a,*], Narek Bayanduryan-Levasgani [b], Mohammad Khosravi [c]

[a] ZHAW Centre for Artificial Intelligence, ZHAW Zürich University of Applied Sciences, Winterthur, 8401, Switzerland
[b] inspire AG, Technoparkstrasse 1, Zurich, CH-8005, Switzerland
[c] Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, Delft, 2628 CD, The Netherlands

ARTICLE INFO

Keywords:
Gaussian process model
Model predictive control
Bayesian optimizaiton
Motion system
Joint optimization

ABSTRACT

Manufacturing productivity demands positioning systems that achieve both high speed and micrometer-level accuracy. Existing approaches rely on time-intensive manual tuning or optimize controller layers independently. We extend prior work on data-driven controller tuning to demonstrate that jointly optimizing Model Predictive Contouring Control (MPCC) planner parameters and low-level controller gains using constrained Bayesian optimization improves performance beyond sequential or isolated tuning strategies. The approach models system performance metrics such as traversal time, tracking accuracy, and vibration levels over complete geometric trajectories as joint Gaussian processes, enabling sample-efficient exploration of the combined parameter space while respecting physical constraints. Numerical results show that joint optimization achieves 8–23% improvement in traversal time and 2.5 - 5 × reduction in maximum contour errors compared to optimizing either layer independently. Experimental validation on precision motion hardware demonstrates that MPCC parameter optimization alone (with pre-tuned low-level gains) achieves 15% improved maximal tracking error at a 6% faster traversal time. The framework is system-agnostic and requires no hardware modifications.

1. Introduction

High-performance positioning systems are essential in manufacturing and must provide exceptional positioning accuracy at high speeds. Such performance can be achieved with model predictive control (MPC) designed to track a 2D contour (Lam et al., 2010a; Liniger et al., 2019), but it requires careful tuning and powerful processing capabilities of the underlying hardware. It is possible to design an ideal trajectory offline using the superior planning capabilities of MPC, which is then input to a low-level PID controller (Yang et al., 2015a). While this strategy improves system tracking performance without requiring fast online computation, it involves adjusting many settings in both the MPC planner and the low-level controller.

One of the main requirements for a successful MPC implementation is the availability of an accurate model of the system dynamics. Alternatively, the performance degrades due to imposed robustness constraints. Instead of limiting the controller to worst-case scenarios, systematic tuning of the parameters in the MPC optimization objective can be exploited for maximum performance (Forgione et al., 2020; Lu et al., 2020; Piga et al., 2019; Sorourifar et al., 2020). The approach relies on efficiently finding parameters, such as optimal weights in the MPC cost, prediction horizon length, and even dynamic model parameters through Bayesian optimization, and tuning is done with respect to optimizing the performance of the system. The performance criteria can be represented by features in the data measured during system operation at different values of the optimized parameters (Khosravi et al., 2020), as demonstrated for applications in motion systems or in HVAC systems (Catenaro et al., 2024; Khosravi et al., 2019a,b; Savaia et al., 2021). Bayesian optimization selects interactively each subsequent candidate configuration of parameters for evaluation in order to minimize the number of evaluations while finding a global optimum (Gardner et al., 2014a). The approach has been successfully applied to real-world applications, including linear and rotational axes embedded in grinding machines and shown to standardize and automate tuning of multiple parameters (Busetto et al., 2023; Khosravi et al., 2022).

Providing optimized trajectories to the low-level controller is an established approach in high-precision machining (Yang et al., 2015b). Contouring MPC (Liniger et al., 2019) is particularly suitable for such set-point optimization due to the convenient formulation of trade-offs between speed and accuracy. In such multi-objective optimization problems, tuning the parameters of an MPC controller is critical, especially when pushing the system to the limit is desired, e.g., in racing (Gharib et al., 2021; Vázquez et al., 2020), and in high-volume, high-precision manufacturing (motion axis and stages, e.g., Balta et al., 2021; König et al., 2021).

* Corresponding author.
E-mail addresses: rupn@zhaw.ch (A. Rupenyan), narek.bayanduryan@inspire.ch (N. Bayanduryan-Levasgani), mohammad.khosravi@tudelft.nl (M. Khosravi).

https://doi.org/10.1016/j.conengprac.2026.106904
Received 6 October 2025; Received in revised form 2 March 2026; Accepted 5 March 2026
Available online 12 March 2026

Numerical results on precision motion systems demonstrate that Bayesian optimization can significantly push the system performance when used for tuning of the parameters (horizon length and weights in the optimization objective) of an MPC-based planner for reference trajectories. In particular, the underlying dynamics model for the predictive controller is low fidelity (Balta et al., 2021; Rupenyan et al., 2021). Optimization is based on a set of metrics designed to quantify the overall performance throughout the trajectory. It has also recently been shown that the performance of the motion system PID controller can be optimized in a run-to-run approach that incorporates safe exploration (König et al., 2025), relying on the acquisition of data from the real system.

This work builds upon prior demonstrations of BO-based controller tuning for motion systems (Khosravi et al., 2022; König et al., 2025; Rupenyan et al., 2021) and hierarchical control structures (Khosravi et al., 2020). Our specific contributions are threefold: (1) We demonstrate numerically that *joint* optimization of high-level MPCC planner parameters and low-level PI gains achieves superior performance both in traversal time and in tracking accuracy, compared to optimizing either layer in isolation, even when the underlying dynamics model is deliberately low-fidelity. This extends (Rupenyan et al., 2021), which optimized only MPCC parameters, and Khosravi et al. (2021, 2020), which studied cascade controllers but not in the MPCC contouring context. (2) We introduce operational constraints to improve the practical feasibility of the MPC formulation (acceleration and velocity ranges) and performance constraints (maximum jerk, maximum contour error, or maximum traversal time) in the BO implementation. (3) We provide experimental validation on micrometer-precision hardware, demonstrating two cost functions and constraint formulations and convergence behavior. The joint optimization benefits are demonstrated numerically, and experimental results focus on MPCC-only tuning on a precision motion system. We demonstrate that performance-based joint tuning can compensate for model inaccuracies without explicit model refinement, as evidenced by the performance improvements achieved despite the simplified second-order dynamics model. These contributions extend MPCC-based tuning to include constraint handling, joint optimization across controller layers, and experimental validation on real hardware for the optimization of the MPCC parameters.

This paper is organized as follows. In Section 2, we introduce the parameter tuning in contouring control with MPC controllers. Section 4 presents the contouring MPC approach, used for trajectory optimization. Section 3 describes the plant and its low-level controller, and in Section 5 we introduce the proposed approach for tuning the MPC and the low-level parameters. We then present numerical results in Section 6 and experimental results in Section 7.

2. Data-driven parameter tuning in contouring control

In this paper, we focus on the precise 2-D positioning problem for a biaxial gantry. The goal is to traverse a path with a given geometry in a minimum time and remain in a specified tolerance band around the path. We measure the deviation from the reference geometry using the $2-$norm and the $\infty-$norm of the error along the specified parametrized trajectory (see Section 4 and Rupenyan et al. (2021) for details on the parametrization), and allow a maximum deviation of 30 μm over the whole trajectory. The movement is further constrained following the physical constraints of the motion stage on its velocity, v, and acceleration, u, where the maximum velocity in both x and y directions is 0.2m/s and the maximum acceleration is 20m/s^2.

The system has a low-level cascade PI controller. Before applying any commands, the reference geometry is processed by a MPC-based planning unit denoted by MPCC, for Model Predictive Contouring Control. The underlying model used in the planner is a low-fidelity linear model identified using a limited number of target geometries for the positioning task. The planner thus introduces several tunable parameters in its optimization objective, without extra complexity in the modelling. The goal is to improve the system's performance by tuning the parameters of the planner and of the low-level controller, while leaving the structure of the controller and the physical properties of the machine intact.

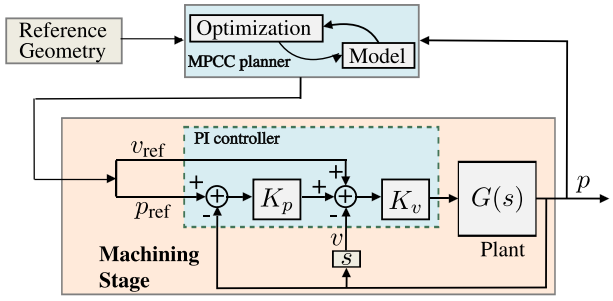


Fig. 1. Schematic of the positioning system, showing the MPCC planner, the machining stage, including the low-level controller, and the identified plant $G(s)$.

The system exhibits high repeatability with run-to-run variations of approximately 2 μm (see Section 6), which is substantially smaller than the positioning errors we aim to minimize (8–30 μm range). This repeatability enables sample-efficient data-driven tuning: performance metrics computed from a single trajectory execution reliably characterize system behavior for a given parameter configuration, without requiring extensive averaging over multiple runs. The tuning procedure thus iteratively evaluates different parameter settings by executing complete geometry traversals and computing global performance metrics (traversal time, maximum contour error, etc.) from the measured position and velocity data.

3. Plant and low level control

Our physical system is a 2-axis gantry stage for (x, y) positioning with industrial-grade actuators and sensors. Following Rupenyan et al. (2021), we model it as a mass-spring-damper system, which is suitable to include friction effects in the motion of the system (Qian et al., 2016). We model each axis independently, with an identical dynamics model, but different identified parameters, indicated by w. The detailed model is provided in Rupenyan et al. (2021).

For the description of the position trajectory, let p_w be the position trajectory obtained by simulating the corresponding dynamics with respect to the parameters w, and J be the error metric between the real measurement of position p and the simulated position p_w defined as $J(w) := \|p - p_w\|_\infty + \|p - p_w\|_2$. The error metric J includes the overshoots via the infinity norm, and the offset, via the 2-norm. The parameters w are estimated via $\text{argmin}_w J(w)$, using the available solvers for nonlinear optimization.

The plant is controlled by a cascade PI controller for tracking the position and velocity reference trajectories (see Fig. 1). The force input applied to the system is

$$u = K_v(K_p e_p + e_v), \tag{1}$$

where (K_p, K_v) are the nominal gains of the controller, and e_p and e_v are the error signals, respectively, in tracking the desired position and velocity trajectories.

The transfer function from the reference trajectories to the actual position and velocity is as follows,

$$\begin{bmatrix} p \\ v \end{bmatrix} = \frac{1}{H(s)} \begin{bmatrix} H_1(s) & H_2(s) \\ sH_1(s) & sH_2(s) \end{bmatrix} \begin{bmatrix} p_{\text{ref}} \\ v_{\text{ref}} \end{bmatrix}, \tag{2}$$

where $p := (x, y)$, $p_{\text{ref}} := (x_{\text{ref}}, y_{\text{ref}})$, $v := (\dot{x}, \dot{y})$, $v_{\text{ref}} := (\dot{x}_{\text{ref}}, \dot{y}_{\text{ref}})$, H_1, H_2 and H are defined as $H_1(s) = K_v K_p G(s)$, $H_2(s) = K_v G(s)$, and $H(s) = K_v s G(s) + 1 + K_v K_p G(s)$. Given (2), we obtain a discrete time model

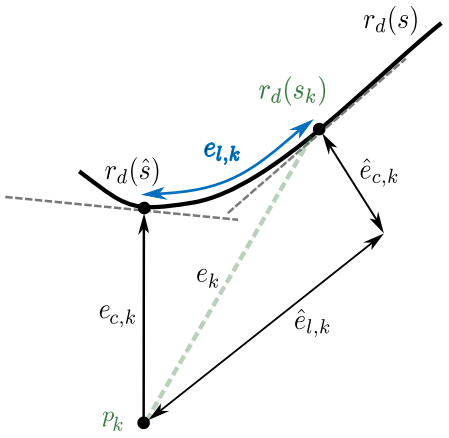


Fig. 2. The contour and the lag errors $e_{c,k}$ and $e_{l,k}$ are the vector components of the total error $\boldsymbol{e_k} = \boldsymbol{r_d}(s_k) - \boldsymbol{p_k}$ in MPCC. Their approximations are visualized and denoted by $\hat{e}_{c,k}$ and $\hat{e}_{l,k}$.

with sampling time $T = 2.5$ms as

$$\begin{cases} \vartheta_{k+1}^{(x)} = A_x \vartheta_k^{(x)} + B_x \begin{bmatrix} x_{\text{ref},k} \\ \dot{x}_{\text{ref},k} \end{bmatrix}, \\ \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} = C_x \vartheta_k^{(x)}, \end{cases} \tag{3}$$

in x-axis, and

$$\begin{cases} \vartheta_{k+1}^{(y)} = A_y \vartheta_k^{(y)} + B_y \begin{bmatrix} y_{\text{ref},k} \\ \dot{y}_{\text{ref},k} \end{bmatrix}, \\ \begin{bmatrix} y_k \\ \dot{y}_k \end{bmatrix} = C_y \vartheta_k^{(y)}, \end{cases} \tag{4}$$

in y-axis, where $[x_k, \dot{x}_k]^\top$, $[y_k, \dot{y}_k]^\top$, $[x_{\text{ref},k}, \dot{x}_{\text{ref},k}]^\top$ and $[y_{\text{ref},k}, \dot{y}_{\text{ref},k}]^\top$ are respectively sampled version of $[x, \dot{x}]^\top$, $[y, \dot{y}]^\top$, $[x_{\text{ref}}, \dot{x}_{\text{ref}}]^\top$ and $[y_{\text{ref}}, \dot{y}_{\text{ref}}]^\top$, and, $\vartheta_k^{(x)}$ and $\vartheta_k^{(y)}$ respectively denote the state vectors in the corresponding discrete-time dynamics.

4. MPC planner

The tracking of a 2D-geometry is described via the model predictive contour control (MPCC) framework, proposed in Lam et al. (2010b). We denote the k^{th} sample of the real position of system by $\boldsymbol{p_k} := (x_k, y_k) \in \mathbb{R}^2$. Following Rupenyan et al. (2021), we introduce a virtual path parameter, s_k, defined as $s_{k+1} = s_k + T v_{s,k}$, where $v_{s,k}$ is the sampled velocity along the path at time step k and T is the sampling time. To avoid projecting the current position onto the path via $\hat{s}_k := \text{argmin}_{s\in[0,L]} \|\boldsymbol{r_d}(s) - \boldsymbol{p_k}\|$ at each time step, MPCC introduces a *virtual path parameter* s_k that approximates this projection dynamically. This parameter evolves according to $s_{k+1} = s_k + T v_{s,k}$, where $v_{s,k} \geq 0$ is the value of the velocity along the path at time step k, an optimization variable that determines the progression rate along the path, and T is the sampling time (Liniger et al., 2019; Rupenyan et al., 2021). This formulation avoids expensive geometric projections and allows the controller to optimize both tracking accuracy and traversal speed simultaneously. s_k is not tied to a pre-computed time-based trajectory. The MPC controller determines the optimal progression rate $v_{s,k}$ online to achieve time-optimal performance while satisfying constraints. The tracking errors are introduced as states with virtual dynamics in the optimization problem. We describe the deviations in following the reference geometry via approximated errors, the contour error $\hat{e}_c$, and the lag error $\hat{e}_l$, following the schematic in Fig. 2.

We define the *total error vector* $\boldsymbol{e_k} = \boldsymbol{r_d}(s_k) - \boldsymbol{p_k}$, decomposed into the *contour error* $e_{c,k}$ (perpendicular distance to the path, representing geometric tracking accuracy) and the *lag error* $e_{l,k}$ (tangential distance along the path, indicating how far the virtual parameter s_k is ahead of or behind the actual system progress). These are approximated in the MPC as $\hat{e}_{c,k}$ and $\hat{e}_{l,k}$ by projecting e_k onto the normal and tangent directions at $\boldsymbol{r_d}(s_k)$ (Fig. 2).

We further perform first order linearization around the operating point, then introduce the following dynamics for the errors

$$\begin{aligned} \hat{e}_{l,k+1} = &\frac{r'_{d,x}(s_k)}{\|\boldsymbol{r'}(s_k)\|}(r_{d,x}(s_k) - x_k - T\dot{x}_k - \frac{1}{2}T^2\ddot{x}_k) \\ &+ \frac{r'_{d,y}(s_k)}{\|\boldsymbol{r'}(s_k)\|}(r_{d,y}(s_k) - y_k - T\dot{y}_k - \frac{1}{2}T^2\ddot{y}_k) + T v_{s,k}, \end{aligned} \tag{5}$$

$$\begin{aligned} \hat{e}_{c,k+1} = &-\frac{r'_{d,y}(s_k)}{\|\boldsymbol{r'}(s_k)\|}(r_{d,x}(s_k) - x_k - T\dot{x}_k - \frac{1}{2}T^2\ddot{x}_k) \\ &+ \frac{r'_{d,x}(s_k)}{\|\boldsymbol{r'}(s_k)\|}(r_{d,y}(s_k) - y_k - T\dot{y}_k - \frac{1}{2}T^2\ddot{y}_k). \end{aligned} \tag{6}$$

The parametric derivative of $\boldsymbol{r_d}$ is denoted by $\boldsymbol{r'_d}(s) := (r'_{d,x}(s), r'_{d,y}(s))$, and $\ddot{x}_k, \ddot{y}_k$ are sampled accelerations of the system in the x and y coordinates.

The virtual path parameter is a good approximation of the minimal deviation between the real and the desired position only when the lag error $\hat{e}_{l,k}$ is small, ensuring that $\hat{e}_{c,k}$ is in turn well approximated. The dynamics of the errors enable their inclusion in the MPC formulation as state variables - namely, the parametric derivatives $\hat{e}_{l,k}$ and $\hat{e}_{c,k}$, and the path parameter s_k. The other states of the real system (e.g., velocity and acceleration of each axis) originate from the identified, discretized state-space model.

We now collect the state variables with real and virtual dynamics in the state vector $\mathbf{z}_k$ as

$$\mathbf{z}_k := [\vartheta_k^{(x)\top}, r_{x,k}, \dot{r}_{x,k}, \vartheta_k^{(y)\top}, r_{y,k}, \dot{r}_{y,k}, s_k, \hat{e}_{l,k}, \hat{e}_{c,k}]^\top, \tag{7}$$

where $r_{x,k}$, $\dot{r}_{x,k}$, $r_{y,k}$, and $\dot{r}_{y,k}$ are sampled version of $r_{d,x}, \dot{r}_{d,x}, r_{d,y}$, and $\dot{r}_{d,y}$, respectively. The inputs are collected in the vector $\mathbf{u}_k$, defined as $\mathbf{u}_k := [\ddot{x}_k \ \ddot{y}_k \ v_{s,k}]^\top$. Following (3)–(6), we obtain a linear time-varying system of the following form.

$$\mathbf{z}_{k+1} = \mathrm{A}_k \mathbf{z}_k + \mathrm{B}_k \mathbf{u}_k + \mathbf{d}_k, \tag{8}$$

where only the error dynamics are time-dependent.

The contouring trajectory optimization problem can now be formulated as

$$\begin{aligned} \min_{\mathbf{Z},\mathbf{U}} \ &\sum_{k=1}^{N-1} \left(\gamma_l \hat{e}_{l,k}^2 + \gamma_c \hat{e}_{c,k}^2 - \gamma_v v_{s,k} + \mathbf{u}_k^\top R \mathbf{u}_k + \Delta_k \mathbf{u}^\top S \Delta_k \mathbf{u} \right) \\ &+ \gamma_{l,N} \hat{e}_{l,N}^2 + \gamma_{c,N} \hat{e}_{c,N}^2 - \gamma_{s,N} s_N, \\ \text{s.t. } &\mathbf{z}_{k+1} = \mathrm{A}_k \mathbf{z}_k + \mathrm{B}_k \mathbf{u}_k + \mathbf{d}_k, \qquad k = 0, ., N-1, \\ &\hat{e}_{c,k} \in \mathcal{T}^c, \quad v_k \in \mathcal{V}, \quad u_k \in \mathcal{U}, \qquad k = 0, ., N-1, \\ &v_N \in \mathcal{V}_N, \quad \hat{e}_{c,N} \in \mathcal{T}_N^c, \end{aligned} \tag{9}$$

where $\mathbf{z}_0$ is given, $\Delta_k \mathbf{u} := \mathbf{u}_k - \mathbf{u}_{k-1}$ is the change in the acceleration, $\mathbf{Z} := (\mathbf{z}_1, \ldots, \mathbf{z}_N)$ and $\mathbf{U} := (\mathbf{u}_0, \ldots, \mathbf{u}_{N-1})$ are the state and input trajectories, γ_l and γ_c are the error weights, R is a diagonal positive definite input weight matrix with non-zero terms $\gamma_{\ddot{x}}$ and $\gamma_{\ddot{y}}$ for the acceleration along the two axes, S is a positive definite weight matrix applied to the acceleration differences with non-zero diagonal terms $\gamma_{\ddot{x}}$ and $\gamma_{\ddot{y}}$, γ_v is a linear reward for path progress and, $\mathcal{T}^c$ is the tolerance band of ± 20 µm, $\gamma_{l,N}$ and $\gamma_{c,N}$ are terminal contouring weights, and $\gamma_{s,N}$ is the weight considered for the progress maximization. The multi-objective cost function of the MPCC planner penalizes the squared longitudinal error $\hat{e}_l^2$ and the squared contouring error $\hat{e}_c^2$, rewards progress at the end of the horizon s_N, encourages fast traversal, and penalizes the acceleration inputs and their change (jerk). Thus, it is designed to achieve accurate and rapid tracking while simultaneously minimizing vibrations in the system.

5. Data-driven tuning for contouring control

In this section, we first present the preliminary concepts required for the proposed data-driven tuning methodology. More precisely, to enable high-performance contouring control without manual calibration, we develop a model-free and data-driven parameter adaption scheme utilizing probabilistic modeling and optimization. Accordingly, we begin by introducing the essential preliminaries, namely Gaussian process regression for modeling system performance metrics and constraints, and Bayesian optimization for efficient parameter selection.

5.1. Preliminaries

5.1.1. Gaussian process models

A Gaussian Process (GP) is a collection of random variables with a jointly Gaussian distribution (Rasmussen, 2006). Let $\mathcal{GP}(\mu, k)$ be a Gaussian process prior for a function $f(\xi)$, where $\mu : \mathbb{X} \to \mathbb{R}$ and $k : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ are respectively the mean and kernel functions in the GP prior taken for f. Define information matrices $\chi_m := [\xi_1, \dots, \xi_m]^\mathsf{T}$. Subsequently, for any $\xi \in \Xi$, we have

$$f^{(\mathrm{j})}(\xi) \sim \mathcal{N}\left(\mu_m^{(\mathrm{j})}(\xi), v_m^{(\mathrm{j})}(\xi)\right), \qquad j = 0, 1, 2, \tag{10}$$

with mean and variance defined as

$$\mu_m^{(\mathrm{j})}(\xi) := \mu^{(\mathrm{j})}(\xi) + \mathrm{k}_m^{(\mathrm{j})}(\xi)^\mathsf{T}(\mathrm{K}_m^{(\mathrm{j})} + \sigma_j^2\mathbb{I})^{-1}(\boldsymbol{\xi}_m^{(\mathrm{j})} - \mu^{(\mathrm{j})}(\chi_m)), \tag{11}$$

and

$$v_m^{(\mathrm{j})}(\xi) := k^{(\mathrm{j})}(\xi, \xi) - \mathrm{k}_m^{(\mathrm{j})}(\xi)^\mathsf{T}(\mathrm{K}_m^{(\mathrm{j})} + \sigma_j^2\mathbb{I})^{-1}\mathrm{k}_m^{(\mathrm{j})}(\xi), \tag{12}$$

where $\mathbb{I}$ denotes the identity matrix, σ_j^2 is the variance of uncertainty in the evaluations of $f_j(\xi)$, vector $\mathrm{k}_m^{(\mathrm{j})}(\xi)$ and $\mu^{(\mathrm{j})}(\chi_m)$ are respectively defined as

$$\mathrm{k}_m^{(\mathrm{j})}(\theta) := [k^{(\mathrm{j})}(\xi, \xi_1), \dots, k^{(\mathrm{j})}(\xi, \xi_m)]^\mathsf{T} \in \mathbb{R}^m, \tag{13}$$

and

$$\mu^{(\mathrm{j})}(\chi_m) = [\mu^{(\mathrm{j})}(\xi_1), \dots, \mu^{(\mathrm{j})}(\xi_m)]^\mathsf{T} \in \mathbb{R}^m, \tag{14}$$

and matrix $\mathrm{K}_m^{(\mathrm{j})}$ is defined as $[k^{(\mathrm{j})}(\xi_{i_1}, \xi_{i_2})]_{i_1,i_2=1}^m \in \mathbb{R}^{m\times m}$.

5.1.2. Bayesian optimization

In constrained Bayesian optimization (Gardner et al., 2014a; Shahriari et al., 2015), we need to solve the following optimization problem

$$\begin{aligned} \min_{\xi\in\Xi} \quad & f(\xi) \\ \text{s.t.} \quad & c(\xi) \le c_{\mathrm{tr}}, \end{aligned} \tag{15}$$

where f is unknown and expensive to evaluate, Ξ is the feasible set for ξ, and c_{tr} is a constraint threshold.

The performance objective function f and the constraint c cannot be analytically calculated, i.e., they might not have a tractable closed-form expression, even when the dynamics of system are known. Each of these functions is given in a *black-box oracle* form, and only noisy samples of them can be observed. In Bayesian optimization (BO), using the available noisy samples, we can create a probabilistic surrogate model using Gaussian process regression (or any model that provides uncertainty estimation) of the objective function, which maps the optimization variables to the observed noisy evaluations of f. Then, instead of optimizing the unknown objective, another function is used, which is easy to access and evaluate, an acquisition function, which trades exploration and exploitation, and chooses the subsequent points for evaluation using the GP mean and uncertainty over the whole available range. Following Rupenyan et al. (2021), we use as an acquisition function the *expected improvement* function, $a_{\mathrm{EI},m} : \Theta \to \mathbb{R}$, where

$$a_{\mathrm{EI},m}(\xi) := \left(\eta_m(\xi)\Phi(\eta_m(\xi)) - \varphi(\eta_m(\xi))\right) v_m^{(0)}(\xi)^{\frac{1}{2}}. \tag{16}$$

Here, Φ and φ respectively denote the cumulative distribution function and the probability density function of the standard normal distribution, and $\eta_m(\xi)$ is defined as

$$\eta_m(\xi) := (\mu_m^{(0)}(\xi) - \zeta_m^{0,+})/v_m^{(0)}(\xi)^{\frac{1}{2}}, \tag{17}$$

where $\zeta_m^{0,+}$ denotes the minimal cost observed within the first m experiments. As a standard practice for constrained problems, we multiply the expected improvement function by the probability of feasibility, to obtain the *constrained expected improvement* (Gardner et al., 2014a) as follows:

$$a_{\mathrm{CEI,m}}(\xi) = a_{\mathrm{EI,m}}(\xi)\,\Phi\left(\frac{q_{1,\mathrm{tr}} - \mu_m^{(1)}(\xi)}{v_m^{(1)}(\xi)^{\frac{1}{2}}}\right)\Phi\left(\frac{q_{2,\mathrm{tr}} - \mu_m^{(2)}(\xi)}{v_m^{(2)}(\xi)^{\frac{1}{2}}}\right). \tag{18}$$

The second term in $a_{\mathrm{CEI,m}}(\xi)$ indicates the feasibility probability of the design parameters ξ. To solve the optimization problem (15), we need to solve the following problem iteratively

$$\xi_{m+1} := \operatorname*{argmax}_{\xi\in\Xi} a_{\mathrm{CEI},m}(\xi). \tag{19}$$

until a stopping criterion is met.

5.2. Data-driven tuning methodology - problem setting

We now use our simple model and the error dynamics within the MPC planner as a base for the MPC approach, while tuning the MPC cost parameters and other low-level control parameters. This tuning aims to both achieve high performance and to compensate for model imperfections. The controuring MPC parameters (9) and the lower level control parameters introduced in Section 3, are thus collected in the vector ξ defined as

$$\xi := [\gamma_c, \gamma_l, \gamma_{\dot{x}}, \gamma_{\dot{y}}, \gamma_{\ddot{x}}, \gamma_{\ddot{y}}, \gamma_v, N, K_p, K_v]^\mathsf{T}. \tag{20}$$

The parameter vector ξ in (20) contains both continuous parameters (weights γ_c, γ_l, etc., and gains K_p, K_v) and one discrete parameter (horizon length N). For the discrete parameter, we define a finite set of candidate values $N \in \{10, 15, 20, 25, 30\}$ based on computational constraints and preliminary manual tuning. The GP model treats N as continuous within its bounds, and the acquisition function is maximized over the continuous relaxation. When a new evaluation point ξ_{m+1} is selected, we round N to the nearest value in the candidate set before performing the experiment. The rounded value of N is used both for the trajectory execution and as the data point recorded in D_m. This approach is standard practice for mixed-integer BO (Garrido-Merchán & Hernández-Lobato, 2020) and works well when the number of discrete values is small.

We now formulate the high-level optimization problem aiming at addressing both accuracy and traversal time. The corresponding cost function and constraints are defined based on the entire position and velocity trajectory. This takes into account the overall performance and the operation limits corresponding to our control scheme. As we will be exploring different configurations of gain parameters, we will work with a predefined set of gains that does not disturb stability in the system. Such sets can be obtained through several strategies depending on available system knowledge: If approximate dynamics are available (as in our case via the identified model in Section 3), classical stability analysis (Nyquist, gain/phase margins) can establish conservative bounds. In case suitable ranges are not reccomended during commissioning, it is possible to start from known stable configurations (e.g., factory defaults), and the parameter ranges can be gradually expaneded while monitoring stability indicators (oscillations, settling time). When no prior knowledge is available, techniques based on safe exploration for Bayesian optimization (Zagorowska et al., 2023) can be employed to learn safe regions online through cautious exploration with probabilistic safety guarantees, even in the presence of drifts and changes in the system (Li et al., 2024). These methods typically require more conservative initial exploration but provide rigorous stability certificates. For our experimental setup, the identified model provides conservative

stability bounds which we validate through initial manual tests before beginning BO.

The second-order mass-spring-damper model in the MPCC planner neglects higher-order dynamics, nonlinear friction effects, and cross-coupling between axes. This low-fidelity model serves as a nominal predictor, while the BO-based parameter tuning compensates for model inaccuracies by directly optimizing parameters based on measured system performance. An alternative approach would be to iteratively refine the model structure or parameters alongside controller tuning. However, this presents several challenges: (i) online model learning requires additional excitation signals that may compromise safety constraints, (ii) model parameter uncertainty propagates through the MPCC optimization, potentially degrading robustness, and (iii) the combined model-controller optimization significantly increases computational complexity.

The goals of the motion planning problem are to traverse the given geometry as fast as possible and to adhere to the predefined tracking tolerances. The first goal is directly reflected by the total number of time steps necessary to traverse the whole geometry, denoted by k_{tot}. Note that k_{tot} implicitly depends on the vector of parameters ξ; accordingly, we define the cost function as

$$g_0(\xi) := k_{\text{tot}}\,.$$

For the second goal, aiming to minimize deviations and oscillations in the system, we introduce two constraints as

$$\begin{aligned} g_1(\xi) &:= \max\left(\left\|[\ddot{x}_k]_{k=1}^{k_{\text{tot}}}\right\|_\infty, \left\|[\ddot{y}_k]_{k=1}^{k_{\text{tot}}}\right\|_\infty\right) \le q_{1,\text{tr}},\\ g_2(\xi) &:= \left\|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\right\|_\infty \le q_{2,\text{tr}}, \end{aligned} \tag{21}$$

where $\ddot{x}_k, \ddot{y}_k$ is sampled jerk of the system in the x and y coordinates, and $q_{1,\text{tr}}$ and $q_{2,\text{tr}}$ are threshold values for the constraints, i.e. for the maximal allowed jerk and the maximal allowed deviation along the whole geometry. The dependency of these constraints on the vector of parameters is highlighted by the argument ξ employed for g_1 and g_2. The first constraint, $g_1(\xi) \le q_{1,\text{tr}}$, limits oscillations in the system due to unaccounted changes in acceleration originating in the MPCC stage. The second constraint, $g_2(\xi) \le q_{2,\text{tr}}$, limits the maximal allowed lateral error along the trajectory, and serves as an additional tuning knob on the tracking error, apart from the contouring approach outlined in Section 4. Reversing the roles of the constraints and the objective function can move the trade-off in a different direction, and the framework proposed here easily enables this possibility. To tune the parameters ξ, we need to solve the following optimization problem:

$$\begin{aligned} \min_{\xi\in\Xi}\quad & g_0(\xi)\\ \text{s.t.}\quad & g_i(\xi) \le q_{i,\text{tr}}, \quad i = 1,2, \end{aligned} \tag{22}$$

with Ξ being the set of admissible ξ vectors. For a given $\xi \in \Xi$, the value of $g_0(\xi)$, $g_1(\xi)$ and $g_2(\xi)$ can be obtained by performing an experiment, where the design parameters are set to ξ, a complete cycle of operation of the system is performed, the values of the three functions are calculated based on the resulting full trajectory of position and velocity. We use Bayesian optimization to solve (22) with a minimal number of evaluations. Both the objective function and the constraints are unknown functions, which can be sampled; thus, constrained BO learns both functions online by iteratively sampling them and finds the optimum based on the Gaussian Process models of both functions. Occasional constraint violations during optimization are acceptable, because the physical system has built-in safety limits (Table 2) that prevent damage, and the constraints in (22) represent performance targets rather than hard safety boundaries. The work in König et al. (2025) demonstrates integration of safe BO for motion control when hard safety constraints are required.

5.3. Data-driven tuning methodology - Bayesian optimization

We assume that we are given an initial set of design parameters $\xi_{\text{init}} := \{\xi_i \in \xi \mid i = 1,\dots,m_{\text{init}}\}$. This set can be obtained either by random feasible perturbations of nominal design parameters, ξ_{nom}, or, one may employ Latin hyper-cube experiment design (Mckay et al., 2000) to have maximally informative ξ_{init}. The proposed tuning approach maintains a set of data $\mathcal{D}_m$ defined as

$$\mathcal{D}_m := \left\{(\xi_i, \zeta_i^{(0)}, \zeta_i^{(1)}, \zeta_i^{(2)}) \,\middle|\, i = 1,\dots,m\right\},$$

where m is the iteration index, ξ_i is the design parameters corresponding to the i^{th} experiment, and, for $j = 0,1,2$, $\zeta_i^{(\text{j})}$ denotes the computed value for $g_j(\xi_i)$ based on the measured data in the i^{th} experiment. More precisely, we define $\zeta_i^{(\text{j})} = g_j(\xi_i) + n_i^{(\text{j})}$, for all i and j, where $n_i^{(\text{j})}$ is a noise term introduced to capture the impact of uncertainty in data collection. At each iteration m, we build the GP surrogate models $\hat{g}_m^{(\text{j})}$ for the objective and the constraints.

Using these probabilistic surrogate GP models, we can select the parameters $\xi_{m+1} \in \xi$, which will provide the subsequent evaluation of g_j. The parameters are effectively selected by the constrained expected improvement acquisition function, which selects the parameter corresponding to the highest feasible expected improvement of the cost following (18). The optimization problem of the acquisition function can be solved using *particle swarm* methods, or local approximations such as Zagorowska et al. (2023). We iteratively perform this for every subsequent ξ_{m+1}, and add the newly evaluated values of $\{g_j(\xi_{m+1})\}_{j=0}^2$ to the dataset

$$\mathcal{D}_{m+1} = \mathcal{D}_m \cup \{(\xi_{m+1}, \zeta_{m+1}^{(0)}, \zeta_{m+1}^{(1)}, \zeta_{m+1}^{(2)})\}\,. \tag{23}$$

This iterative procedure generates a sequence of $\{\xi_m | m = 1,2,\dots\}$ which converges to the solution of (22) (Gardner et al., 2014b).

Fig. 3 summarizes the Bayesian optimization-based scheme for data-driven tuning of MPCC. We use squared-exponential (RBF) kernels for $k^{(j)}$, $j = 0,1,2$, with automatic relevance determination (ARD), i.e., each parameter in ξ has its own lengthscale. Using ξ_{init}, the kernel hyperparameters (lengthscales and output variance) are optimized by maximizing the marginal likelihood. After the initial training phase (20–30 samples for our experiments), we fix these hyperparameters for the remainder of the optimization to ensure stable convergence. The learned ARD lengthscales provide insight into parameter sensitivity: shorter lengthscales indicate parameters to which the objective is more sensitive. The hierarchical structure of the controller encodes different sensitivities, and the more sensitive MPC-controller parameters (see also Table 1) correspond to shorter length scales. Thus, parameters with long length scales could be fixed earlier, also reducing the dimensionality for further optimization iterations.

Furthermore, the optimization algorithm is terminated following a stopping criterion. Various criteria can be applied, such as a limit for the maximum number of iterations (as in our implementation), or observation of repeated consecutive solutions of (19) with approximately minimal observed cost (Khosravi et al., 2021). We set the mean functions as $\mu^{\text{j}} = 0$, $j = 0,1,2$ (Rasmussen, 2006). If we have prior information on the shape and structure of g_j, $j = 0,1,2$, e.g., from similar experiments or numerical simulations, we can employ other options for the mean functions (Rasmussen, 2006). The full implementation is provided in Algorithm 1.

For repetitive positioning tasks on stable systems, the proposed performance-based tuning can compensate for model mismatch without MPC model refinement. Thus, the simplest “off-the-shelf” model should be sufficient to improve the system’s performance. The parameters ξ in (20) encode corrections for unmodeled dynamics through their influence on the MPCC cost function. As demonstrated further in the experimental results, Table 4, this strategy achieves 15% improvement in tracking accuracy (maximal tracking error) despite the simplified model.

The proposed approach optimizes 8–10 parameters in Eq. (20), which is within the effective range of standard GP-based Bayesian optimization. For industrial systems requiring higher-dimensional tuning (e.g., > 20 parameters), several extensions are available: local search region methods that constrain exploration to promising subspaces (Paulson et al., 2023), trust-region approaches that iteratively refine local

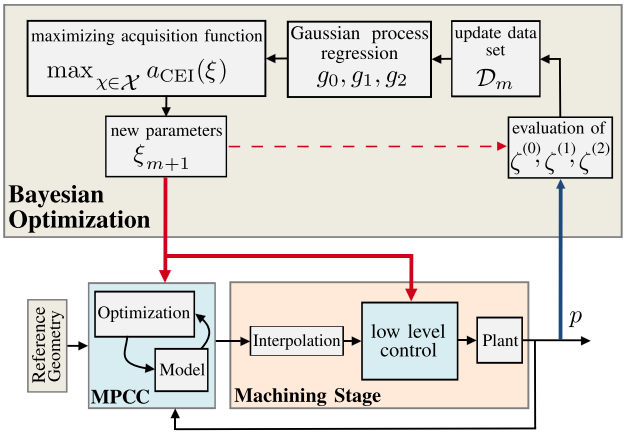


Fig. 3. The scheme of data-driven tuning of MPCC based on Bayesian optimization.

approximations (Eriksson et al., 2019), and decomposition strategies that exploit hierarchical controller structure to tune subsystems sequentially or in groups (Zagorowska et al., 2023). Alternatively, dimensionality reduction techniques such as active subspace methods or random embeddings can identify low-dimensional representations of the parameter space where optimization remains tractable (Nayebi et al., 2019).

Algorithm 1 Constrained Bayesian optimization for controller tuning.

Input: Initial parameter set $\xi_{\text{init}} = \{\xi_1, \ldots, \xi_{m_{\text{init}}}\}$, constraint thresholds $q_{1,\text{tr}}, q_{2,\text{tr}}$, maximum iterations $m_{\max}$
Output: Optimized parameters ξ^*
Initialize dataset $\mathcal{D}_{m_{\text{init}}} = \emptyset$
for $i = 1$ to m_{init} **do**
  Execute trajectory with parameters ξ_i on the physical system
  Measure position and velocity over complete trajectory
  Compute $\zeta_i^{(0)} = g_0(\xi_i)$, $\zeta_i^{(1)} = g_1(\xi_i)$, $\zeta_i^{(2)} = g_2(\xi_i)$
  $\mathcal{D}_{m_{\text{init}}} \leftarrow \mathcal{D}_{m_{\text{init}}} \cup \{(\xi_i, \zeta_i^{(0)}, \zeta_i^{(1)}, \zeta_i^{(2)})\}$
end for
Initialize GP models $\hat{g}^{(j)}_{m_{\text{init}}}$ for $j = 0, 1, 2$ using $\mathcal{D}_{m_{\text{init}}}$
Optimize kernel hyperparameters by maximizing marginal likelihood, then fix
Set $m \leftarrow m_{\text{init}}$
repeat
  Compute acquisition function $a_{\text{CEI},m}(\xi)$ using Eq. (18)
  $\xi_{m+1} \leftarrow \arg\max_{\xi\in\Xi} a_{\text{CEI},m}(\xi)$
  Execute trajectory with parameters ξ_{m+1} on the physical system
  Measure position and velocity over complete trajectory
  Compute $\zeta_{m+1}^{(0)} = g_0(\xi_{m+1})$, $\zeta_{m+1}^{(1)} = g_1(\xi_{m+1})$, $\zeta_{m+1}^{(2)} = g_2(\xi_{m+1})$
  $\mathcal{D}_{m+1} \leftarrow \mathcal{D}_m \cup \{(\xi_{m+1}, \zeta_{m+1}^{(0)}, \zeta_{m+1}^{(1)}, \zeta_{m+1}^{(2)})\}$
  Update GP posteriors $\hat{g}^{(j)}_{m+1}$ for $j = 0, 1, 2$ using $\mathcal{D}_{m+1}$
  $m \leftarrow m + 1$
until $m \geq m_{\max}$ or stopping criterion met
return $\xi^* \leftarrow$ best feasible parameter from $\mathcal{D}_m$

6. Numerical results

This section presents numerical validation of the joint optimization framework. We compare three strategies: (1) optimizing MPCC parameters only with fixed low-level gains, (2) optimizing low-level gains only with fixed MPCC parameters, and (3) joint optimization of all parameters. This comparative study quantifies the added value of joint tuning, a contribution distinct from prior work (Rupenyan et al., 2021) which demonstrated only MPCC optimization and (Khosravi et al., 2021) which addressed cascade controllers without trajectory planning and MPC-based control parameters. The experimental validation in Section 7 focuses on MPCC-only tuning due to hardware access constraints and the marginal gains from low-level tuning demonstrated here.

We study the performance of the introduced approach on a geometry containing corners (octagon) and on a smooth geometry (double circle shape) with different curvatures, as shown in Fig. 4. We use Matlab in conjunction with Yalmip/Gurobi to solve the corresponding contouring MPC quadratic program in a receding horizon fashion and the GPML library for Gaussian process modeling. The repeatability of the system is characterized by deviations of $2\mu m$ between runs, which is a small fraction of our error bounds of $20\mu m$. This repeatable motion enables pre-optimizing the planner and the corresponding input references (position and velocity reference for the low-level controller, for the whole geometry), before measuring the performance over the entire trajectory.

The two error metrics used to compare performance are defined as follows: $\|\hat{e}_c\|_\infty := \|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\|_\infty$, and $\|\hat{e}_c\|_2 := \|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\|_2$, over the whole geometry.

We have previously reported in Rupenyan et al. (2021) that adding the MPC planner is beneficial and results in a large improvement over the traversal time, while the deviations are maintained within the bounds, whereas the low-level controller without an MPC planner exceeds the bounds by up to $45\mu m$.

We now aim to optimize the results further using the MPC Planner. This involves reducing manual tuning of the MPC weights and demonstrating that jointly tuning MPC Planner parameters and low-level controller gains further improves performance.

6.1. BO-based tuning of the MPC parameters and of the low-level controller parameters

We proceed with tuning the MPC planner parameters, while the low-level controller gains are fixed, following the optimization problem from (22). We start the optimization with a training phase of 20–30 experiments (either randomly selected, or using a design of experiments approach such as Latin Hypercube design). This ensures good priors of the Gaussian process models, however, the optimization can also start from one random sample and zero mean for the Gaussian process. The threshold in the performance constraint from (22) is set to $q_{1,\text{tr}} = 2000m/s^3$, and we introduce an additional tracking error constraint with $q_{2,\text{tr}} = 5\mu m$ which further guides the selection of parameters beneficial for improving tracking. This global tracking error constraint has the effect of increasing the traversal time by 10% compared to manual tuning, while maintaining the maximal deviation within the bounds (see Table 1). While the traversal time increases slightly compared to manually tuned MPC, the maximum deviation error $\|\hat{e}_c\|_\infty$ decrease is 5-fold, especially for the double circle geometry. The constraints on the jerk successfully result in avoiding large jumps in acceleration. The performance metrics corresponding to the manually tuned MPC planner are on the top row in Table 1.

As discussed in more detail in Rupenyan et al. (2021), optimising the low-level controller gains has only a marginal effect on tracking performance. We provide the performance metrics here for the sake of comparison, in Table 1. Since the constraint imposed on the jerk does not apply to these parameters, the deviation error $\|\hat{e}_c\|_2$ increases.

6.2. Joint BO of the MPCC and low level parameters

Finally, we optimize the MPC parameters jointly with low-level controller gains. The optimization is performed with safety and performance constraints on the tracking accuracy and on the change of acceleration, following (22). As the number of optimization variables is now higher compared to the previous cases, we increase the number of training samples in the initialization phase to 30 samples, chosen again with Latin hypercube sampling. The initialization phase provides informative priors for the GP models and enables safe exploration by identifying feasible regions before optimization begins. Alternatively, it can be replaced by manual tuning using expert intuition. Fig. 6

Joint parameter tuning achieves both shorter traversal time and more efficient deceleration and acceleration when compared to the nominal case. The resulting smaller deviation from the reference geometry is

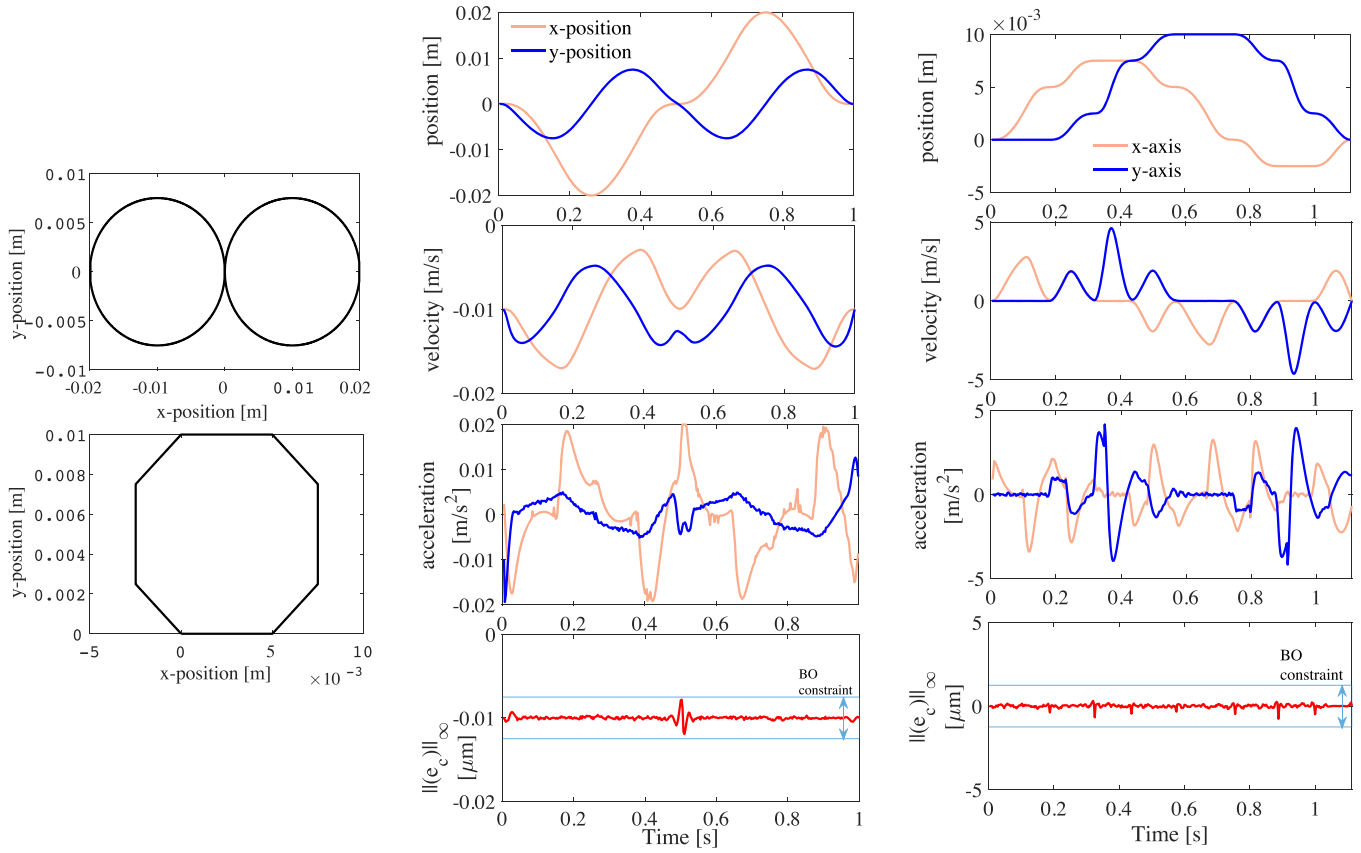


Fig. 4. Left panel: Double circle and octagon target geometries. Center and right panels: Position, velocity, acceleration, and maximal contour error resulting from joint optimization of the MPC and the cascade controller for double circle (center) and octagon (right) geometries.

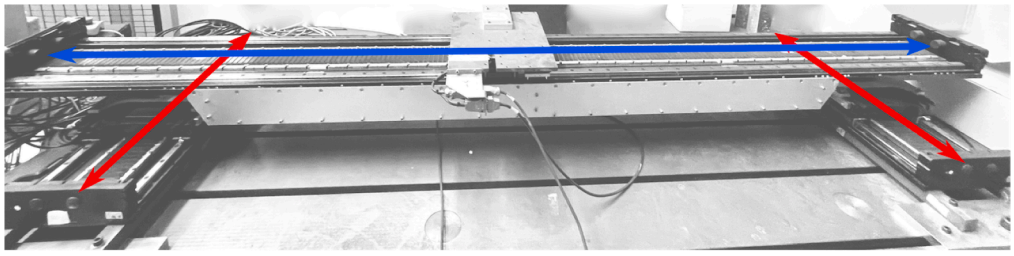


Fig. 5. Biaxial motion stage used in this work, with the two axes indicated by arrows.

shown in Fig. 4. Relaxing the constraint allows for optimizing the traversal time while exploiting all the freedom within the provided bounds. After the initial learning phase, the algorithm finds the optimum within 20 BO iterations for the smooth geometry, whereas for the octagon, the number of iterations is larger. The confidence interval in the cost prediction narrows, especially for the octagon trajectory, possibly due to a better pronounced minimum in the cost of this geometry.

The proposed iterative tuning method can be compared with other approaches that learn from previous iterations, such as iterative learning control (ILC) methods (Barton & Alleyne, 2008). The key difference lies in how learning occurs across task repetitions. The BO approach optimizes static controller parameters (MPCC weights and low-level gains) that remain fixed during trajectory execution, whereas ILC learns feedforward control signals that are updated iteration-to-iteration based on tracking errors, directly compensating for repetitive disturbances and model inaccuracies. ILC-based methods can achieve superior tracking performance (35-47% contour error reduction over 100 iterations in Barton & Alleyne, 2008), requiring exact trajectory repetition and provide improvements only for specific, repeated tasks. In contrast, the BO framework is capable of generalization across different trajectories executing on the same system, as the optimized parameters encode corrections for systematic errors rather than trajectory-specific feedforward signals. Additionally, BO-based tuning requires only black-box access to adjust controller parameters, making it more practical for industrial systems where modifying low-level control software is often infeasible or undesirable.

7. Experimental results

We now present the experimental confirmation of the presented tuning approach, applied to the MPCC parameters, which provide higher performance gain than the already optimized low level controller parameters.

7.1. Experimental set-up

The experimental setup is a biaxial motion stage, shown in Fig. 5, and consists of an ETEL DXL-LM325 two-stage motion system. The

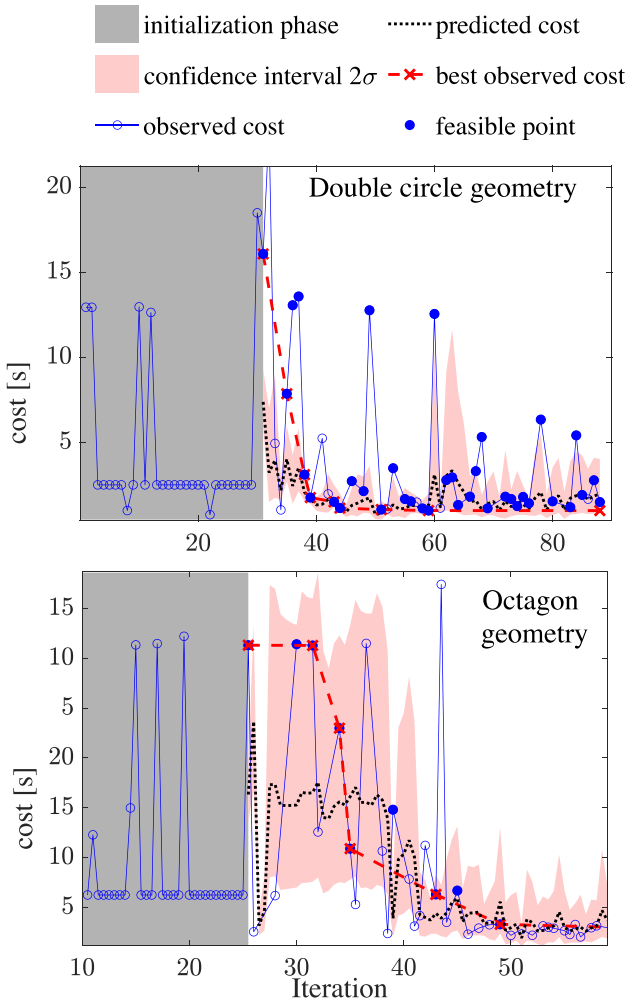


Fig. 6. Evolution of BO iterations, until optimization terminates.

Table 1
Performance metrics evaluating tracking accuracy and time. The MPCC controller including the $\|\hat{e}_c\|_\infty$ constraint is separately indicated.

MPCC - manual tuning, (fixed low-level parameters)	time [s]	$\|\hat{e}_c\|_\infty$ [μm]	$\|\hat{e}_c\|_2$ [μm]
double circle	1.29	20	3.8
octagon	1.22	19.6	0.78
BO - MPCC, (fixed low-level parameters)			
double circle	1.11	20	7.4
octagon	0.97	8.9	0.55
BO - MPCC, (fixed low-level parameters) $\|\hat{e}_c\|_\infty$ constraint			
double circle	1.259	3.78	0.29
octagon	1.12	7.8	0.29
BO - low-level parameters, (fixed MPCC parameters)			
double circle	0.93	20	3.9
octagon	0.97	20	1.17
BO - MPCC, (tuning all parameters) $\|\hat{e}_c\|_\infty$ constraint			
double circle	0.99	4.4	2.28
octagon	1.11	2.86	0.26

Table 2
Software-limited values and operational figures of the experimental setup.

Property	Value	Unit
Maximum acceleration	40	m s^{-2}
Maximum velocity	1.5	m s^{-1}
Working area X axis	$[-0.19, 0.19]$	m
Working area Y axis	$[-1.305, 1.305]$	m
Main control loop frequency	10	kHz
Repeatability	2	μm

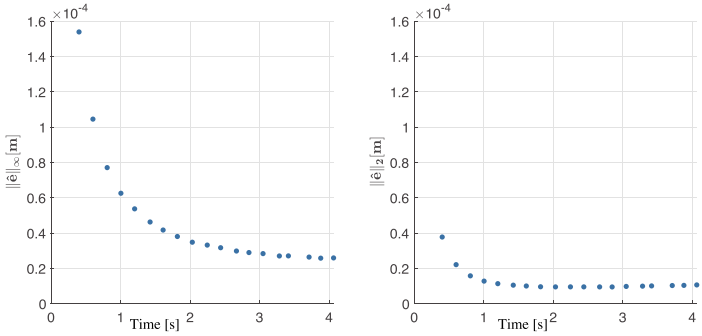


Fig. 7. Error metrics plotted against traversal time for the nominal case. Each point is an average of 5 repetitions.

motion system is instrumented with quadrature encoders that measure the position of both axes and is actuated with ETEL LMS15-100 linear motors. The system is equipped with a feedback controller that adjusts the motor voltages to track a supplied reference trajectory, shown in Fig. 1. The motion stage has limits on the acceleration, velocity, and position of the reference trajectory to prevent accidental damage. More details about it are provided in Balula et al. (2024), and the main characteristics of the setup are provided in Table 2.

7.2. Nominal performance

For the evaluation of the nominal performance, the octagon geometry is used directly as the input to the low-level controller, without any optimization. We set the sampling time to $T = 2.5$ms to match the sampling time used in the numerical study. As shown in Fig. 7, the error metrics indicate that contour error decreases with lower velocities (i.e., longer traversal times), but the rate of improvement diminishes as traversal time is increased further. Thus, empirically, the minimal possible $\|\hat{e}\|_\infty$ is found to be at around $25\mu m$, corresponding to the slowest measured trajectories, whereas the mean error over one trajectory ($\|\hat{e}\|_2$) is at $10\mu m$, and is reached at moderately fast trajectories.

7.3. Performance following BO for trajectory optimization

This section demonstrates the tuning of the MPCC controller on the real system for two formulations of the cost and the constraints, corresponding to different priorities: the first is the traversal time as a main objective, with a fixed contour error; the second is the tracking performance as a main objective, with a fixed traversal time as a constraint. The nominal trajectories used for comparison are selected from the previously sampled nominal set such that their traversal times closely match those of the corresponding optimized trajectories. This enables a consistent comparison of the reported tracking performance metrics.

7.3.1. Traversal time as BO objective

We now run Bayesion optimization to optimize the MPCC parameters θ, excluding the low-level controller gains, according to the scheme in Fig. 3. We initialize with one randomly selected sample, i.e., without any collection of training samples. At first, our BO cost function

Table 3
Performance metrics (average of 15 repetitions) following BO of the MPCC parameters, using the traversal time as BO cost.

	time [s]	$\|\|e_c\|\|_\infty$ [μm]	mean (e_c) [μm]
optimized	3.27	25.6	13.8
nominal	3.29	27.8	10.0

Table 4
Performance metrics (average of 15 repetitions) following BO of the MPCC parameters using the maximal contour error as a BO cost.

	time [s]	$\|\|e_c\|\|_\infty$ [μm]	mean (e_c) [μm]
optimized	2.29	27.2	13.9
nominal	2.44	32.0	9.6

corresponds to the total number of time steps, as explained in Section 5.2 and the enforced constraints correspond to the two contour errors, $g_1(\theta) := \left\|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\right\|_\infty \leq q_{1,\text{tr}}$, and $g_2(\theta) := \left\|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\right\|_2 \leq q_{2,\text{tr}}$ where $q_{1,\text{tr}} = 35\mu$m and $q_{2,\text{tr}} = 25\mu$m, set according to the observed nominal performance. At each iteration of the Bayesian optimization, the MPCC model is run with the selected parameters θ and the input $\vec{u}_{1:k_{\text{tot}}}$, which has been optimized with regards to the MPCC cost function, is used as the reference for the machine. The resulting measurement data from the run is then used in the calculation of the BO constraints. As a stopping criterion, a maximum number of 50 iterations is chosen. This approach, however, brings a marginal improvement over the nominal performance. The corresponding performance metrics are shown in Table 3. The average errors are calculated from 15 repetitions for both the nominal and optimized trajectories. The sample standard deviation of all reported error metrics is below 0.2μm, an order of magnitude smaller than the difference in maximum error of 2.2 μm between the nominal and optimized cases.

7.3.2. *Maximum error as BO objective*

We also tried a different formulation of the objective function and constraints for the Bayesian optimization by swapping the time duration with the maximal contour error, $g_0(\theta) := \left\|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\right\|_\infty$ and $g_1(\theta) = k_{\text{tot}} \leq q_{1,\text{tr}}$, $g_2(\theta) := \left\|[\hat{e}_{c,k}]_{k=1}^{k_{\text{tot}}}\right\|_2 \leq q_{2,\text{tr}}$. In this case, $q_{1,\text{tr}} = 1500$ timesteps, equivalent to 3.75s given the sampling time of 2.5ms, and $q_{2,\text{tr}} = 40\mu$m.

Fig. 8 shows the corresponding evolution of BO iterations. The resulting reference and output trajectories of the best feasible point are shown in Figs. 9 and 10. It is evident that the optimized trajectory slows down near the corners and is faster on straight sections of the geometry. When the parameter optimization requires multiple evaluations, the computational load of the GP model could become heavy. In this case, local optimization approaches as demonstrated in Paulson et al. (2023), Zagorowska et al. (2025) have been useful to overcome the GP matrix inversion bottleneck.

The corresponding performance metrics are summarized in Table 4, again showing the average results from 15 repetitions. This time, we have a 15% improvement in the maximal contour error at a 6% faster traversal time. While the average error is not improved, it remains well below the constraint threshold in the optimized case. The sample deviation observed for all reported metrics is again below 0.2μm, indicating that the differences reported in Table 4 are consistent across repetitions and statistically significant. Visual inspection of Fig. 8 suggests that performance plateaued after approximately 35–40 iterations. In comparison, manual tuning by an expert control engineer typically requires 20–30 trial-and-error experiments (depending on the number of parameters to tune) to achieve satisfactory performance.

While Section 6 demonstrates numerically that joint optimization of MPCC and low-level parameters achieves superior performance, the

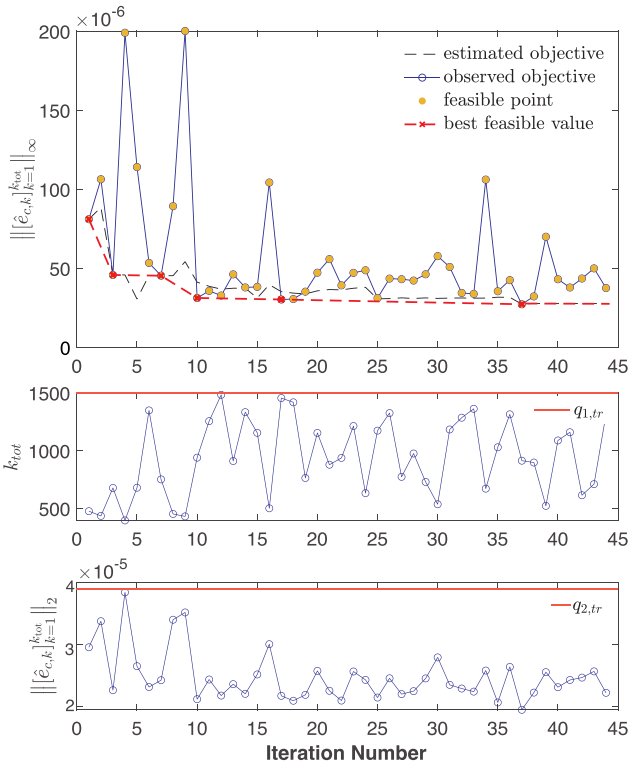


Fig. 8. Evolution of Bayesian optimization iterations.

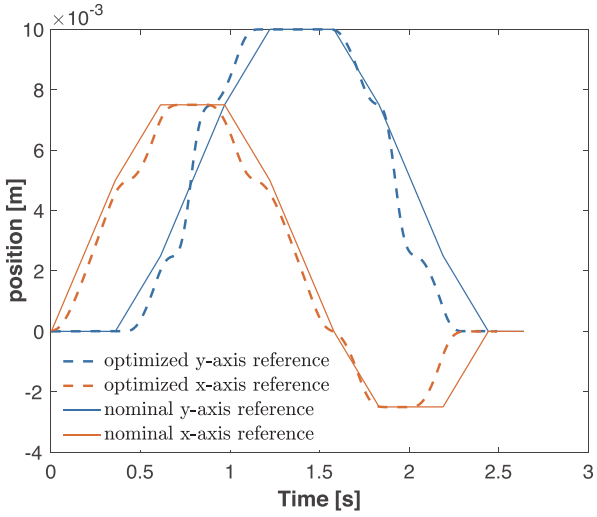


Fig. 9. Optimized and nominal reference trajectories for the alternative BO formulation.

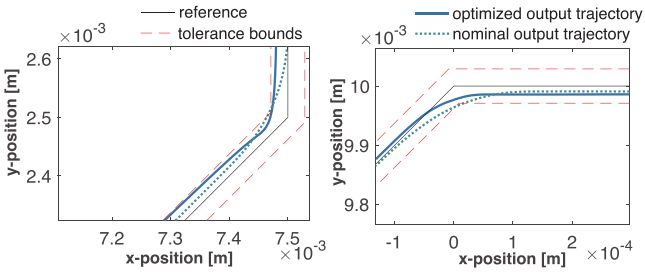


Fig. 10. Optimized and nominal output trajectories at the corners of the octagonal trajectory measured on the experimental gantry setup.

focus of the experimental validation is solely on MPCC parameter tuning with fixed low-level gains, due to restricted machine availability. In practice, the dimensionality of the optimization problem would be increased from 8 to 10 parameters, requiring more experimental trials to achieve convergence. As the MPCC parameters exhibit higher sensitivity to performance, as reflected by the fivefold reduction in maximum contour error reported in Table 1, the experimental results provide strong evidence supporting the effectiveness of the proposed approach. Future work will validate joint tuning experimentally, with a focus on how actuator nonlinearities, sensor noise, and unmodeled friction dynamics affect the convergence and performance of high-dimensional Bayesian optimization on real hardware.

Nonetheless, as our numerical experiments have shown, the proposed methodology can be utilized for performance improvement of wide range of systems with similar setup and structure. MPCC planner parameters ($\gamma_c, \gamma_l, \gamma_v$, etc.) primarily encode trade-offs between speed, accuracy, and smoothness that may generalize well across machines of the same type executing similar tasks, particularly when the underlying MPCC model structure and geometric constraints are identical. Low-level controller gains (K_p, K_v) are more sensitive to system-specific dynamics (mass, friction, actuator characteristics) and will require refinement.

8. Conclusion

This paper demonstrated numerically and experimentally how an improvement in the productivity in positioning systems can be achieved without any changes in the design of the system or its software. We first optimize the input to a low level controller using a contouring predictive control approach. We then propose an automated tuning procedure for selecting the parameters that enable maximal performance. In our proposed sample-efficient tuning algorithm, the performance metrics associated with the full geometry traversal are modelled as Gaussian processes. The resulting models are used to form the cost and the constraints in a constrained Bayesian optimization algorithm, where they enable the trade-off between fast traversal, high tracking accuracy, and suppression of vibrations in the system. We showed that this data-driven tuning of all the parameters compensates for model imperfections and results in improved performance, in terms of time and tracking accuracy. The experimental validation of the algorithm on a micrometer precision motion system shows up to a 19% reduction in maximum contour error, depending on cost function formulation, combined with a 6% improvement in traversal time. Further increase in the system performance was demonstrated numerically, and can be achieved by tuning the system controller's gain parameters. The algorithm can be extended towards geometry-independent tuning, by parametrically including the target geometry in the trajectory optimization stage, and in the modelling of the performance metrics.

CRediT authorship contribution statement

Alisa Rupenyan: Writing – review & editing, Writing – original draft, Visualization, Resources, Project administration, Funding acquisition, Formal analysis, Conceptualization; **Narek Bayanduryan-Levasgani:** Software, Investigation, Data curation; **Mohammad Khosravi:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

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

Acknowledgements

We thank Timo Roth for his support in parts of the numerical implementation. This work was supported in part by Stiftung Rieter and Rieter AG, Winterthur, by NCCR Automation, a National Centre of Competence in Research, funded by the Swiss National Science Foundation (grant number 51NF40_225155), and in part by the Swiss National Science Foundation under Postdoc. Mobility Grant 211104.

Supplementary material

Supplementary material associated with this article can be found in the online version at 10.1016/j.conengprac.2026.106904.

References

Balta, E. C., Barton, K., Tilbury, D. M., Rupenyan, A., & Lygeros, J. (2021). Learning-based repetitive precision motion control with mismatch compensation. In *2021 60th IEEE conference on decision and control (cdc)* (pp. 3605–3610).

Balula, S., Liao-Mcpherson, D., Rupenyan, A., & Lygeros, J. (2024). Data-driven reference trajectory optimization for precision motion systems. *Control Engineering Practice*, *144*, 105834.

Barton, K. L., & Alleyne, A. G. (2008). A cross-coupled iterative learning control design for precision motion control. *IEEE Transactions on Control Systems Technology*, *16*(6), 1218–1231.

Busetto, R., Lucchini, A., Formentin, S., & Savaresi, S. M. (2023). Data-driven optimal tuning of BLDC motors with safety constraints: A set membership approach, *IEEE/ASME Transactions on Mechatronics*,*28*(4), 1975–1983.

Catenaro, E., Aarnoudse, L., Formentin, S., & Oomen, T. (2024). Efficient tuning for motion control in diverse systems: A Bayesian framework. *IFAC-PapersOnLine*, *58*(15), 354–359.

Eriksson, D., Pearce, M., Gardner, J., Turner, R. D., & Poloczek, M. (2019). Scalable global optimization via local Bayesian optimization, in: *Advances in neural information processing systems*, H.Wallach, H.Larochelle, A.Beygelzimer, F.d'Alché-Buc, E.Fox, R.Garnett (Eds.), , Curran Associates, Inc, (*vol. 32*)

Forgione, M., Piga, D., & Bemporad, A. (2020). Efficient calibration of embedded MPC. *IFAC-PapersOnLine*, *53*(2), 5189–5194.

Gardner, J., Kusner, M., Zhixiang, K., Weinberger, J., & Cunningham (2014a). Bayesian optimization with inequality constraints. *International Conference on Machine Learning*, *32*, 937–945.

Gardner, J., Kusner, M., Zhixiang, K., Weinberger, J., & Cunningham (2014b). Bayesian optimization with inequality constraints, in: *International conference on machine learning* (*vol. 32*). Bejing, China. PMLR, pp. 937–945.

Garrido-Merchán, E. C., & Hernández-Lobato, D. (2020). Dealing with categorical and integer-valued variables in Bayesian optimization with Gaussian processes. *Neurocomputing*, *380*, 20–35.

Gharib, A., Stenger, D., Ritschel, R., & Voßwinkel, R. (2021). Multi-objective optimization of a path-following MPC for vehicle guidance: A Bayesian optimization approach. In *2021 European control conference (ECC)* (pp. 2197–2204).

Khosravi, M., Behrunani, V. N., Myszkorowski, P., Smith, R. S., Rupenyan, A., & Lygeros, J. (2021). Performance-driven cascade controller tuning with Bayesian optimization, *IEEE Transactions on Industrial Electronics*, *69*(1), 1032–1042.

Khosravi, M., Behrunani, V., Smith, R. S., Rupenyan, A., & Lygeros, J. (2020). Cascade control: Data-driven tuning approach based on Bayesian optimization. In *Ifac world congress 2020*.

Khosravi, M., Eichler, A., Schmid, N., Heer, P., & Smith, R. S. (2019a). Controller tuning by bayesian optimization an application to a heat pump. In *European control conference* (pp. 1467–1472).

Khosravi, M., König, C., Maier, M., Smith, R. S., Lygeros, J., & Rupenyan, A. (2022). Safety-aware cascade controller tuning using constrained bayesian optimization. *IEEE Transactions on Industrial Electronics*, *70*(2), 2128–2138.

Khosravi, M., Schmid, N., Eichler, A., Heer, P., & Smith, R. S. (2019b). Machine learning-based modeling and controller tuning of a heat pump, in: *Journal of physics: Conference series* (*vol. 1343*)(1). IOP Publishing, 012065.

König, C., Krishnadas, R., Balta, E. C., & Rupenyan, A. (2025). Adaptive Bayesian optimization for high-precision motion systems *IEEE Transactions on Automation Science and Engineering*,*22*, 15627–15637.

König, C., Turchetta, M., Lygeros, J., Rupenyan, A., & Krause, A. (2021). Safe and efficient model-free adaptive control via Bayesian optimization. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 9782–9788.

Lam, D., Manzie, C., & Good, M. (2010a). Model predictive contouring control. In *Conference on decision and control* (pp. 6137–6142).

Lam, D., Manzie, C., & Good, M. (2010b). Model predictive contouring control. In *49th ieee conference on decision and control* (pp. 6137–6142).

Li, J., Zagorowska, M., Pasquale, G. D., Rupenyan, A., & Lygeros, J. (2024). Safe time-varying optimization based on Gaussian processes with spatio-temporal kernel. *Advances in Neural Information Processing Systems*, *37*, 95326–95355.

Liniger, A., Varano, L., Rupenyan, A., & Lygeros, J. (2019). Real-time predictive control for precision machining, In *IEEE 58th conference on decision and control* (pp. 7746–7751).

Lu, Q., Kumar, R., & Zavala, V. M. (2020). MPC controller tuning using Bayesian optimization techniques. arXiv:2009.14175.

Mckay, M. D., Beckman, R. J., & Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, *42*(1), 55–61.

Nayebi, A., Munteanu, A., & Poloczek, M. (2019). A framework for Bayesian optimization in embedded subspaces. in: *Proceedings of the 36th international conference on machine learning*, ser. proceedings of machine learning K. Chaudhuri and R. Salakhutdinov, Eds., *97*, 4752–4761.

Paulson, J. A., Sorouifar, F., Laughman, C. R., & Chakrabarty, A. (2023). LSR-BO: Local search region constrained bayesian optimization for performance optimization of vapor compression systems. In *2023 American control conference (ACC)* (pp. 576–582). IEEE.

Piga, D., Forgione, M., Formentin, S., & Bemporad, A. (2019). Performance-oriented model learning for data-driven mpc design. *IEEE Control Systems Letters*, *3*(3), 577–582.

Qian, R., Luo, M., Zhao, J., & Li, T. (2016). Novel sliding mode control for ball screw servo system. In *MATEC Web of Conferences*, EDP Sciences, *54*, 03007.

Rasmussen, C. E. (2006). In Gaussian processes for machine learning. MIT Press.

Rupenyan, A., Khosravi, M., & Lygeros, J. (2021). Performance-based trajectory optimization for path following control using Bayesian optimization. In *2021 60th IEEE conference on decision and control (cdc)* (pp. 2116–2121). IEEE.

Savaia, G., Sohn, Y., Formentin, S., Panzani, G., Corno, M., & Savaresi, S. M. (2021). Experimental automatic calibration of a semi-active suspension controller via Bayesian optimization. *Control Engineering Practice*, *112*, 104826.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & Freitas, N. D. (2015). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175.

Sorourifar, F., Makrygirgos, G., Mesbah, A., & Paulson, J. A. (2021). A data-driven automatic tuning method for MPC under uncertainty using constrained Bayesian optimization. In *IFAC-PapersOnLine*, *54*, 243–250.

Vázquez, J. L., Brühlmeier, M., Liniger, A., Rupenyan, A., & Lygeros, J. (2020). Optimization-based hierarchical motion planning for autonomous racing. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 2397–2403).

Yang, S., Ghasemi, A. H., Lu, X., & Okwudire, C. E. (2015a). Pre-compensation of servo contour errors using a model predictive control framework. *International Journal of Machine Tools and Manufacture*, *98*, 50–60.

Yang, S., Ghasemi, A. H., Lu, X., & Okwudire, C. E. (2015b). Pre-compensation of servo contour errors using a model predictive control framework. *International Journal of Machine Tools and Manufacture*, *98*, 50–60.

Zagorowska, M., König, C., Yu, H., Balta, E. C., Rupenyan, A., & Lygeros, J. (2023). Efficient safe learning for controller tuning with experimental validation. arXiv:2310.17431.

Zagorowska, M., König, C., Yu, H., Balta, E. C., Rupenyan, A., & Lygeros, J. (2025). Efficient safe learning for controller tuning with experimental validation. *Engineering Applications of Artificial Intelligence*, *143*, 109894.