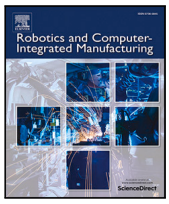
Robotics and Computer–Integrated Manufacturing 98 (2026) 103137

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

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



Full length article

Increasing dynamic accuracy of machine tools using predictive feedforward optimization with hybrid modeling

Haijia Xu [a,*], Christoph Hinze [a], Andrea Iannelli [b], Zexu Zhou [a], Alexander Verl [a]

[a] Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart, Seidenstr. 36, Stuttgart, 70174, Germany
[b] Institute for Systems Theory and Automatic Control, University of Stuttgart, Pfaffenwaldring 9, Stuttgart, 70569, Germany

ARTICLE INFO

Keywords:
Predictive feedforward
Hybrid modeling
Receding horizon optimization
Precision control

ABSTRACT

The paper presents an online optimization-based feedforward design framework using hybrid modeling to increase the dynamic accuracy of machine tools. Designed for use in dynamics simulation and feedforward compensation, the hybrid model combines a physics-based model of the multibody dynamics and a data-driven Gaussian process regressor of the output discrepancy. The feedforward control is based on the predictor–simulator separation, where the accurate but tractable nonlinear hybrid model is used for dynamics simulation, and the linearized predictor is adopted for optimal feedforward design with a receding horizon approach based on convex programming. This strategy allows the advanced modeling techniques to be used for real-time dynamics compensation in an open-loop fashion, where the associated convex optimization problem can be solved efficiently and reliably. We propose a methodological approach that covers the entire design procedure from dynamics modeling to control architecture selection and parameter tuning, providing an end-to-end strategy for practical applications. The algorithm is validated on a real-time industrial CNC machine, where the average computation time is 63 μs on an Intel i5 CPU. Compared to the industry standard baseline feedforward control, the proposed feedforward framework reduces the mean absolute contour error by 46.1% and 56.8% for constant velocity tracking and freeform butterfly path following, respectively. Even with a mismatch of 30 % in the model parameters, the presented feedforward still reduces the error by 38.5% compared to the baseline.

1. Introduction

Increasing the dynamic accuracy of machine tools is key to improving machining accuracy and, therefore, the economics of machine tool operation. As of today, practically every machine tool is controlled by model-free cascaded control, applied independently to each axis. By using dynamics models, the motion of the machine and its drives can be predicted more accurately. This prediction can be used for model-based control [1] and feedforward approaches [2] to improve the dynamic accuracy of feed drives and machine tools. At present, this is mainly achieved using analytical first-principles models [3–5]. Some effects are difficult to model analytically, such as motor torque ripple [6] and friction [7], which vary even between two instances of the same motor type and also depend on motor speed, load and temperature. In addition, the increase in modeling accuracy leads to an increase in the state dimension of the dynamics model, which increases the complexity of state estimation and system identification.

In recent years, machine learning approaches have become increasingly successful for machine monitoring [8–10] and dynamics simulation [11–13]. For precision applications, however, machine learning faces a scaling problem: the axes may move over a range of several meters, while the required machining precision is on the order of microns. For pure black box learning approaches, this leads to convergence problems where the micro behavior of the model is much harder to train due to the different problem scales.

Recent results incorporate prior knowledge into the learning process, where the dominant system behavior is captured by first-principles models and utilized for learning. These include transfer learning [14,15], where machine learning models are pre-trained with simulation data provided by a physics-based model, and then retrained with experimental data for refinement. Another approach is to integrate the physics-based model directly into the machine learning model structure [16–19], which filters the features of the learning targets and informs the machine learning model with the physics knowledge. Furthermore, the strategy of hybrid modeling transforms the learning task from the entire system characteristics into the prediction residual between the physics model and the true system [20–23]. However, while these advanced learning techniques allow accurate prediction of

* Corresponding author.
E-mail address: haijia.xu@isw.uni-stuttgart.de (H. Xu).

https://doi.org/10.1016/j.rcim.2025.103137
Received 13 November 2024; Received in revised form 21 May 2025; Accepted 6 September 2025
Available online 16 September 2025

system behavior, their application to increasing the dynamic accuracy of machine tools, such as feedforward compensation, which requires the inverse of the model from output to input [2], remains unclear.

Several possible inversion-based approaches have been investigated to integrate the learning-based models into feedforward compensation. Inverse learning captures the inverse dynamics of the systems and directly searches for the optimal feedforward under a given reference, such as the deep-neural-network-based inversion [24], physics-guided inverse neural networks [25,26], to name a few. Another possible approach is to limit the complexity at the modeling stage, and to take a differentially flat analytical model as basic structure [2,27]. The data-driven methods are used to learn the residual, while the overall extended hybrid model still remains invertible or can be compensated via an additive term of the control signal [28]. However, their common drawback is that they rely on the assumption of a differentially flat physics model (directly invertible) to provide the basic structural information for dynamics inversion. This means that the mechanics are assumed to exhibit only rigid body motion, which does not hold especially for multi-axis CNC machines with compliant mechanics [1, §7.2].

Optimization-based methods that approximately solve optimal control problems have also received special attention due to their inherent ability to take advantage of precise dynamics models for feedforward optimization. The optimal control problem can be solved offline for motion planning tasks subject to a fixed Ref. [29], which is not suitable for CNC machine applications, where the trajectories are determined in real time by the CNC kernel. Another possibility is to use model predictive control (MPC) [30,31], where the optimization problem is solved online and is able to deal with the changing setpoints in real time. These include predictive seam tracking [32], online trajectory optimization [33,34], and contour error compensation [35], to name a few. However, the non-convex optimization problem associated with the nonlinear dynamics model is practically intractable in real-time execution on CNC machines (sampling time $\lesssim$1 ms). In addition, because the dynamics of machine tools are often subject to variations over their lifetime, the robustness of MPC with a nonlinear learning-based model still remains an open issue for industrial practice [31]. It is thus clear that whereas machine learning and optimization techniques have attracted much attention in the field, work on increasing the dynamics accuracy of machine tools with hybrid modeling in real-time applications is still scarce.

The feedforward framework in this paper is purely open-loop, allowing the compensation scheme to be calculated directly from the reference signal in the CNC or inverter. Also, this approach emphasizes the separate use of an accurate but complicated simulator from a linearized predictor, and allows the efficient use of hybrid modeling for error compensation in an open-loop feedforward fashion. Unlike works describing the full system dynamics with learning-based models [14,36], our presented hybrid modeling approach captures the dominant dynamics using a multibody model based on first principles, which will be called *physics-based model* throughout the paper in accordance to the data-driven modeling literature, cf. [15,37]. This ensures that the entire hybrid model stays close to the true system based on the prior knowledge and generalizes well to unseen dataset. Also, in contrast to the work that directly adopts the nonlinear dynamics model for online optimization [33,36], our presented separation strategy takes the nonlinear hybrid model for dynamics simulation and the linearized model for control optimization. By leveraging the technique of receding horizon optimization with convex programming, the associated optimal feedforward control problem is solved very efficiently and reliably, thereby enabling real-time capability even when integrating machine learning components. The main contribution of this paper can be summarized as follows:

1. Feedforward control-oriented hybrid modeling strategy of industrial machines combining physics-based multibody dynamics and data-driven discrepancy.
2. Online optimization-based feedforward design with convex quadratic program (QP) formulation via separation of hybrid simulator and linearized predictor.
3. Open-loop design of the compensation scheme, allowing calculation as a separate function block in the inverter or CNC without changing the standard control structure.
4. Practical guides to implementation on industrial machines, covering model identification, hyperparameter tuning, and embedded numerical optimization.
5. Validated real-time capability and performance improvement on a multi-axis CNC machine, with experimental data openly available in [38].

The remaining part is organized as follows: Section 2 introduces the architecture of the proposed feedforward framework. Section 3 proposes the hybrid modeling structure of the drive control loop, combining the physics-based multibody dynamics and the data-driven discrepancy. Section 4 proposes the optimization-based feedforward control design approaches. Section 5 provides guidelines on practical implementation including gain selection and numerical optimization. Section 6 presents an experimental validation of the proposed predictive feedforward scheme on industrial hardware. Section 7 gives the concluding remarks.

2. The architecture of the proposed method

The control architecture of the proposed predictive feedforward framework, applied to the independent axis control of a machine tool, is illustrated in Fig. 1. The drive dynamics, governed by a proportional–integral (PI) velocity controller, are represented by a nonlinear hybrid model that combines multibody dynamics (physics-based model) with a data-driven output discrepancy model. An open-loop linearization-based MPC feedforward scheme is used to compute the optimal control input online, steering the simulator to the reference trajectory. This optimized control signal is applied to the real system as an optimal feedforward input. The current state for the MPC scheme is simulated by the simulation model, while the future states within the optimization horizon are predicted by the linearized predictor based on the state of the simulator. The predictive feedforward control does not rely on feedback from the real system; instead, it depends only on the reference trajectory (e.g., from the CNC). As a result, it cannot destabilize the system and can be implemented as a separate compensation function block in the CNC or frequency inverter.

3. Hybrid modeling of drive control loop

3.1. Physics-based model of machine tool motion dynamics

The major motion dynamics of a machine tool with n_j axes can be captured by using a continuous-time multibody model, see Fig. 2 for the studied machine tool. To capture the compliant behavior of the axes and possibly the gearboxes, we distinguish between the measured angle of the motor shaft of each axis $\tilde{\boldsymbol{\theta}} \subset \mathbb{R}^{n_\mathrm{j}}$ and the load-side axis positions $\boldsymbol{q} \in \mathbb{R}^{n_\mathrm{j}}$, measured with a set of secondary encoders. For ease of notation and to scale the model uniformly, we also use the motor position converted to load-side coordinates $\boldsymbol{\theta} := \boldsymbol{U}\tilde{\boldsymbol{\theta}}$, where $\boldsymbol{U}$ is a diagonal matrix with $\{\boldsymbol{U}\}_{j,j} \ll 1$ containing the transmission ratio from the motor angle to the load-side coordinate of axis j. The multibody system dynamics of the studied machine tool (see e.g., [39]) can be formulated as the model given by

$$\underbrace{\begin{bmatrix} \boldsymbol{M}_\theta & \boldsymbol{M}_{\theta q} \\ \boldsymbol{M}_{\theta q}^\top & \boldsymbol{M}_q \end{bmatrix}}_{=:\boldsymbol{M}} \begin{bmatrix} \ddot{\boldsymbol{\theta}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{bmatrix} + \begin{bmatrix} \boldsymbol{\tau}_\mathrm{c} + \boldsymbol{U}^{-1}\boldsymbol{\tau}_\mathrm{f,m} \\ -\boldsymbol{\tau}_\mathrm{c} + \boldsymbol{\tau}_\mathrm{f,l} \end{bmatrix} = \begin{bmatrix} \boldsymbol{U}^{-1}\boldsymbol{\tau}_\mathrm{m} \\ \boldsymbol{0} \end{bmatrix}. \tag{1}$$

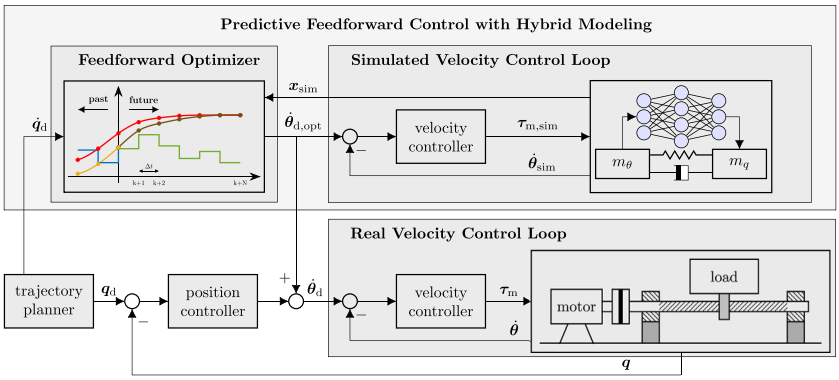


Fig. 1. Proposed optimization-based feedforward control framework with hybrid modeling applied to independent axis control.

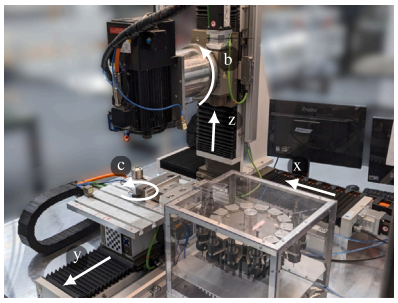


Fig. 2. Five-axis milling machine as test bench.

The upper part of the equations of motion describes the motor dynamics with motor friction $\boldsymbol{\tau}_{\mathrm{f,m}}$ (e.g. $\boldsymbol{\tau}_{\mathrm{f,m}} := \boldsymbol{F}_c \operatorname{sgn}(\dot{\boldsymbol{\theta}}) + \boldsymbol{F}_v \dot{\boldsymbol{\theta}}$ with Coulomb and viscous friction) and are driven by motor torques $\boldsymbol{\tau}_m$. The lower part describes the dynamics of the coupled multibody system as in classical robotics, where Coriolis, centrifugal and gravity terms are summarized in the nonlinear vector $\boldsymbol{n}$ with an additional load-side friction term $\boldsymbol{\tau}_{\mathrm{f,l}}$. Compliant coupling between motor and load coordinates of each axis is captured by substituting the compliance force for $\boldsymbol{\tau}_{\mathrm{c}}$. For the remainder of the paper, we use linear stiffness and damping as

$$\boldsymbol{\tau}_{\mathrm{c}} = \boldsymbol{K}(\boldsymbol{q} - \boldsymbol{\theta}) + \boldsymbol{D}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{\theta}}), \tag{2}$$

with diagonal stiffness matrix $\boldsymbol{K} = \operatorname{diag}\left(\left[k_1, \dots, k_{n_{\mathrm{j}}}\right]\right)$ and $\boldsymbol{D} = \operatorname{diag}\left(\left[d_1, \dots, d_{n_{\mathrm{j}}}\right]\right)$. The motor-side and load-side inertias are described by the diagonal matrices $\boldsymbol{M}_\theta = \operatorname{diag}\left(\left[m_{\theta_1}, \dots, m_{\theta_{n_{\mathrm{j}}}}\right]\right)$ and $\boldsymbol{M}_q = \operatorname{diag}\left(\left[m_q, \dots, m_{q_{n_{\mathrm{j}}}}\right]\right)$, respectively. The inertia coupling between the motor and load is neglected (i.e., $\boldsymbol{M}_{\theta q} = \boldsymbol{0}$) for simplicity. Also, external process forces due to the contact of the tool with the environment are not included. Considering them for feedforward compensation would require additional sensors or estimation of Cartesian contact forces $\boldsymbol{h}_{\mathrm{e}}$, which can be added to the load-side part as $\boldsymbol{J}^{\top}(\boldsymbol{q})\boldsymbol{h}_{\mathrm{e}}$ via the machine's Jacobian $\boldsymbol{J}^{\top}(\boldsymbol{q})$.

Using the state vector $\tilde{\boldsymbol{x}} = \begin{bmatrix}\boldsymbol{\theta}^{\top} & \boldsymbol{q}^{\top} & \dot{\boldsymbol{\theta}}^{\top} & \dot{\boldsymbol{q}}^{\top}\end{bmatrix}^{\top}$, the multibody dynamics model (1) can be written in state-space form as

$$\dot{\tilde{\boldsymbol{x}}} = \begin{bmatrix} \boldsymbol{0}_{2n_{\mathrm{j}}} & \boldsymbol{I}_{2n_{\mathrm{j}}} \\ -\boldsymbol{M}^{-1}\bar{\boldsymbol{K}} & -\boldsymbol{M}^{-1}\bar{\boldsymbol{D}} \end{bmatrix} \tilde{\boldsymbol{x}} + \begin{bmatrix} \boldsymbol{0}_{2n_{\mathrm{j}}\times 1} \\ \boldsymbol{M}^{-1}\begin{bmatrix} -\boldsymbol{U}^{-1}\boldsymbol{\tau}_{\mathrm{f,m}} \\ \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{\tau}_{\mathrm{f,l}} \end{bmatrix} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{2n_{\mathrm{j}}\times n_{\mathrm{j}}} \\ \boldsymbol{M}^{-1}\begin{bmatrix} \boldsymbol{U}^{-1} \\ \boldsymbol{0}_{n_{\mathrm{j}}} \end{bmatrix} \end{bmatrix} \boldsymbol{\tau}_{\mathrm{m}}, \tag{3}$$

where the stiffness and damping matrices are given by

$$\bar{\boldsymbol{K}} := \begin{bmatrix} \boldsymbol{K} & -\boldsymbol{K} \\ -\boldsymbol{K} & \boldsymbol{K} \end{bmatrix}, \quad \bar{\boldsymbol{D}} := \begin{bmatrix} \boldsymbol{D} & -\boldsymbol{D} \\ -\boldsymbol{D} & \boldsymbol{D} \end{bmatrix}$$

and the motor torque vector $\boldsymbol{\tau}_{\mathrm{m}}$ as input. The dynamics of the motor current control loop are neglected (assuming commanded $\boldsymbol{\tau}_{\mathrm{m,d}}$ equals actual $\boldsymbol{\tau}_{\mathrm{m}}$), as their dynamics are much faster in practice (>factor 5 [1, §7]) than the achievable bandwidth of the mechanics.

3.2. Extended state-space model of velocity control loop

Practically all industrial motor drives of each axis in a machine tool use proportional–integral (PI) controllers for speed control. A key factor of the presented architecture is the modeling of the mechanics with PI velocity controllers and the use of the velocity controlled plant for feedforward design (*loop inversion* [2, §4.2]), which provides the following main features

1. The PI controller reduces the influence of nonlinearity on the identification such as nonlinear friction and pose-dependent stiffness, which are typical of ball screw drives (the most common form of linear feed drive) [1].
2. The PI controller regulates not only the physical system to the reference trajectory, but also the simulator and optimization components, even under model and numerical errors [2, §4.2].

These properties ensure that an accurate simulation model with an extended PI controller can be obtained with less identification effort, and does not diverge from the physical control system in real-time feedforward applications, even if no measurement is taken for feedforward, as shown in Section 6.2.

The PI velocity controller determines the commanded motor torque $\boldsymbol{\tau}_{\mathrm{m,d}}$ using the velocity error $\boldsymbol{e}_v$ between the commanded velocity $\dot{\boldsymbol{\theta}}_{\mathrm{d}}$ and the motor velocity $\dot{\boldsymbol{\theta}}$, given by

$$\boldsymbol{\tau}_{\mathrm{m,d}} = \boldsymbol{K}_p \left(\dot{\boldsymbol{\theta}}_{\mathrm{d}} - \dot{\boldsymbol{\theta}} + \boldsymbol{K}_i \int \left(\dot{\boldsymbol{\theta}}_{\mathrm{d}} - \dot{\boldsymbol{\theta}}\right) \mathrm{d}t \right), \tag{4}$$

with proportional gains $\boldsymbol{K}_p = \operatorname{diag}\left(\left[k_{p,1}, \dots, k_{p,n_{\mathrm{j}}}\right]\right)$ and integral gains $\boldsymbol{K}_i = \operatorname{diag}\left(\left[k_{i,1}, \dots, k_{i,n_{\mathrm{j}}}\right]\right)$ of each axis, which are exactly known, as they can be extracted from the frequency inverter. Note that the error integral $\boldsymbol{e}_{\mathrm{m}} = \int (\dot{\boldsymbol{\theta}}_{\mathrm{d}} - \dot{\boldsymbol{\theta}})\,\mathrm{d}t$ is not equal to the motor position error, since the actual commanded velocity $\dot{\boldsymbol{\theta}}_{\mathrm{d}}$ is the combination of velocity feedforward and feedback signal of position controller, which is different from the desired velocity value $\dot{\boldsymbol{q}}_{\mathrm{d}}$ given by trajectory planner, see Section 4.2.

$$\dot{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \boldsymbol{0}_{2n_\mathrm{j}} & \boldsymbol{I}_{2n_\mathrm{j}} & \boldsymbol{0}_{2n_\mathrm{j}\times n_\mathrm{j}} \\ -\boldsymbol{M}^{-1}\begin{bmatrix} \bar{\boldsymbol{K}} & \bar{\boldsymbol{D}}+\begin{bmatrix} \boldsymbol{U}^{-1}\boldsymbol{K}_p & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} & \begin{bmatrix} -\boldsymbol{U}^{-1}\boldsymbol{K}_p\boldsymbol{K}_i \\ \boldsymbol{0}_{n_\mathrm{j}} \end{bmatrix} \end{bmatrix} \\ \boldsymbol{0}_{n_\mathrm{j}\times 2n_\mathrm{j}} & \begin{bmatrix} -\boldsymbol{I}_{n_\mathrm{j}} & \boldsymbol{0}_{n_\mathrm{j}} \end{bmatrix} & \boldsymbol{0}_{n_\mathrm{j}} \end{bmatrix}}_{=:\tilde{f}_c(\boldsymbol{x})} \boldsymbol{x} + \begin{bmatrix} \boldsymbol{0}_{2n_\mathrm{j}\times 1} \\ \boldsymbol{M}^{-1}\begin{bmatrix} -\boldsymbol{U}^{-1}\boldsymbol{\tau}_{\mathrm{f,m}} \\ \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}})+\boldsymbol{\tau}_{\mathrm{f,l}} \end{bmatrix} \\ \boldsymbol{0}_{n_\mathrm{j}\times 1} \end{bmatrix}$$

$$+ \underbrace{\begin{bmatrix} \boldsymbol{0}_{2n_\mathrm{j}\times n_\mathrm{j}} \\ \boldsymbol{M}^{-1}\begin{bmatrix} \boldsymbol{U}^{-1}\boldsymbol{K}_p \\ \boldsymbol{0}_{n_\mathrm{j}} \end{bmatrix} \\ \boldsymbol{I}_{n_\mathrm{j}} \end{bmatrix}}_{=:\tilde{g}_c(\boldsymbol{x})} \underbrace{\dot{\boldsymbol{\theta}}_\mathrm{d}}_{=:\boldsymbol{u}\subset\mathbb{R}^{n_u}} \tag{6a}$$

$$\boldsymbol{y} = \tilde{\boldsymbol{h}}(\boldsymbol{x}). \tag{6b}$$

Box I.

A state-space formulation of the PI velocity controller is obtained by assuming $\boldsymbol{\tau}_{\mathrm{m,d}} \approx \boldsymbol{\tau}_\mathrm{m}$ and introducing a new state of the integrated error as

$$\dot{\boldsymbol{e}}_\mathrm{m} = \dot{\boldsymbol{\theta}}_\mathrm{d} - \dot{\boldsymbol{\theta}} \qquad \text{(motor speed error)} \tag{5a}$$

$$\boldsymbol{\tau}_\mathrm{m} = \boldsymbol{K}_p\left(\dot{\boldsymbol{e}}_\mathrm{m} + \boldsymbol{K}_i\boldsymbol{e}_\mathrm{m}\right) \quad \text{(velocity controller output)} \tag{5b}$$

and, hence, the PI-controlled input-affine state-space formulation with model state $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{\theta}^\top & \boldsymbol{q}^\top & \dot{\boldsymbol{\theta}}^\top & \dot{\boldsymbol{q}}^\top & \boldsymbol{e}_\mathrm{m}^\top\end{bmatrix}^\top \subset \mathbb{R}^{n_x}$, $n_x = 5n_\mathrm{j}$ is given by the Eqs. (6a) and (6b) in Box I.

The measurement equations are directly the load-side axis velocities in the ideal case, i.e. $\tilde{\boldsymbol{h}}(\boldsymbol{x}) = \dot{\boldsymbol{q}} = \begin{bmatrix}\boldsymbol{0}_{n_\mathrm{j}\times 3n_\mathrm{j}} & \boldsymbol{I}_{n_\mathrm{j}} & \boldsymbol{0}_{n_\mathrm{j}}\end{bmatrix}\boldsymbol{x}$, but may contain slight nonlinearity in the measurement systems.

3.3. Model discretization

Finally, to use the model in a data-driven framework, the state-space Eqs. (6a) are discretized with a fixed step size Δt equal to the sampling time of the velocity controller. This has to be done taking into account the dominant frequencies of the model and the chosen step size to ensure numerical stability of the integration scheme. For the presented group of mechanical systems and sampling frequency above 1 kHz, a 4th order Runge–Kutta integration [40] is usually sufficient, resulting in an analytical model in discrete-time form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{g}(\boldsymbol{x}_k)\boldsymbol{u}_k \qquad \text{(analytical state equations)} \tag{7a}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k) \qquad \text{(analytical output equations)} \tag{7b}$$

with inputs $\boldsymbol{u}_k = \dot{\boldsymbol{\theta}}_\mathrm{d}(k\,\Delta t) \subset \mathbb{R}^{n_u}$ being assumed to be piecewise constant between time steps and outputs $\boldsymbol{y}_k = \dot{\boldsymbol{q}}(k\,\Delta t) \subset \mathbb{R}^{n_y}$.

3.4. Learning model discrepancy of the control loop

By leveraging machine learning methods, we enhance the physics-based model (7) by incorporating regressors that account for output discrepancies, thereby improving prediction accuracy of the input–output behavior. The output function extended by discrepancy learning is formulated as

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{\Phi}_{\boldsymbol{y}}(\boldsymbol{x}_k), \qquad \text{(augmented output)} \tag{8}$$

where the state equations stay the same as in (7a). This approach ensures that the system's input–output behavior is more accurately predicted by combining the analytical model with the learned discrepancy model. As shown in Fig. 3, the regressors $\boldsymbol{\Phi}_{\boldsymbol{y}}$ are trained to $\boldsymbol{\Phi}_{\boldsymbol{y}} \to \boldsymbol{\xi}_k := \boldsymbol{y}_{k,\mathrm{meas}} - \boldsymbol{y}_k$ using the output error, i.e. the difference between measured output $\boldsymbol{y}_{k,\mathrm{meas}}$ and simulated output $\boldsymbol{y}_k$. Other possibilities to train regressors would be state augmentation (replacing $\boldsymbol{f}(\boldsymbol{x}_k)$ by $\boldsymbol{f}(\boldsymbol{x}_k)+\boldsymbol{\Phi}_{\boldsymbol{x}}$) or input function augmentation (replacing $\boldsymbol{g}(\boldsymbol{x}_k)$ by $\boldsymbol{g}(\boldsymbol{x}_k)+\boldsymbol{\Phi}_{\boldsymbol{u}}$). To focus on the demonstration of the proposed feedforward framework, we only consider the output regressors in this work, see [37] for a more detailed comparison.

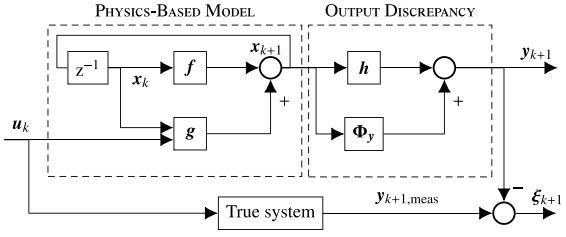


Fig. 3. Signal flow of the prediction using physics-based model (7) with chosen output discrepancy augmentation.

Various functional approximation methods would work here as a regressor of $\boldsymbol{\Phi}_{\boldsymbol{y}}$, e.g. neural networks, regression trees, to name a few. In this work, we use Gaussian Processes (GP) as regressors, because they can be learned from small data sets and generalize well to unseen data with guaranteed smoothness [41]. GP also fall back to their prior in regions far away from their training data, i.e., the output augmentation returns $\boldsymbol{0}$ in regions, where there is not enough training data.

The input for GP training is the vector-valued position and velocity $\tilde{\boldsymbol{q}}_j = [\{\boldsymbol{q}\}_j, \{\dot{\boldsymbol{q}}\}_j]^\top$, $j = 1,\dots,n_\mathrm{j}$ for each axis, which is part of the state vector of (7a). The output of GP model is the corresponding element of the discrepancy vector $\{\boldsymbol{\xi}_k\}_j$, for which the Eq. (7) is simulated to generate the output sequence of the physics-based model, as shown in Fig. 3. The GP model for each axis j is independently formulated as

$$\{\boldsymbol{\xi}_k\}_j = \boldsymbol{\Phi}_{\boldsymbol{y},j}(\tilde{\boldsymbol{q}}_j) + \varepsilon_j, \quad j = 1,\dots,n_\mathrm{j}, \quad \varepsilon_j \in \mathcal{N}(0,\sigma_j^2), \tag{9}$$

where ε_j is the remaining Gaussian noise with variance σ_j^2, assuming that $\boldsymbol{\Phi}_{\mathrm{y},j}$ is a universal function approximator. With the training data set $\tilde{\boldsymbol{Q}}_j = [\tilde{\boldsymbol{q}}_{j,1},\dots,\tilde{\boldsymbol{q}}_{j,n_D}]$ and $\boldsymbol{\xi}_j = [\xi_{j,1},\dots,\xi_{j,n_D}]$ of length n_D. The prediction of $\boldsymbol{\Phi}_{\boldsymbol{y}}(\boldsymbol{x})$ at an arbitrary test input $\tilde{\boldsymbol{q}}_j$ is given by the posterior mean and variance

$$\mathrm{E}[\boldsymbol{\Phi}_{\boldsymbol{y},j}(\tilde{\boldsymbol{q}}_j)] = k(\tilde{\boldsymbol{q}}_j,\tilde{\boldsymbol{Q}}_j)^\top \underbrace{(k(\tilde{\boldsymbol{Q}}_j,\tilde{\boldsymbol{Q}}_j)+\sigma_j^2)^{-1}\{\boldsymbol{\xi}_k\}_j}_{\coloneqq\boldsymbol{\beta}_j} \tag{10}$$

$$\sigma^2[\boldsymbol{\Phi}_{\boldsymbol{y},j}(\tilde{\boldsymbol{q}}_j)] = k(\tilde{\boldsymbol{q}}_j,\tilde{\boldsymbol{q}}_j) - k(\tilde{\boldsymbol{q}}_j,\tilde{\boldsymbol{Q}}_j)^\top (k(\tilde{\boldsymbol{Q}}_j,\tilde{\boldsymbol{Q}}_j)+\sigma_j^2)^{-1} k(\tilde{\boldsymbol{q}}_j,\tilde{\boldsymbol{Q}}_j). \tag{11}$$

The GP model in (10) and (11) allows approximating functions with scalar outputs, as shown in (9), while neglecting the correlation between model outputs. This premise is satisfied by the discrepancy signals when applied to the independent axis control, where cross-axis correlations are negligible. The kernel function $k(\cdot,\cdot)$ measures the similarity in input space, which is given by a squared exponential kernel here

$$k_{\mathrm{SE}}(\boldsymbol{x},\boldsymbol{x}') = \sigma_{\mathrm{S}}^2 \exp\left(\|\boldsymbol{L}^{-1}(\boldsymbol{x}-\boldsymbol{x}')\|^2/2\right) \tag{12}$$

where σ_{S}^2 is the signal variance determining the expected distance of $\boldsymbol{\Phi}_{\boldsymbol{y},j}$ to its mean value. The length scale $\boldsymbol{L} = \mathrm{diag}\left(l_1, l_2\right)$ captures the correlation of neighboring points in each dimension of the input space, influencing the width of the kernel [41].

4. Predictive feedforward optimization

The principle of the presented Model Predictive Feedforward Control (MPFFC) is to solve a linear tracking MPC problem repeatedly online. The control input computed from the optimization problem steers the nonlinear hybrid model to the given reference trajectory, and is used as the optimal feedforward control of the real control system. The key ingredient is the distinct use of the model used internally by the MPC scheme for prediction (*predictor*) and the nonlinear hybrid model used for dynamics simulation (*simulator*). This allows the more accurate but tractable hybrid model to be used for feedforward design in a computationally efficient and reliable way through the repeated online solution of a convex optimization problem.

4.1. Linearization and correction

Considering the hybrid model (7a), (8) for the state $\boldsymbol{x}_k$ at time step t_k, we define the linearized dynamics model throughout the length of the prediction horizon N as

$$\boldsymbol{A}_k := \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_k}, \qquad \boldsymbol{B}_k := \boldsymbol{g}(\boldsymbol{x}_k), \qquad \boldsymbol{C}_k := \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_k}, \tag{13}$$

which leads to the linearized prediction model at time step $t_k = k\Delta t, k \in \mathbb{N}_0$ with prediction step $i \in \{0,\dots,N\}$

$$\boldsymbol{x}_{i+1|k} = \boldsymbol{A}_k \boldsymbol{x}_{i|k} + \boldsymbol{B}_k \boldsymbol{u}_{i|k} + \boldsymbol{\delta}_{\boldsymbol{x},k} \quad \text{(dynamics)} \tag{14a}$$

$$\boldsymbol{y}_{i|k} = \boldsymbol{C}_k \boldsymbol{x}_{i|k} + \boldsymbol{\delta}_{\boldsymbol{y},i|k} \quad \text{(output)} \tag{14b}$$

where the initial condition $\boldsymbol{x}_{0|k}$ of the linear predictor (14) is set to the simulated state $\boldsymbol{x}_{k,\mathrm{sim}}$ of the nonlinear hybrid model (7a), (8) at each time step k. Note that the linearization is around the current state $\boldsymbol{x}_{k,\mathrm{sim}}$ and does not require future states or current input due to the input affine formulation. The additive correction terms $\boldsymbol{\delta}_{\boldsymbol{x},k}$ and $\boldsymbol{\delta}_{\boldsymbol{y},i|k}$ account for the known model mismatch between the simulator and the linearized predictor, and the output discrepancy captured by the data-driven regressor. The state correction term $\boldsymbol{\delta}_{\boldsymbol{x},k}$ is approximated as constant during the prediction horizon and is calculated as

$$\boldsymbol{\delta}_{\boldsymbol{x},k} = \boldsymbol{f}(\boldsymbol{x}_{0|k}) - \boldsymbol{A}_k \boldsymbol{x}_{0|k}. \tag{15}$$

Similarly, the output correction term $\boldsymbol{\delta}_{\boldsymbol{y},i|k}$ with $i \in \{1,\dots,N-1\}$ is given by

$$\boldsymbol{\delta}_{\boldsymbol{y},i|k} = \boldsymbol{h}(\boldsymbol{x}_{0|k}) - \boldsymbol{C}_k \boldsymbol{x}_{0|k} + \boldsymbol{\Phi}_{\boldsymbol{y},i|k}, \quad i \in \{0,\dots,N\}. \tag{16}$$

The term $\boldsymbol{h}(\boldsymbol{x}_{0|k}) - \boldsymbol{C}_k \boldsymbol{x}_{0|k}$ accounts for the output linearization error, which is assumed to be constant inside the prediction horizon. The discrepancy model depends on the state, which is only given at the prediction step $i = 0$ but is unknown for the future steps within the prediction horizon. To get a more accurate prediction than choosing a constant value, the discrepancy model is linearized using a first-order Taylor expansion of $\boldsymbol{\Phi}_{\boldsymbol{y},i}$ w.r.t. time, given by

$$\boldsymbol{\Phi}_{\boldsymbol{y},i|k} \approx \boldsymbol{\Phi}_{\boldsymbol{y}}(\boldsymbol{x}_{0|k}) + i\Delta t \left.\frac{\partial \boldsymbol{\Phi}_{\boldsymbol{y}}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{0|k}} \dot{\boldsymbol{x}}_{0|k}, \quad i \in \{0,\dots,N\}, \tag{17}$$

which similarly captures the information of the known derivative and avoids any dependence on future states. Note, that $\boldsymbol{x}_{0|k}$ is obtained from the simulated state, $\dot{\boldsymbol{x}}_{0|k}$ is obtained by replacing the commanded velocity with its desired value $\dot{\boldsymbol{\theta}}_{\mathrm{d}} = \dot{\boldsymbol{q}}_{\mathrm{d}}$ in Eq. (6a). This linearization strategy reduces the computation of the GP model from N to two evaluations at each time step (GP and its derivative), which allows embedding the GP model in the optimization-based feedforward design at high sampling frequencies ($\gg$1 kHz, see Section 6.1).

4.2. MPFFC setup

At each time step t_k, the predictive controller searches for the optimal control action to steer the nonlinear hybrid simulator to the reference output trajectory, which also achieves the optimal feedforward control for the real control system using the hybrid model in an optimization-based way.

Problem 1. Given the state of the simulator $\boldsymbol{x}_{k,\mathrm{sim}}$ as well as the linearization of the nonlinear dynamics (14), the optimal control problem underlying the MPFFC scheme at time step k is as follows

$$\begin{aligned}
\min_{\boldsymbol{x}_{i|k},\boldsymbol{u}_{i|k},\boldsymbol{y}_{i|k}} \; & \sum_{i=0}^{N-1} \left(\left\|\boldsymbol{y}_{i|k} - \boldsymbol{y}^{\mathrm{r}}_{k+i}\right\|^2_{\boldsymbol{Q}} + \left\|\boldsymbol{u}_{i|k} - \boldsymbol{u}^{\mathrm{r}}_{k+i}\right\|^2_{\boldsymbol{R}} \right) \\
& + \left\|\boldsymbol{y}_{N|k} - \boldsymbol{y}^{\mathrm{r}}_{k+N}\right\|^2_{\boldsymbol{Q}_f} \qquad (18) \\
\text{s.t.} \quad & \boldsymbol{x}_{0|k} = \boldsymbol{x}_{k,\mathrm{sim}} && \text{(initial condition)} \\
& \boldsymbol{x}_{i+1|k} = \boldsymbol{A}_k \boldsymbol{x}_{i|k} + \boldsymbol{B}_k \boldsymbol{u}_{i|k} + \boldsymbol{\delta}_{\boldsymbol{x},k} && \text{(dynamics)} \\
& \boldsymbol{y}_{i|k} = \boldsymbol{C}_k \boldsymbol{x}_{i|k} + \boldsymbol{\delta}_{\boldsymbol{y},i|k} && \text{(output)} \\
& \boldsymbol{x}_{i|k} \in \mathcal{X}, \quad i \in \{1,\dots,N\} && \text{(state constraints)} \\
& \boldsymbol{u}_{i|k} \in \mathcal{U}, \quad i \in \{0,\dots,N-1\}. && \text{(input constraints)}
\end{aligned}$$

Within the prediction horizon N, the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are the weights for the tracking error and input deviation, respectively, where a separate terminal weight $\boldsymbol{Q}_f$ is used for the last tracking error to improve the tracking performance without introducing terminal constraints [31]. The reference output $\boldsymbol{y}^{\mathrm{r}}$ and the reference input $\boldsymbol{u}^{\mathrm{r}}$ can be obtained by the trajectory planner (i.e. the desired axis velocities $\boldsymbol{q}_{\mathrm{d}}$ from the CNC).

The optimization variables are the states $\boldsymbol{x}_{i|k}$, the inputs $\boldsymbol{u}_{i|k}$ and the outputs $\boldsymbol{y}_{i|k}$. In order to write the optimization problem in convex quadratic programming (QP) form, which can be solved efficiently, the input and state constraints are assumed to be polytopic (in the form of linear inequalities). Possible applications include describing box constraints, or limiting the change rate of variables. One result of the optimization problem is the sequence of optimal inputs $\boldsymbol{U}_k^*$, of which only the first entry $\boldsymbol{u}_k^* = \{\boldsymbol{U}_k^*\}_1$ is used in each iteration. This gives the calculation of MPFFC summarized in Algorithm 1.

4.3. Offset-free output tracking with integral action

It is well known that MPC in general cannot guarantee an offset-free output tracking behavior due to model mismatch, leading to biased feedforward commands [42]. Such a model mismatch can be expected with the proposed framework, since the optimization model is linearized and, furthermore, the optimization can be stopped early to meet the time requirements of online computation. To achieve the required

Algorithm 1 Predictive Feedforward Control Scheme.

For each time step k

1. Compute the linearized dynamics matrices $\boldsymbol{A}_k, \boldsymbol{B}_k, \boldsymbol{C}_k$ (13) and correction terms $\boldsymbol{\delta}_{\boldsymbol{x},k}, \boldsymbol{\delta}_{\boldsymbol{y},i|k}$ (15), (16).
2. Set the simulated current state as initial state for the linearized model as $\boldsymbol{x}_{0|k} = \boldsymbol{x}_{k,\mathrm{sim}}$ and solve problem (18).
3. Apply the optimized control $\boldsymbol{u}_k^*$ to nonlinear simulator (7a), (8).
4. Apply the optimized control $\boldsymbol{u}_k^*$ to real control system as an additive feedforward.

offset-free tracking behavior despite possible model uncertainties [43], the integrated output tracking error is approximated as

$$\boldsymbol{d}_k = \int_{t_0}^{t_k} (\boldsymbol{y}_{\mathrm{d}}(\tau) - \boldsymbol{y}(\tau))\mathrm{d}\tau \approx \boldsymbol{d}_{k-1} + (\boldsymbol{y}_{\mathrm{d},k-1} - \boldsymbol{y}_{k-1}) \cdot \Delta t \tag{19}$$

and added to the output correction term $\boldsymbol{\delta}_{\boldsymbol{y},k}$ of the linearized predictor in (16), given by

$$\boldsymbol{\delta}_{\boldsymbol{y},i|k} = \left(\boldsymbol{h}(\boldsymbol{x}_{0|k}) + \boldsymbol{\Phi}_{\boldsymbol{y},i|k}\right) - \boldsymbol{C}_k \boldsymbol{x}_{0|k} + k_{\mathrm{int}} \boldsymbol{d}_k, \tag{20}$$

where k is the current time step. The only term varying over the prediction horizon is still the linear approximation of the GP $\boldsymbol{\Phi}_{\boldsymbol{y},i|k}$. Since the inputs and outputs of the feedforward model are both velocities, the integral gain k_{int} [1/s] is the inverse of the convergence time of the integral disturbance suppression, which can be directly prescribed (e.g. to the bandwidth of the velocity control loop). In addition, the integrator also compensates for drifts in the simulator states, e.g. due to gravity, which would cause the simulated PI controller to counteract and result in non-zero velocity command.

5. Practical guidelines on system identification and feedforward optimization

5.1. Identification of dynamics model

Elasticity model

The multibody dynamics model (3) are treated as n_{j} independent SISO systems, and the transfer behavior of the axis model is similar to that of the well-known elastic coupled two-mass dynamics model [27]. After neglecting the nonlinear vector $\boldsymbol{n}$ of (3) and the friction and the coupling effects (off-diagonal elements of mass matrix), the frequency-domain motor response from motor torque τ_{m} to motor velocity $\dot{\theta}$ for each axis reads

$$\left.\frac{\dot{\theta}}{\tau_{\mathrm{m}}}\right|_{\boldsymbol{\tau}_{\mathrm{f}},\boldsymbol{n}=0} = \frac{1}{U} \cdot \frac{\frac{1}{m_\theta} \cdot s^2 + \frac{d}{m_\theta m_q} \cdot s + \frac{k}{m_\theta m_q}}{s^3 + \frac{m_\theta + m_q}{m_\theta m_q} d \cdot s^2 + \frac{m_\theta + m_q}{m_\theta m_q} k \cdot s} \tag{21}$$

and the mechanics response from the motor velocity $\dot{\theta}$ to the joint velocity $\dot{q}$

$$\left.\frac{\dot{q}}{\dot{\theta}}\right|_{\boldsymbol{\tau}_{\mathrm{f}},\boldsymbol{n}=0} = \frac{\frac{d}{m_q} \cdot s + \frac{k}{m_q}}{s^2 + \frac{d}{m_q} \cdot s + \frac{k}{m_q}} \tag{22}$$

The model parameters m_θ, m_q, k and d are then identified in the frequency domain for each axis separately using nonlinear least squares [44, §9.9] by fitting these two transfer functions.

Friction model

Friction can be measured at different velocities and fitted as the sum of viscous and Coulomb friction for each axis separately. Since it is not possible to measure them separately without an additional sensor, the total viscous friction is assumed to be equally distributed between the motor and the load, and the Coulomb friction is assumed to act only on the motor side, similar to [27]. In addition, the sign function of the Coulomb friction $\mathrm{sign}(\dot{\theta})$ is approximated by the hyperbolic tangent function $\tanh(\alpha \cdot \dot{\theta})$ to avoid numerical problems caused by non-smooth jumps around zero [45].

5.2. Hyperparameter selection

Gaussian process model

The noise variance σ_i^2 of (9) is set as the square of the maximum relative error of the encoders. The signal variance σ_{S}^2 of (12) is estimated from the variance of the measured output discrepancy. The length scale parameters $\boldsymbol{L} = \mathrm{diag}(l_1, l_2)$ of (12) can be chosen iteratively to balance the smoothness of the input space with a good prediction result, which can also be obtained by likelihood maximization or cross-validation [41, §5.4].

Model predictive control

The prediction horizon N is a trade-off between tracking performance and computation time [31, §7], and can be chosen by starting from a large value (upper bounded by computation time) and reducing it before any obvious performance degradation occurs in the simulation. The cost function is normalized by the inverse of the squared mean of the output error, which can be estimated from the measurement. This normalizes the cost and, therefore, improves numerical conditioning of the optimization problem. In addition, the selection of output weight Q and input weight R are simplified by setting $R = 1$. Then, the weight Q is selected to balance between small tracking errors (large Q) and small control input (small Q). This can be selected by reducing the Q from a large value until an observation of a significant difference of the optimized control input. The terminal weight is then selected as $Q_f = \kappa Q$ and the factor $\kappa > 1$ can be set to a large value to improve the tracking performance of MPC scheme [31].

5.3. Real-time implementation

Local approximation of GP model

To achieve fast approximate GP prediction for real-time applications, the full model (10) is locally approximated by the nearest neighbor method [46]. The basic idea is that the kernels of the GP model affect the prediction result only locally, and the data points closest to the test input point have the most influence. The closest points $\tilde{\boldsymbol{Q}}_j^*$ within predefined box constraints are searched at each prediction step along each direction of the input space, which can be easily implemented by index searching for the uniformly discretized input space. The local approximation is computed by multiplying $k(\tilde{\boldsymbol{q}}_j, \tilde{\boldsymbol{Q}}_j^*)^\top$ by the GP coefficients $\boldsymbol{\beta}_j^*$ corresponding to $\tilde{\boldsymbol{Q}}_j^*$, cf. (10). The size of the box constraints is a trade-off between the prediction accuracy and the computational complexity, and can be determined by comparing the local approximation with the full model for a given level of accuracy.

Numerical solution of MPC problem

The feedforward scheme is implemented independently for each axis, neglecting the coupling effects between different axes. The overall dynamics model (3) is treated as n_{j} independent SISO systems, and thus the GP prediction and numerical optimization can be computed in parallel on different isolated CPUs, making the total computation time independent of the number of axes, as opposed to sequential computation. Second, the optimization problem (18) is formulated as a sparse quadratic program by including all states and inputs into the optimization variable [30, §8.5.3], where the system dynamics are expressed by local equality constraints between neighboring states. The main advantage of the sparse formulation is that only the operations of the non-zero elements need to be computed, which can reduce the computational complexity, especially when the optimization problem contains many zeros, as is the case in our MPC. The resulting optimization problem is solved by the Operator Splitting Solver (OSQP) [47] and is warm-started from the previous solution at each solution step.

Table 1
Computation time of MPFFC scheme measured on different CPUs.

CPU time	i5-4670 (2.7 GHz)	i7-11850HE (2.6 GHz)	i9-10900KF (3.7 GHz)
mean [µs]	63	64	33
max. [µs]	96	82	46

6. Experimental results

To demonstrate the benefits of the proposed framework, an experimental validation is performed on an industrial setup running for one single axis and three linear axes combined. The setup and identification steps are presented, followed by a comparison of the tracking performance of a single axis with other feedforward design methods and a robustness analysis of the MPFFC scheme with respect to parametric and nonparametric perturbations. Finally, a freeform tracking experiment is conducted to three axes executing G-code of the CNC to demonstrate the effect on the contour error of the machine. All experimental data are available on [38].

6.1. Experimental setup and computational requirements

The experimental setup considers the three translational axes (x,y,z) of the five-axis milling machine shown in Fig. 2. The drives are Rexroth MS2N03-D0BYN with a rated torque of 0.68 N m, maximum torque of 6.8 N m and a rated velocity of 5700 1/min. The axes are Franke TSL06U ball screw drives with a spindle lead of 5 mm, and effectively reachable lengths of 0.36, 0.18 and 0.18 m in x, y and z directions, respectively. The motors are controlled by Rexroth ctrlX DRIVE inverters and commanded from a Beckhoff TwinCAT3 PLC/CNC system with a sampling rate of $(\Delta t)^{-1} = 1$ kHz. All parts of the feedforward scheme, including the GP prediction and the numerical optimization, are implemented in PLC and C++ code on the TwinCAT 3 control system.

The local evaluation of the GP is constrained locally to a box by requiring a remaining accuracy of 99% compared to the full model prediction, resulting in constraints of ±20 mm in position and ±40 mm/s in velocity. The MPC algorithm (Problem 1) is tuned according to Section 5.2, setting the weights $Q = 10, R = 1, Q_f = 100 \cdot Q = 1000$ and prediction horizon $N = 5$ for all three axes. A maximum number of iterations of 25 is used to ensure an upper bound in the computation time. The MPFFC algorithm is implemented under independent tasks distributed across different CPU cores and computed in parallel for each axis. The computation time of the MPFFC algorithm, including the GP prediction and the numerical optimization, is measured in the TwinCAT 3 system on different CPUs given in Table 1. For example, on an i7-11850HE (2.6 GHz) CPU, the mean and maximum execution times are 64 and 82 µs. Even in the worst case with the i5-4670 (2.7 GHz) CPU, the maximum computation time is 96 µs, which is only 10% of the sampling time. In addition, a total memory of 37.6 kB is required to store the prediction parameters $\boldsymbol{\beta}$ in (10) in double precision for all three axes.

6.2. System identification

The identification of the multi-body dynamics model of the motion stage is treated as the identification of three two-mass dynamics models including the Coulomb friction [27]. For the z-axis, the gravity is modeled additionally. The identification is shown exemplarily for the y-axis in the following, which has the lowest stiffness. The results for all three axes can be found in the dataset [38].

For the identification of the motor drive dynamics and axis elasticity, the frequency response functions (FRFs) of the motor (21) (from motor torque τ_m to motor velocity $\dot{\theta}$) and the mechanics (22) (from motor velocity $\dot{\theta}$ to load-side axis velocity $\dot{q}$) are fitted to the experimental data in the frequency domain using a least squares method.

Table 2
Identified parameters of the multibody dynamics model.

Drive	m_θ [kg]	m_q [kg]	k [N/m]	d [N m/s]	d_θ, d_q [N m/s]	F_c [N]
x-axis	138.8	10.7	$5.3 \cdot 10^6$	$4.4 \cdot 10^2$	$1.3 \cdot 10^3$	348.1
y-axis	99.5	18.5	$9.5 \cdot 10^5$	$1.1 \cdot 10^3$	$9.6 \cdot 10^2$	138.9
z-axis	85.7	20.0	$1.6 \cdot 10^7$	$4.8 \cdot 10^3$	$1.4 \cdot 10^3$	408.1

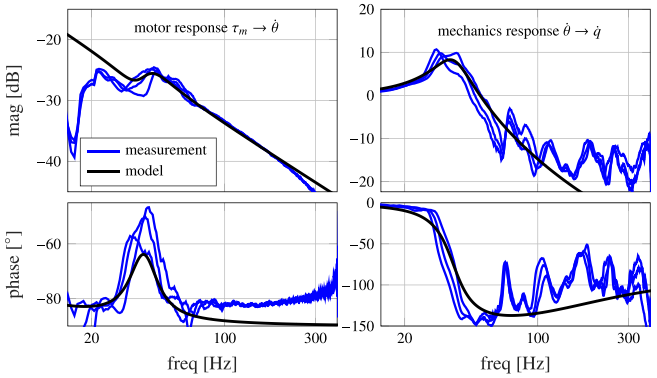


Fig. 4. Comparison of the frequency response of the y-axis and the identified axis model.

The FRFs are determined using sinusoidal velocity sweeps with linearly increasing frequency $f \in [10, 400]$ Hz measured at different starting positions over the axis range. An offset velocity of 10 mm/s is used to minimize the nonlinear influence of the friction $\boldsymbol{\tau}_f$.

The identification is performed for the three axes by fitting the transfer functions (21) and (22) in the frequency domain [44, §9.9], and the second axis with the lowest stiffness is shown in Fig. 4. The resulting parameters are given in Table 2. Obviously, the two-mass dynamics model only captures the transfer behavior up to the first natural frequency. To identify a more accurate model that captures the higher frequency content from 80 Hz, additional states must be introduced to describe the higher order dynamics. However, these states are not directly measurable and identification would require additional sensors [48]. On the other hand, the industrial reference trajectories generated by the CNC are band-limited [1, §7.2.1], and the higher order dynamics are outside the interesting frequency band for feedforward control.

The output discrepancy is measured (see Section 3.4) over the workspace at the commanded velocity $\dot{\boldsymbol{q}}_\mathrm{d} = \{20, 60, 100\}$ mm/s with a grid of 40 mm/s, which captures the difference between the measured load-side axis velocity and the simulated output of the extended loop model in Section 3.2. The hyperparameters of the GP regressors are selected according to Section 5.2, and the results of all three axes are given in [38].

The GP model is validated on the test bench at unseen velocities during the training phase to access its generalizability. The normalized validation result of the GP regression model at $\dot{\boldsymbol{q}}_\mathrm{d} = \{40, 80\}$ mm/s is shown in Fig. 5. Overall, this result shows a high coefficient of determination $R^2 = 93\%$ between measurement and prediction, and a mean absolute prediction error of 1.43 µm, which illustrates the generalizability of the GP model to unseen operating conditions. Furthermore, even in the case of significant prediction error due to factors such as wear or varying lubrication conditions, the measurement data can still be stored for subsequent updating of the regressor parameters.

Finally, the identified three-axis hybrid simulation model is validated with a multi-axis experiment tracking a freeform butterfly contour at a feedrate of 6000 mm/min, compare [49] and Section 6.5. The axes are controlled by industrial standard P-PI cascade controllers with a nominal velocity reference as feedforward. The simulation error $\boldsymbol{\xi}_k := \boldsymbol{q}_{k,\mathrm{meas}} - \boldsymbol{q}_{k,\mathrm{sim}}$ of the load-side axis position are quantified in Table

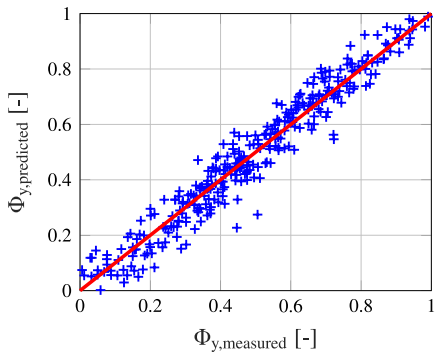


Fig. 5. Validation of normalized GP prediction on y-axis at unseen velocities.

Table 3
Simulation error of hybrid model validated by multi-axis experiment.

Absolute simulation error	x-axis	y-axis	z-axis
Mean [μm]	28.1	30.3	12.3
Maximum [μm]	124.1	196.8	60.9

Table 4
Tracking performance of the respective controllers.

	baseline	torqueff	rigid-velff	MPFFC
mae(e_q) [μm]	4.38	4.40	4.31	2.18
max($\lvert e_q \rvert$) [μm]	144.82	138.63	140.78	72.60

3. Compared to the axis operating range, even the worst case simulation errors at the transient stage are only 0.03%, 0.11% and 0.03% for the x, y and z axes respectively.

6.3. Axis tracking performance

A key performance indicator for axis motion is the tracking performance for linear motion, which is an indicator for the surface finish quality of workpieces produced on a machine tool. The tracking behavior is studied for a single axis (y-axis) using a fast linear trajectory (G01) using a speed profile in both directions with dynamic limits of v_{max} = 90 mm/s, a_{max} = 5000 mm/s^2 and j_{max} = 30 000 mm/s^3. For comparison, standard P-PI cascaded control with constant parameters is used together with the following feedforward schemes:

baseline Industry standard feedforward using the speed of the nominal reference trajectory as velocity feedforward.

torqueff In addition to the velocity feedforward described above, a motor torque feedforward is added to compensate for the rigid body drive inertia and friction [27].

rigid-velff Analytical velocity feedforward assuming rigid-body mechanics model, derived in Appendix.

MPFFC Our proposed optimization-based predictive velocity feedforward combining physics-based elasticity model and learning-based discrepancy model (Algorithm 1).

The position and velocity of the tracking experiment is shown in Fig. 6. The resulting tracking error is shown in Fig. 7 and further summarized in Table 4. For a quantitative comparison, the performance is evaluated with the mean absolute error (mae) and the maximum absolute error (max), where the mean absolute tracking error during the constant velocity phase mae(e_q) is measured for $t \in [0.4, 1.6] \cup [2.2, 3.4]$ s.

It is observed, that the MPFFC outperforms the other feedforward schemes by reducing the mae by more than 49.4% compared to the

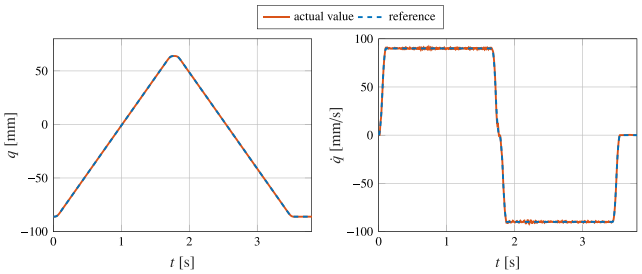


Fig. 6. Position (left) and velocity (right) of MPFFC for tracking the reference trajectory.

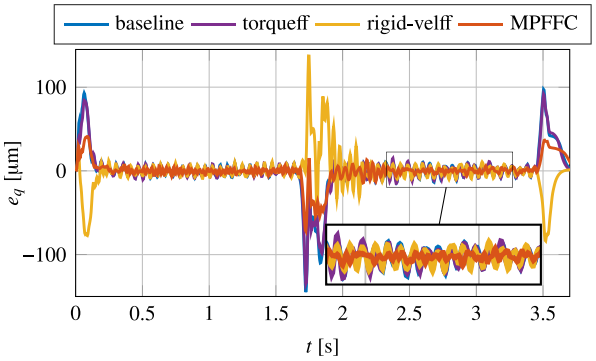


Fig. 7. Measured transient tracking behavior of different feedforward schemes.

Table 5
Power consumption on y-axis with different feedforward schemes.

DC power	baseline	torqueff	rigid-velff	MPFFC
Mean [W]	126.9	125.0	126.3	124.3
Maximum [W]	315.7	304.9	348.2	341.4

other feedforward methods. Similarly, the maximum tracking error is reduced by 49.9% and 47.6% compared to the baseline and the torque feedforward of acceleration and friction, respectively. The additional torque feedforward does not show any significant improvements, which may be due to uncertainties in the torque equations, such as more complex friction and other unmodeled nonlinearities, e.g., the effects of lead errors and backlash typically appear themselves together with the elastic deformation, which makes it difficult to model them separately without additional measurement equipment [27,50]. Furthermore, the rigid body model neglects the elasticity of the drive train and approximates the mechanical dynamics with a first-order lag term. This could work for axes with a higher stiffness ratio, but in this case there is too much compliance, meaning that the lowest natural frequency is due to the drive mechanics. Inversion-based feedforward design of the velocity loop with a rigid mechanical model assumption leads to overshoot behavior in this case.

An interesting point for the feedforward controller is the power consumption, which was measured for 10 repetitions of the tracking trajectory using the active DC voltage and the drive current. As can be seen from Fig. 8 and Table 5, the average power consumption of the MPFFC is slightly lower compared to the baseline velocity feedforward control, which could be due to less steady-state errors at constant velocities (which will be analyzed further in the following section). The maximum power consumption, which occurs during the acceleration phase, is slightly higher by 8%, which is the price for the 50% smaller tracking errors in these transient phases.

To evaluate the steady-state errors, three different constant velocities $\dot{q}_d \in \{70, 80, 90\}$ mm/s are chosen, which are unseen operating

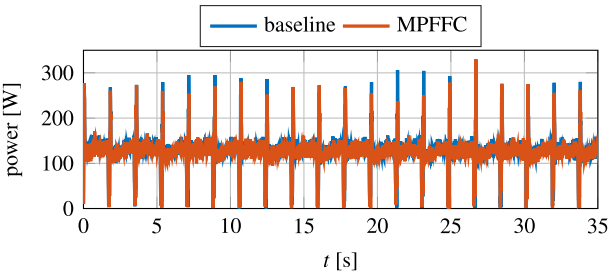
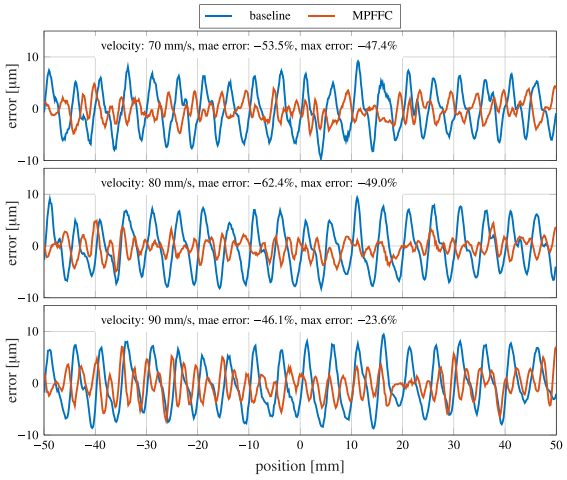


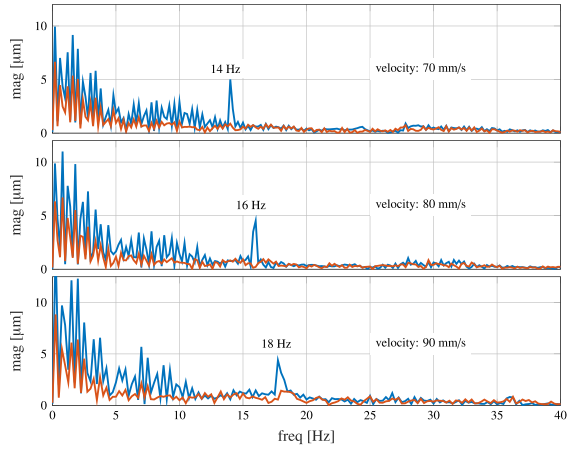
Fig. 8. Measured power consumption on y-axis during 10 repetitions of the axis tracking experiment.

Table 6
Stationary tracking errors at unseen velocities.

		baseline	MPFFC
vel. 70 mm/s	mae(e_q) [µm]	3.35	1.56
	max(|e_q|) [µm]	9.62	5.06
vel. 80 mm/s	mae(e_q) [µm]	3.62	1.36
	max(|e_q|) [µm]	9.67	4.93
vel. 90 mm/s	mae(e_q) [µm]	3.84	2.07
	max(|e_q|) [µm]	9.47	7.24



(a) Attenuation of steady-state tracking errors in the time domain.



(b) Attenuation of steady-state tracking errors in the frequency domain.

Fig. 9. Measured constant-velocity tracking performance on the *y*-axis under unseen operation conditions.

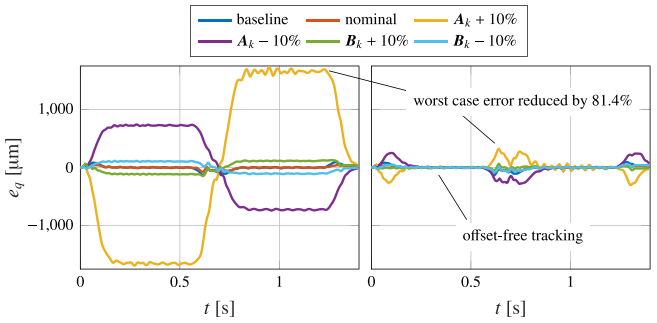


Fig. 10. Robustness test against nonparametric perturbation of state and input matrices by ±10% (left: without integral action, right : with integral action).

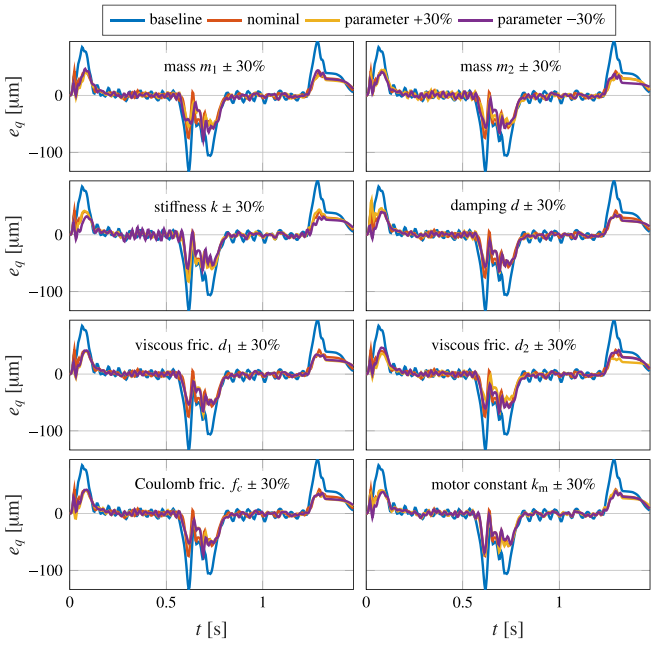
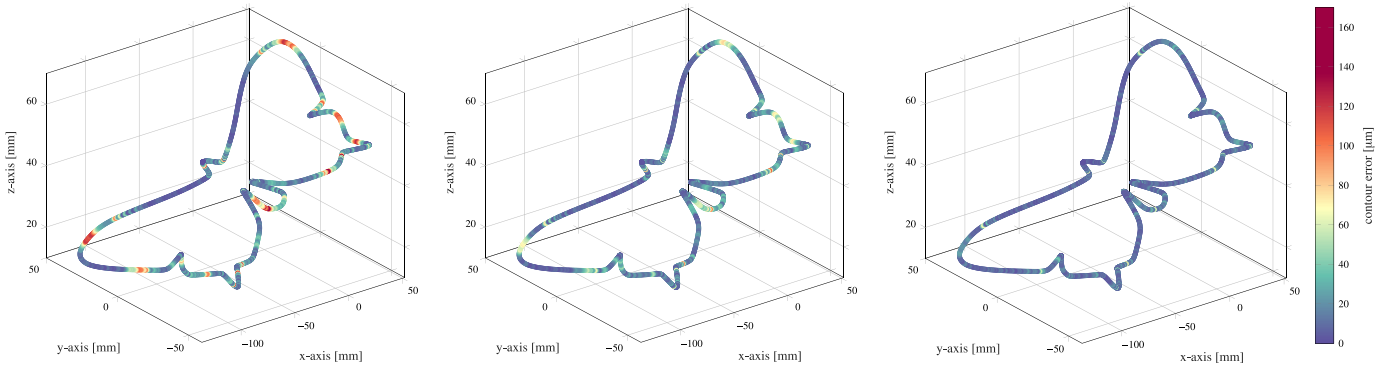


Fig. 11. Robustness test against variation of model parameters by ±30%.

points during the GP training to test the generalizability of the GP model. Fig. 9(a) shows the steady-state tracking behavior with the corresponding feedforward controllers, a summary of the tracking errors is given in Table 6. Even under all unseen operating conditions, the MPFFC still achieves an error reduction by at least 46.1% of mae and the maximum error is reduced between 23.6% and 49.0%. These results are also confirmed by analyzing the tracking errors in the frequency domain by using the Fast Fourier Transform (FFT) in Fig. 9(b). It can be seen that by using the baseline controller, the velocity-dependent frequency content of the tracking error, which is mainly caused by the spindle lead error, has a single dominant harmonic frequency and also some quasi-dynamic content as lower frequencies. The proposed MPFFC greatly attenuates both these frequency contents through the discrepancy learning with GP, resulting in improved feedforward control performance.

6.4. Robustness analysis

In addition to the performance studies, the robustness of the presented feedforward scheme is also investigated experimentally, which is separated into a study considering nonparametric perturbations of the state and input matrices ($\boldsymbol{A}_k$ and $\boldsymbol{B}_k$), and a robustness test against varying model parameters.



(a) Butterfly with baseline **(b)** Butterfly with MPFFC **(c)** Butterfly with MPFFC and reduced stiffness

Fig. 12. Experimental comparison of contour errors on the butterfly trajectory with three axes and a feedrate of 6000 mm/min.

To conceptually illustrate the benefits of introducing the additional integral action to the MPFFC scheme as discussed in Section 4.3, the transient tracking behavior is tested by varying directly the prediction matrices $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$ of the MPFFC scheme by ±10%. The comparison of the MPFFC scheme with and without the addition integral action is shown in Fig. 10, where the tracking error at the transient stage are measured during a fast linear trajectory (G01) moving in both directions with dynamic limits given in Section 6.3. It can be seen that, despite a direct variation of the entire state matrix $\boldsymbol{A}_k$ by +10%, the addition integral action of the MPFFC scheme can still counteract the unseen nonparametric perturbation of the system matrices to ensure offset-free output tracking, and reduce the worst-case tracking error at the transient stage by 81.4% compared to the MPFFC design without integral action.

In addition, plant parameters typically deteriorate over the life of the machine due to external influences, such as temperature changes, which typically affect the friction. Thus, the robustness is also investigated experimentally against the mismatch of model parameters by ±30%. The transient tracking results of the MPFFC scheme are shown in Fig. 11. It can be observed, that even such a relatively large parameter variation does not significantly affect the resulting tracking performance of the MPFFC scheme. The maximum degradation of the mae tracking error compared to the nominal MPFFC design is 4.7%. Furthermore, the worst case MPFFC still improves the mae error by more than 38.5% compared to the baseline feedforward. This illustrates the significant robustness of the MPFFC scheme against parametric model uncertainties.

Another interesting observation is that varying some model parameters, such as reducing the model stiffness, actually improves the tracking error by an average of 11.9% compared to the nominal design. It may seem that the identification result is not very accurate in some parameters, and by changing these parameters, the resulting performance can be improved. However, this is due to the fact that the model parameters are mostly identified in the frequency domain, and the identified simulator is then used for feedforward design in the time domain. It should be noted that an accurate frequency response only aims to capture the dominant dynamics such as vibration modes, but does not necessarily imply the most accurate feedforward control action due to the neglected nonlinearity in the modeling. On the other hand, direct identification of model parameters in the time domain is still intractable in practice due to inevitably neglected unmeasurable nonlinear effects [27,50], especially for multi-axis machines with compliant mechanics. Therefore, a systematic method to use the degrees of freedom provided by the model parameters for tuning still remains an open question for future analysis.

Table 7
Contour error of the 3-axis butterfly toolpath (feedrate: 6000 mm/min).

	baseline	MPFFC nom.	MPFFC tuned
Mean [µm]	25.67	17.27	11.09
Maximum [µm]	167.98	109.21	84.61

6.5. Multi-axis tracking experiment on the test bench

To demonstrate the multi-axis tracking performance of the proposed MPFFC approach, a three-axis experiment was performed on reference commands generated by an industrial CNC using G-code on the x, y and z axes of the test bench. A freeform butterfly contour was used as the toolpath [49], which was converted to linear G01 segments. Also, the movement of the z-axis was scaled from the x-axis according to the workspace to obtain a 3D toolpath, see Fig. 12. The trajectory was planned using TwinCAT CNC (with a feedrate of 6000 mm/min and axis dynamics $a_{\max,x|y|z} = 5000$ mm/s^2, $j_{\max,x|y|z} = 50\,000$ mm/s^3) and smoothed using the High Speed Cutting (HSC) contour smoothing feature [51] with an allowed path deviation of 0.8 mm. All axis configuration parameters, as well as the G-code used for the trajectory and the desired and actual axis motion can be found in the data repository [38]. The metric for comparison is the contour error, which was calculated using Dynamic Time Warping (`dtw` in Matlab with Euclidean distance). The MPFFC was also done with a variant where the stiffness k for the model was reduced by $\{50, 25, 50\}$%, respectively. This reduction was found heuristically and is included to show the potential of the proposed approach with further parameter tuning. The resulting contour errors are visualized graphically in Fig. 12 and numerically summarized in Table 7. The mean absolute contour error is reduced by 32.7% using the nominal MPFFC and by further tuning the MPFFC model a reduction of 56.8% is achieved. The peak contour error is even reduced by 34.9% MPFFC and 49.6% using the MPFFC with tuned model parameters. This trend is confirmed by visual inspection of the contour error in Fig. 12, where the high tracking error segments (yellow to red) are significantly reduced.

6.6. Discussion on applications

The experimental results confirm the benefits of the presented optimization-based feedforward framework with hybrid modeling to increase the dynamic path accuracy of machine tools. This is particularly beneficial for processes without external forces, such as laser processes, where the dynamics of the machine tools dominate the finishing quality and can be accurately captured by combining the physics-based and machine learning methods. For applications where the attenuation of unknown disturbances is more important, such as milling, dedicated feedback control methods or online learning schemes are more suitable.

The proposed feedforward scheme can be directly transferred to the industrial hardware for independent axis control without modifying the existing cascaded control structure. Due to its open-loop nature (feedforward only) and its computational efficiency (average runtime 63 μs on an i5 CPU including optimization and GP prediction), the algorithm can be implemented either directly in the frequency converter as a separate function block, or in the CNC kernel and commanded via field bus technologies (e.g. Profinet, EtherCAT).

7. Conclusion

In this paper, an optimization-based predictive feedforward control framework with hybrid modeling is presented and applied to independent axis control of a milling machine. The hybrid model, developed with a particular focus on its use in dynamics simulation, combines a physics-based model of the multibody dynamics and a Gaussian process regressor of the output discrepancy. The feedforward design method separates the nonlinear hybrid model used for simulation from the linearized predictor used for input optimization. This strategy allows the accurate but complicated hybrid dynamics model to be used for real-time feedforward control in a computationally efficient way via online convex optimization. Extensive experimental results on a milling machine illustrate the real-time capability and the significant improvement in dynamics path accuracy under different operating conditions, without introducing additional power consumption. Furthermore, the robustness of the proposed feedforward scheme to parametric and nonparametric model uncertainties is demonstrated experimentally.

Future work includes the automatic parameterization of the presented MPFFC scheme, including simultaneous tuning of control and model parameters, which may provide more degrees of freedom for performance improvement. Another interesting extension under this subject is the integration of additional acceleration sensors at the machine tool end, which could be combined with the proposed feedforward framework for damping control of multi-axis processing.

CRediT authorship contribution statement

Haijia Xu: Validation, Investigation, Writing – review & editing, Software, Methodology, Writing – original draft, Conceptualization. **Christoph Hinze:** Writing – original draft, Conceptualization, Investigation, Software, Funding acquisition, Writing – review & editing, Methodology. **Andrea Iannelli:** Methodology, Conceptualization, Writing – review & editing. **Zexu Zhou:** Data curation, Writing – review & editing, Visualization. **Alexander Verl:** Resources, Supervision.

Declaration of competing interest

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

Acknowledgments

This work is supported by the Ministry of Science, Research and Arts of the Federal State of Baden-Württemberg within the Innovation-Campus Future Mobility (ICM), Germany. We would also like to thank Lihan Xu for his support in implementing the numerical solvers on the TwinCAT 3 system and Marcel Dzubba for the fruitful discussions on scientific writing with LaTeX.

Appendix. Analytical velocity loop inversion with rigid-body mechanics model

Consider only the rigid body mechanics model of drive motor, the frequency response can be written as

$$G_{\mathrm{RB}} = \frac{\dot{\theta}}{U^{-1}\tau_{\mathrm{m}}} = \frac{1}{m_\theta s + d} \tag{A.1}$$

and the proportional–integral (PI) velocity controller

$$K_{\mathrm{vel}} = \frac{\tau_{\mathrm{m,d}}}{e_{\dot{\theta}}} = k_p\left(1 + \frac{k_i}{s}\right) \tag{A.2}$$

Because the current control loop is much faster than mechanics [1, §7] and consider $\tau_{\mathrm{m}} \approx \tau_{\mathrm{m,d}}$, the PI-controlled velocity control loop can be represented as

$$G_{\mathrm{vel}} = \frac{\dot{\theta}}{\dot{\theta}_{\mathrm{d}}} = \frac{U^{-1}G_{\mathrm{RB}}K_{\mathrm{vel}}}{1 + U^{-1}G_{\mathrm{RB}}K_{\mathrm{vel}}} \tag{A.3}$$

$$= \frac{1 + \frac{1}{k_i}s}{1 + \frac{Ud + k_p}{k_p k_i}s + \frac{Um_\theta}{k_p k_i}s^2} \tag{A.4}$$

where the inverse feedforward can be represented as an infinite impulse response (IIR) filter followed by a first-order lag term.

Data availability

All experimental data are openly available on [38].

References

[1] Y. Altintas, A. Verl, C. Brecher, L. Uriarte, G. Pritschow, Machine tool feed drives, CIRP Ann. 60 (2) (2011) 779–796, http://dx.doi.org/10.1016/j.cirp.2011.05.010.
[2] G.M. Clayton, S. Tien, K.K. Leang, Q. Zou, S. Devasia, A review of feedforward control approaches in nanopositioning for high-speed SPM, J. Dyn. Syst. Meas. Control. 131 (6) (2009) http://dx.doi.org/10.1115/1.4000158.
[3] H.N. Huynh, Y. Altintas, Multibody dynamic modeling of five-axis machine tool vibrations and controller, CIRP Ann 71 (1) (2022) 325–328, http://dx.doi.org/10.1016/j.cirp.2022.04.003.
[4] C.-P. Wang, K. Erkorkmaz, J. McPhee, S. Engin, In-process digital twin estimation for high-performance machine tools with coupled multibody dynamics, CIRP Ann 69 (1) (2020) 321–324, http://dx.doi.org/10.1016/j.cirp.2020.04.047.
[5] P. Poignet, M. Gautier, W. Khalil, M.T. Pham, Modeling, simulation and control of high speed machine tools using robotics formalism, Mechatron. 12 (3) (2002) 461–487, http://dx.doi.org/10.1016/S0957-4158(01)00003-4.
[6] P.V.D. Braembussche, J. Swevers, H.V. Brussel, P. Vanherck, Accurate tracking control of linear synchronous motor machine tool axes, Mechatron. 6 (5) (1996) 507–521, http://dx.doi.org/10.1016/0957-4158(95)00090-9.
[7] H.-T. Yau, J.-J. Yan, Adaptive sliding mode control of a high-precision ball-screw-driven stage, Nonlinear Anal. Real World Appl. 10 (3) (2009) 1480–1489, http://dx.doi.org/10.1016/j.nonrwa.2008.01.025.
[8] V. Nasir, F. Sassani, A review on deep learning in machining and tool monitoring: methods, opportunities, and challenges, Int. J. Adv. Manuf. Technol. 115 (9–10) (2021) 2683–2709, http://dx.doi.org/10.1007/s00170-021-07325-7.
[9] X. Li, X. Liu, C. Yue, S.Y. Liang, L. Wang, Systematic review on tool breakage monitoring techniques in machining operations, Int. J. Mach. Tools Manuf. 176 (2022) 103882, http://dx.doi.org/10.1016/j.ijmachtools.2022.103882.
[10] T. Bergs, D. Biermann, K. Erkorkmaz, R. M'Saoubi, Digital twins for cutting processes, CIRP Ann 72 (2) (2023) 541–567, http://dx.doi.org/10.1016/j.cirp.2023.05.006.
[11] R.T. Chen, Y. Rubanova, J. Bettencourt, D.K. Duvenaud, Neural ordinary differential equations, Adv. Neural Inf. Process. Syst. 31 (2018).
[12] M. Perčić, S. Zelenika, I. Mezić, Artificial intelligence-based predictive model of nanoscale friction using experimental data, Friction 9 (6) (2021) 1726–1748, http://dx.doi.org/10.1007/s40544-021-0493-5.
[13] G. Pillonetto, A. Aravkin, D. Gedon, L. Ljung, A.H. Ribeiro, T.B. Schön, Deep networks for system identification: A survey, Autom. 171 (2025) 111907, http://dx.doi.org/10.1016/j.automatica.2024.111907.
[14] Y.-P. Liu, Y. Altintas, Predicting the position-dependent dynamics of machine tools using progressive network, Precis. Eng. 73 (2022) 409–422, http://dx.doi.org/10.1016/j.precisioneng.2021.10.010.
[15] C.-Y. Tai, Y. Altintas, A hybrid physics and data-driven model for spindle fault detection, CIRP Ann 72 (1) (2023) 297–300, http://dx.doi.org/10.1016/j.cirp.2023.04.054.

[16] S. Guo, M. Agarwal, C. Cooper, Q. Tian, R.X. Gao, W. Guo, Y. Guo, Machine learning for metal additive manufacturing: Towards a physics-informed data-driven paradigm, J. Manuf. Syst. 62 (2022) 145–163, http://dx.doi.org/10.1016/j.jmsy.2021.11.003.

[17] C. Cooper, J. Zhang, R.X. Gao, Error homogenization in physics-informed neural networks for modeling in manufacturing, J. Manuf. Syst. 71 (2023) 298–308, http://dx.doi.org/10.1016/j.jmsy.2023.09.013.

[18] V. Ostad Ali Akbari, A. Eichenberger, K. Wegener, Physics-supported Bayesian machine learning for chatter prediction with process damping in milling, CIRP J. Manuf. Sci. Technol. 55 (2024) 165–173, http://dx.doi.org/10.1016/j.cirpj.2024.09.014.

[19] R. Wang, Q. Song, Y. Peng, J. Qin, Z. Liu, Z. Liu, Toward digital twins for high-performance manufacturing: Tool wear monitoring in high-speed milling of thin-walled parts using domain knowledge, Robot. Comput.-Integr. Manuf. 88 (2024) 102723, http://dx.doi.org/10.1016/j.rcim.2024.102723.

[20] Y. Jiang, J. Chen, H. Zhou, J. Yang, G. Xu, Residual learning of the dynamics model for feeding system modelling based on dynamic nonlinear correlate factor analysis, Appl. Intell. 51 (7) (2021) 5067–5080, http://dx.doi.org/10.1007/s10489-020-02096-2.

[21] C.-H. Chou, M. Duan, C.E. Okwudire, A linear hybrid model for enhanced servo error pre-compensation of feed drives with unmodeled nonlinear dynamics, CIRP Ann. 70 (1) (2021) 301–304, http://dx.doi.org/10.1016/j.cirp.2021.04.070.

[22] Z. Huang, M. Fey, C. Liu, E. Beysel, X. Xu, C. Brecher, Hybrid learning-based digital twin for manufacturing process: Modeling framework and implementation, Robot. Comput.-Integr. Manuf. 82 (2023) 102545, http://dx.doi.org/10.1016/j.rcim.2023.102545.

[23] S. Ji, H. Ni, T. Hu, J. Sun, H. Yu, H. Jin, DT-CEPA: A digital twin-driven contour error prediction approach for machine tools based on hybrid modeling and sparse time series, Robot. Comput.-Integr. Manuf. 88 (2024) 102738, http://dx.doi.org/10.1016/j.rcim.2024.102738.

[24] S. Zhou, M.K. Helwa, A.P. Schoellig, An inversion-based learning approach for improving impromptu trajectory tracking of robots with non-minimum phase dynamics, IEEE Robot. Autom. Lett. 3 (3) (2018) 1663–1670, http://dx.doi.org/10.1109/lra.2018.2801471.

[25] M. Bolderman, M. Lazar, H. Butler, Physics–guided neural networks for inversion–based feedforward control applied to linear motors, in: 2021 IEEE Conference on Control Technology and Applications, CCTA, IEEE, 2021, http://dx.doi.org/10.1109/ccta48906.2021.9659174.

[26] M. Bolderman, H. Butler, S. Koekebakker, E. van Horssen, R. Kamidi, T. Spaan-Burke, N. Strijbosch, M. Lazar, Physics-guided neural networks for feedforward control with input-to-state-stability guarantees, Control Eng. Pract. 145 (2024) 105851, http://dx.doi.org/10.1016/j.conengprac.2024.105851.

[27] A. Kamalzadeh, D.J. Gordon, K. Erkorkmaz, Robust compensation of elastic deformations in ball screw drives, Int. J. Mach. Tools Manuf. 50 (6) (2010) 559–574, http://dx.doi.org/10.1016/j.ijmachtools.2010.03.001.

[28] H. Xu, C. Hinze, A. Iannelli, A. Verl, Robust inversion-based feedforward control with hybrid modeling for feed drives, IEEE Trans. Control Syst. Technol. (2024) 1–14, http://dx.doi.org/10.1109/tcst.2024.3512862.

[29] M.G. Tamizi, M. Yaghoubi, H. Najjaran, A review of recent trend in motion planning of industrial robots, Int. J. Intell. Robot. Appl. 7 (2) (2023) 253–274, http://dx.doi.org/10.1007/s41315-023-00274-2.

[30] J. Rawlings, D. Mayne, M. Diehl, Model Predictive Control: Theory, Computation, and Design, Nob Hill Publishing, 2017, URL: https://books.google.de/books?id=MrJctAEACAAJ.

[31] M. Schwenzer, M. Ay, T. Bergs, D. Abel, Review on model predictive control: an engineering perspective, Int. J. Adv. Manuf. Technol. 117 (5–6) (2021) 1327–1349, http://dx.doi.org/10.1007/s00170-021-07682-3.

[32] G. Shang, L. Xu, Z. Li, Z. Zhou, Z. Xu, Digital-twin-based predictive compensation control strategy for seam tracking in steel sheets welding of large cruise ships, Robot. Comput.-Integr. Manuf. 88 (2024) 102725, http://dx.doi.org/10.1016/j.rcim.2024.102725.

[33] J. Huber, C. Gruber, M. Hofbaur, Online trajectory optimization for nonlinear systems by the concept of a model control loop - applied to the reaction wheel pendulum, in: 2013 IEEE International Conference on Control Applications, CCA, IEEE, 2013, http://dx.doi.org/10.1109/cca.2013.6662871.

[34] D. Erdogan, S. Jakubek, C. Mayr, C. Hametner, Model predictive feedforward control for high-dynamic speed control of combustion engine test beds, IEEE Open J. Ind. Appl. 2 (2021) 82–92, http://dx.doi.org/10.1109/ojia.2021.3073884.

[35] J. Yang, H.-T. Zhang, H. Ding, Contouring error control of the tool center point function for five-axis machine tools based on model predictive control, Int. J. Adv. Manuf. Technol. 88 (9–12) (2016) 2909–2919, http://dx.doi.org/10.1007/s00170-016-8979-4.

[36] F. Bonassi, C.F.O. da Silva, R. Scattolini, Nonlinear MPC for offset-free tracking of systems learned by GRU neural networks, IFAC-Pap. 54 (14) (2021) 54–59, http://dx.doi.org/10.1016/j.ifacol.2021.10.328.

[37] M.R. Ebers, K.M. Steele, J.N. Kutz, Discrepancy modeling framework: Learning missing physics, modeling systematic residuals, and disambiguating between deterministic and random effects, SIAM J. Appl. Dyn. Syst. 23 (1) (2024) 440–469, http://dx.doi.org/10.1137/22m148375x.

[38] C. Hinze, H. Xu, Replication Data for: Increasing dynamic accuracy using predictive feedforward with hybrid modeling [dataset], 2024, http://dx.doi.org/10.18419/darus-4513.

[39] M.W. Spong, S. Hutchinson, M. Vidyasagar, Robot Modeling and Control, John Wiley & Sons, 2020.

[40] J.C. Butcher, A history of runge-kutta methods, Appl. Numer. Math. 20 (3) (1996) 247–260.

[41] C.K. Williams, C.E. Rasmussen, Gaussian Processes for Machine Learning, Vol. 2, MIT press Cambridge, MA, 2006.

[42] G. Pannocchia, Offset-free tracking MPC: A tutorial review and comparison of different formulations, in: 2015 European Control Conference, ECC, IEEE, 2015, http://dx.doi.org/10.1109/ecc.2015.7330597.

[43] V. Leipe, C. Hinze, A. Lechler, A. Verl, Model predictive control for compliant feed drives with offset-free tracking behavior, Prod. Eng. 17 (6) (2023) 805–814.

[44] R. Pintelon, J. Schoukens, System Identification - A Frequency Domain Approach, John Wiley & Sons, Inc., 2012, http://dx.doi.org/10.1002/9781118287422.

[45] D. Zheng, J. Na, X. Ren, G. Herrmann, S. Longo, Adaptive control of robotic servo system with friction compensation, in: 2011 IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM), IEEE, 2011, http://dx.doi.org/10.1109/ramech.2011.6070497.

[46] R.B. Gramacy, D.W. Apley, Local Gaussian process approximation for large computer experiments, J. Comput. Graph. Statist. 24 (2) (2015) 561–578, http://dx.doi.org/10.1080/10618600.2014.914442.

[47] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, S. Boyd, OSQP: an operator splitting solver for quadratic programs, Math. Program. Comput. 12 (4) (2020) 637–672, http://dx.doi.org/10.1007/s12532-020-00179-2.

[48] M. Ruderman, T. Bertram, Modeling and observation of hysteresis lost motion in elastic robot joints, IFAC Proc. Vol. 45 (22) (2012) 13–18, http://dx.doi.org/10.3182/20120905-3-hr-2030.00061.

[49] M. Liu, Y. Huang, L. Yin, J. Guo, X. Shao, G. Zhang, Development and implementation of a NURBS interpolator with smooth feedrate scheduling for CNC machine tools, Int. J. Mach. Tools Manuf. 87 (2014) 1–15, http://dx.doi.org/10.1016/j.ijmachtools.2014.07.002.

[50] P. Mesmer, M. Neubauer, A. Lechler, A. Verl, Robust design of independent joint control of industrial robots with secondary encoders, Robot. Comput.-Integr. Manuf. 73 (2022) 102232, http://dx.doi.org/10.1016/j.rcim.2021.102232.

[51] B.G..C. K.G., HSC functions and contouring methods: TX1270: Twincat CNC, 2024, URL: https://infosys.beckhoff.com/english.php?content=../content/1033/hsc/index.html (Accessed 28 October 28).