Feedrate Optimization via Pass-to-Pass Learning–Applied to 2.5D Contour Machining under Servo Error and Spindle Torque Constraints

Cheng-Hao Chou , Milad Azvar , Chenhui Shao , Chinedum Okwudire *

Posted Date: 30 March 2026

doi: 10.20944/preprints202603.2368.v1

Keywords: computer numerical control (CNC); feedrate optimization; iterative learning; servo error; cutting force

 doi:10.20944/preprints202603.2368.v1

Article

Feedrate Optimization via Pass-to-Pass Learning–Applied to 2.5D Contour Machining under Servo Error and Spindle Torque Constraints

Cheng-Hao Chou, Milad Azvar, Chenhui Shao and Chinedum Okwudire *

Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109 USA
* Correspondence: okwudire@umich.edu

Abstract

Repeated machining passes (i.e., continuous toolpaths) are common in CNC manufacturing, including multi-level machining of prismatic parts and iso-contour passes in contour machining. They present an opportunity to exploit pass-to-pass learning to improve productivity without sacrificing quality through feedrate optimization. Traditional iterative learning methods provide a means to exploit pass-to-pass learning for quality improvements, but they are not well-suited to feedrate optimization because the reference trajectories change as the feedrate increases. In the authors' prior work, learning-based feedrate optimization was demonstrated for repeated machining along identical toolpaths. This paper extends that concept to the more challenging case of similar but non-identical cutting paths, as encountered in contour machining. A pass-to-pass learning strategy is proposed in which corresponding sections of non-identical iso-contour passes are identified using a contour-matching method based on geometric similarity. Bayesian linear regression models are then used to learn and predict contour error and spindle torque across passes, with uncertainty explicitly quantified through credible intervals. These predictions are embedded in a window-based feedrate optimization framework solved via sequential linear programming, enabling feedrate maximization subject to kinematic, contour-error, and spindle-torque constraints. The proposed approach is experimentally validated on a 3-axis desktop CNC milling machine through multiple 2.5D contour machining case studies. Results show that the method can rapidly approach near-optimal feedrates after only a few passes, culminating in up to 16.4% increase in productivity compared to an equivalent learning-based feedrate optimization approach for identical toolpaths.

Keywords: computer numerical control (CNC); feedrate optimization; iterative learning; servo error; cutting force

1. Introduction

In CNC machining, it is common to repeat similar or identical cutting operations with the same machine tool. For example, in mass production, where large numbers of identical parts are manufactured, identical cutting operations are repeated from part to part. Similarly, when machining prismatic parts, such as spur gears, the tool also repeatedly follows identical cutting paths at different z-levels to create a 3-D geometry. In addition, contour machining shapes the final part through the cutting operations of iso-contour passes, where the contours are not identical but similar from pass to pass. All of these repetitive cutting operations motivate learning-based strategies – including learning from part to part or from pass to pass – to enhance machining performance.

Productivity, quality, and cost are the key requirements in the manufacturing process. To improve productivity, driven by machine speed, and quality, driven by machine accuracy, at low cost, various software-based optimization and control strategies have been proposed. Among learning-based approaches, iterative learning control (ILC) has been explored in the literature to improve machine

 doi:10.20944/preprints202603.2368.v1

performance. ILC iteratively updates the control input using data from the previous iteration to minimize the system's error [1]. The ILC-based strategy has been applied to CNC interpolators and to the control of various feed drive systems for tracking and contour error compensation [2–7]. However, these traditional ILC methods typically require repeated reference or repeated disturbances. These requirements limit the effectiveness of traditional ILC for feedrate optimization applications, where the reference trajectory changes significantly from iteration to iteration as the feedrate increases.

Alternative ILC strategies have been proposed to address its limitations. In particular, one strategy is to introduce basis functions to enhance the extrapolation performance of ILC [8,9]. Another strategy is to learn the parameters of the pre-structure compensation model [10], rather than the compensation signal itself, thereby enabling ILC to be applied to varying tasks. However, these ILC studies still focus solely on minimizing servo errors. It improves machine accuracy but does not address machine productivity, which manufacturers care more about once the error has been controlled within a tolerable threshold.

To explicitly enhance machine productivity, feedrate optimization has been applied to increase the machine's speed within certain constraints. Various studies have been performed on feedrate optimization with kinematic constraints, including limits on velocity, acceleration, and jerk [11–14]. In addition, geometric errors, such as chord error, have been included as constraints in some literature [15–17]. However, while kinematic constraints can be related to machine capabilities or, indirectly, to machine error, these studies do not consider the machine's dynamics, which significantly affect speed-related machining errors.

To explicitly account for speed-related machine errors, servo error models have also been incorporated into feedrate optimization. For example, Sencer et al. generated the feedrate profile such that its frequency spectrum was suppressed near the machine's vibration modes [18]. Chen et al. estimated the steady-state contour error by using the frequency response magnitude of a transfer function model [19]. In addition, contour error has served as a constraint in the optimization problem and is predicted by static [20,21] or dynamic servo models [22], as well as by friction dynamics [21]. Kim and Okwudire further combined feedrate optimization and servo error pre-compensation, based on a physics-based model [23].

In addition to the servo error of the feed drives, cutting force presents another speed-related consideration in the cutting operation, as it affects surface quality, machine tool wear, and other factors. Cutting force may also act as an external disturbance to the motion system, potentially resulting in larger servo errors. Accordingly, feedrate optimization accounting for cutting force has also been studied. For example, Oh et al. aimed to optimize the feedrate to maintain a constant cutting force based on an analytical cutting force model [24]. Erkorkmaz et al. proposed a feedrate optimization framework incorporating kinematic constraints and cutting force limits [25]. However, all of the above-mentioned works rely solely on offline models, which cannot capture unmodeled dynamics or machine uncertainties.

To overcome these issues, online feedback or online learning can be adopted. For instance, in Mansour and Seethaler's work [26], cutting force measurements are fed back to override the scheduled feedrate to avoid exceeding the allowable chip thickness or in-plane force. Chang et al. proposed an online model predictive control (MPC) framework with real-time observer-based state feedback for feedrate optimization [27]. Kim et al. used a window-to-window linear regression model, online, to improve servo-error prediction for feedrate optimization, thereby mitigating the mismatch between model estimates and actual measurements [28]. This work was further extended by using a Bayesian regression framework to quantify the model confidence [29].

Despite the growing research in online-learning-based feedrate optimization, to the best of the authors' knowledge, few studies have explored online learning across different machining tasks to iteratively optimize feedrate. One of the few relevant works was proposed by Rattunde et al. [30], which used Gaussian Processes to perform iterative feedrate scheduling under spindle power constraints. In addition, recent work by the present authors [31], proposed a part-to-part learning model for servo

 doi:10.20944/preprints202603.2368.v1

error prediction. The learning model gradually increased its confidence as more data were gathered online and was applied to feedrate optimization, progressively increasing the feedrate. However, the prior work focused only on learning contour errors for identical toolpaths. It did not consider cutting forces or non-identical toolpaths. Therefore, building on the method proposed in [31], this paper makes the following original contributions:

1. It proposes an online pass-to-pass learning model that iteratively learns the servo error and spindle torque (proportional to tangential cutting force) from past machining tasks with similar but non-identical contours.
2. It develops a feedrate optimization framework that utilizes the proposed pass-to-pass learning model to iteratively optimize the feedrate of CNC machine feed drives subject to kinematic, servo error, and spindle torque constraints.
3. It presents experimental 2.5D contour-machining case studies showing that the proposed pass-to-pass feedrate optimization autonomously optimizes feedrate, yielding up to 16.4% improvement in productivity over the method in [31] due to its ability to handle non-identical passes.

The outline of the rest of the paper is as follows. Section 2 introduces the proposed pass-to-pass learning model for servo error and spindle torque predictions, along with the contour matching technique used for non-identical contours in contour machining. Section 3 describes the overall feedrate optimization framework, including the overall formulation of the optimization problem and the remedy for infeasibility. Then, experimental validation and discussion of the results are included in Section 4, followed by the conclusion and future work in Section 5.

2. Pass-to-Pass Learning Model for Servo Error and Spindle Torque Predictions

An overview of the proposed feedrate optimization based on pass-to-pass learning is presented in Figure 1. For a given part geometry, the pass-to-pass learning model (within the digital twin (DT) framework) uses the measurements from past machining passes to learn the machine's contour error and spindle torque and to optimize the feedrate for the current machining pass. After the current pass is completed, the contour error and spindle torque are measured and used to update the learning model and optimize the feedrate for the next machining pass.

This section describes the proposed learning models for predicting contour error and spindle torque. Note that the scope of this paper will focus on 2.5-dimensional (2.5D) contour milling on a single machine tool using the same type of endmill with a fixed axial (z axis) depth of cut.

 doi:10.20944/preprints202603.2368.v1

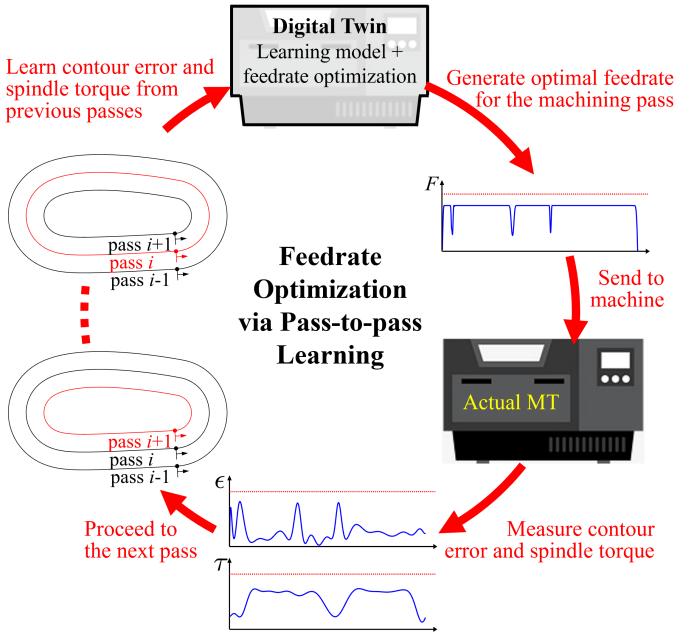

Figure 1. The overview of the feedrate optimization via pass-to-pass learning presented in this paper. The proposed method iteratively learns machine behavior, optimizes feedrate, and collects measurements, from one machining pass to the next.

2.1. Pass-to-Pass Bayesian Regression Model

Given similar cutting contours, one can expect the contour error and spindle torque to exhibit similar trends throughout the cutting operation. Accordingly, following the part-to-part learning model designed in [31], the pass-to-pass learning models for the contour error ϵ and the spindle torque τ are respectively formulated as

$$\hat{\epsilon} = f_\epsilon(\boldsymbol{\phi}_\epsilon; s) \tag{1}$$

$$\hat{\tau} = f_\tau(\boldsymbol{\phi}_\tau; s) \tag{2}$$

where s is the path variable representing the progress of the cutting path and thereby corresponding to the specific location, $\boldsymbol{\phi}_\epsilon$ includes the features affecting the contour error, while $\boldsymbol{\phi}_\tau$ includes the features associated with the spindle torque.

Then, the path variable s is sampled at critical locations, denoted as s_j, $j = 0, 1, \ldots, N$, and individual models are constructed for the contour error and spindle torque at each sampled location s_j, i.e.,

$$\hat{\epsilon}_j = f_\epsilon(\boldsymbol{\phi}_\epsilon; s_j) = f_{\epsilon_j}(\boldsymbol{\varphi}_{\epsilon_j}) \tag{3}$$

$$\hat{\tau}_j = f_\tau(\boldsymbol{\phi}_\tau; s_j) = f_{\tau_j}(\boldsymbol{\varphi}_{\tau_j}) \tag{4}$$

where $\boldsymbol{\varphi}_\epsilon$ and $\boldsymbol{\varphi}_\tau$ are the active features affecting $\hat{\epsilon}_j$ and $\hat{\tau}_j$, which are subsets of $\boldsymbol{\phi}_\epsilon$ and $\boldsymbol{\phi}_\tau$.

Finally, Bayesian linear regression models are adopted to formulate the learning model as

$$\hat{\epsilon}_j = \boldsymbol{\varphi}_{\epsilon_j}^T \boldsymbol{\beta}_{\epsilon_j} + \eta_\epsilon \tag{5}$$

$$\hat{\tau}_j = \boldsymbol{\varphi}_{\tau_j}^T \boldsymbol{\beta}_{\tau_j} + \eta_\tau \tag{6}$$

where $\boldsymbol{\beta}_{\epsilon_j}$ and $\boldsymbol{\beta}_{\tau_j}$ are the stochastic weights to be learned from the past machining data, and $\eta_\epsilon \sim \mathcal{N}(0, \sigma_{\eta_\epsilon}^2)$ and $\eta_\tau \sim \mathcal{N}(0, \sigma_{\eta_\tau}^2)$ are the error terms that are expected to include measurement noise and unmodeled dynamics that cannot be captured given the model structure.

 doi:10.20944/preprints202603.2368.v1

Remark 1. *Bayesian linear regression is selected in this paper as the learning model, because it can quantify model uncertainty based on prior assumptions, and its linearity facilitates simpler control or optimization. However, other models, such as Gaussian Processes, may be adopted for more complicated scenarios.*

The performance of the pass-to-pass learning model is highly dependent on the selection of the feature design for $\boldsymbol{\varphi}_\epsilon$ and $\boldsymbol{\varphi}_\tau$ as well as the noise variances $\sigma^2_{\eta_\epsilon}$ and $\sigma^2_{\eta_\tau}$. In the case study presented in Section 4, the feature vectors $\boldsymbol{\varphi}_\epsilon$ and $\boldsymbol{\varphi}_\tau$ are designed as

$$\boldsymbol{\varphi}^T_{\epsilon_j} = \begin{bmatrix} \dot{s} & \frac{dx}{ds} & \frac{dy}{ds} & \rho & \alpha \end{bmatrix}\Big|_{s=s_j} \tag{7}$$

$$\boldsymbol{\varphi}^T_{\tau_j} = \begin{bmatrix} \dot{s} & \rho & \alpha \end{bmatrix}\Big|_{s=s_j} \tag{8}$$

where $\dot{s}$ is the time derivative of the path progress, which corresponds to the feedrate of the machine. For the geometry-related features, $\frac{dx}{ds}$ and $\frac{dy}{ds}$ are the derivatives of x- and y-axis reference positions with respect to the path variable s, ρ is the curvature, and α is the radial depth of cut. All of the features are evaluated at the sample location s_j. Note that compared to $\boldsymbol{\varphi}_{\tau_j}$, $\boldsymbol{\varphi}_{\epsilon_j}$ includes two additional features $\frac{dx}{ds}$ and $\frac{dy}{ds}$, as these are mainly to detect the locations of motion reversal, where quadrant glitches occur and have significant effects on the contour error but not on the spindle torque.

Remark 2. *The feature vector design introduced above is based on the characteristics of the machine used for the case studies in Section 4 and is simplified to include only the most critical feedrate- and geometry-related features. However, the design of the feature vector can be tailored to a specific machine or application, such as by including higher-order derivative terms.*

As for the noise variances, since the magnitudes of the unmodeled dynamics are unknown a priori, they are determined adaptively from the prediction errors of past machining tasks. More specifically, the noise variances for the contour error and spindle torque are initially set to relatively small values. Then, as more contours are machined, the deviations between the prediction and the measurements are recorded, from which the root mean squares (RMS) prediction errors are calculated. Finally, among the RMS prediction errors at all sample points, the maximum value is selected as the noise variance. That is, suppose that the sample point s_j has the highest RMS prediction errors. Then, the noise variance for all sample points are set to

$$\sigma_{\eta_\epsilon} = \text{rms}(\epsilon_j - \boldsymbol{\varphi}^T_{\epsilon_j}\boldsymbol{\beta}_{\epsilon_j}) \tag{9}$$

$$\sigma_{\eta_\tau} = \text{rms}(\tau_j - \boldsymbol{\varphi}^T_{\tau_j}\boldsymbol{\beta}_{\tau_j}) \tag{10}$$

The prior distribution of the model weights in this paper are designed based on σ_{η_ϵ} and σ_{η_τ}. In particular, the priors are defined as $\boldsymbol{\beta}_{\epsilon_j,0} \sim \mathcal{N}(0, \sigma^2_{\beta_{\epsilon_j},0}\mathbf{I})$ and $\boldsymbol{\beta}_{\tau_j,0} \sim \mathcal{N}(0, \sigma^2_{\beta_{\tau_j},0}\mathbf{I})$, where the prior variances $\sigma^2_{\beta_{\epsilon_j},0}$ and $\sigma^2_{\beta_{\tau_j},0}$ are set according to the fixed, predefined ratio of $\frac{\sigma^2_{\beta_{\epsilon_j},0}}{\sigma^2_{\eta_\epsilon}}$ and $\frac{\sigma^2_{\beta_{\tau_j},0}}{\sigma^2_{\eta_\tau}}$, which are related to the regularization factor in ridge regression [32].

Once the prior distributions of the model weights and noise variances are determined, the posterior distributions of the model weights are computed using the conventional Bayesian update formula [32]. For example, the posterior of the contour error model weights at location s_j, denoted as $\boldsymbol{\beta}_{\epsilon_j,1} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\beta}_{\epsilon_j,1}}, \boldsymbol{\Sigma}_{\beta_{\epsilon_j,1}})$, is computed as follows:

$$\begin{aligned} \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{\epsilon_j,1}} &= (\boldsymbol{\Sigma}^{-1}_{\boldsymbol{\beta}_{\epsilon_j,0}} + \frac{1}{\sigma^2_{\eta_\epsilon}}\boldsymbol{\Phi}_{\epsilon_j}\boldsymbol{\Phi}^T_{\epsilon_j})^{-1} \\ \boldsymbol{\mu}_{\boldsymbol{\beta}_{\epsilon_j,1}} &= \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{\epsilon_j,1}}(\boldsymbol{\Sigma}^{-1}_{\boldsymbol{\beta}_{\epsilon_j,0}}\boldsymbol{\mu}_{\boldsymbol{\beta}_{\epsilon_j,0}} + \frac{1}{\sigma^2_{\eta_\epsilon}}\boldsymbol{\Phi}_{\epsilon_j}\boldsymbol{\epsilon}_j) \end{aligned} \tag{11}$$

 doi:10.20944/preprints202603.2368.v1

where $\boldsymbol{\epsilon}_j$ includes the contour error measurements at s_j for all previous passes, and $\boldsymbol{\Phi}_{\epsilon_j}$ is a concatenated matrix composed of the corresponding feature vectors $\boldsymbol{\varphi}_\epsilon$.

In order to quickly initialize training, i.e., to prevent overfitting during the initial phase, and to enable the usage of the trained model in optimization, augmentation of the training samples is introduced. Note that adding more sample points s_j may not be desired, as it increases the number of models as defined in Equations (1) and (2) rather than increasing the amount of training samples per model. Specifically, ϵ_j is augmented with the data from its neighboring points, e.g., the points within the intervals of

$$s_{j,aug} \in \left(\frac{s_{j-1} + s_j}{2}, \frac{s_j + s_{j+1}}{2} \right). \tag{12}$$

After training, the prediction of the contour error at s_j for any given machining settings is calculated by

$$\begin{aligned} \mu_{\epsilon_j,1} &= \boldsymbol{\varphi}_{\epsilon_j}^T \boldsymbol{\mu}_{\boldsymbol{\beta}_{\epsilon_j,1}} \\ \sigma_{\epsilon_j,1}^2 &= \boldsymbol{\varphi}_{\epsilon_j}^T \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{\epsilon_j,1}} \boldsymbol{\varphi}_{\epsilon_j} + \sigma_{\eta_\epsilon} \end{aligned} \tag{13}$$

Finally, the predicted mean and the credible intervals, which are defined as 3-sigma of the standard deviation, are given by

$$\hat{\epsilon}_j + \delta\hat{\epsilon}_j = \mu_{\epsilon_j,1} + 3\sigma_{\epsilon_j,1} \tag{14}$$

The same procedure is repeated independently for all sampled locations $j = 0, 1, \ldots, N$, as well as for the training of the spindle torque model weights β_{τ_j} and the prediction of the spindle torque $\hat{\tau}_j + \delta\hat{\tau}_j$.

2.2. Application to Non-identical Cutting Passes

The pass-to-pass learning model presented in 2.1 is similar to the part-to-part learning model proposed in [31]. However, an additional step is required for the pass-to-pass learning model because the path variable s used in [31] does not necessarily indicate the same section of the contour across non-identical cutting passes. For example, when moving along an arc, the outer contour has to travel a longer distance than the inner contour. Therefore, given the same value of s, the corresponding location on the outer contour actually lags behind that of the inner contour, as demonstrated in Figure 2(a). This mismatch in the geometry feature can result in incorrect transfer of learning across different passes, degrading both model prediction accuracy and optimization performance.

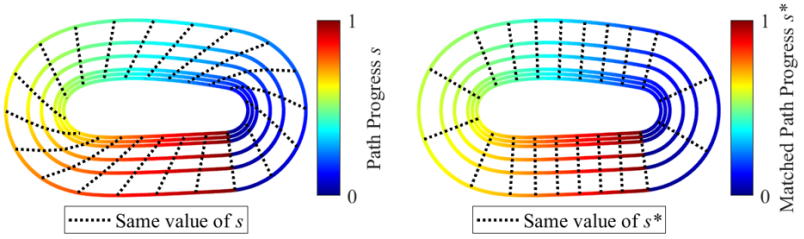

Figure 2. The comparison of the corresponding locations on the iso-contour machining passes given the same values of (a) path variable without contour matching s and (b) path variable with contour matching s^*.

 doi:10.20944/preprints202603.2368.v1

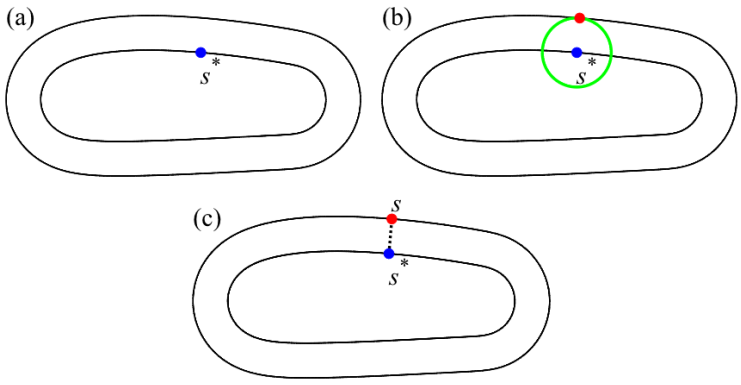

Figure 3. The illustration of the contour matching process between two iso-contour machining passes, which includes (a) identifying a sample location on the inner contour, (b) finding the closest point on the outer contour, and (c) constructing an s–s^* lookup table.

Therefore, to overcome this issue, this section proposes a solution to correctly identify the matched sections with similar geometric features. The general idea of the solution is depicted in Figure 3. First, the sample locations are defined on the innermost contour and denoted by s^*, since the innermost contour directly affects the quality of the final part. Second, for each sample point on the innermost contour, the closest point on each outer contour is identified, with the corresponding path variable s recorded. Finally, an s–s^* lookup table is constructed to map between the original s values on the outer contours and the corresponding s^* values on the innermost contour. Once the s–s^* lookup table is obtained, the terms related to the path variable s in Equations (3) and (4) are rewritten, based on the matched path variable s^*, as

$$\hat{\epsilon}_j = f_\epsilon(\boldsymbol{\phi}_\epsilon; s_j^*) = f_{\epsilon_j}(\boldsymbol{\varphi}_{\epsilon_j}) \tag{15}$$

$$\hat{\tau}_j = f_\tau(\boldsymbol{\phi}_\tau; s_j^*) = f_{\tau_j}(\boldsymbol{\varphi}_{\tau_j}) \tag{16}$$

while the corresponding feature vectors $\boldsymbol{\varphi}_\epsilon$ and $\boldsymbol{\varphi}_\tau$, originally defined in Equations (7) and (8), are also re-defined based on s_j^* as

$$\boldsymbol{\varphi}_{\epsilon_j}^T = \left[\dot{s} \quad \frac{dx}{ds} \quad \frac{dy}{ds} \quad \rho \quad \alpha\right]\Big|_{s^*=s_j^*} \tag{17}$$

$$\boldsymbol{\varphi}_{\tau_j}^T = \left[\dot{s} \quad \rho \quad \alpha\right]\Big|_{s^*=s_j^*} \tag{18}$$

The result of contour matching is presented in Figure 2(b), which shows that sections with similar geometric features, e.g., straight lines or arcs, are grouped together.

3. Framework of Iterative Feedrate Optimization

Once the learning model is constructed, the feedrate optimization (FO) problem can be formulated. The FO framework used in this paper is similar to the window-based linear programming strategy proposed in [33], which partitions the linear programming-based FO problem into multiple overlapping windows and solves them sequentially.

3.1. Window-based Feedrate Optimization

The goal of FO is to maximize the path progress s of the contour subject to kinematic, contour error, and spindle torque constraints. In particular, for contour error and spindle torque constraints, credible intervals are also considered, as they reflect the learning model's confidence and may help prevent constraint violations.

 doi:10.20944/preprints202603.2368.v1

To this end, given the window size N_w, as well as the feedrate limit F_{max}, acceleration limit A_{max}, jerk limit J_{max}, contour error limit E_{max}, and spindle torque limit T_{max}, the FO problem is formulated as follows:

$$\begin{aligned}
&\min_{s} \sum_{k=0}^{N_w} -s(k) = -\mathbf{1}^T \boldsymbol{s} \\
&\text{subject to} \\
&s(k-1) \leq s(k) \\
&0 \leq L\frac{D[\boldsymbol{s}]}{T_s} \leq F_{max} \\
&\left|\frac{D^2[\boldsymbol{x}_d(s)]}{T_s^2}\right|, \left|\frac{D^2[\boldsymbol{y}_d(s)]}{T_s^2}\right|, \left|\frac{D^2[\boldsymbol{s}]}{T_s^2}\right| \leq A_{max} \\
&\left|\frac{D^3[\boldsymbol{x}_d(s)]}{T_s^3}\right|, \left|\frac{D^3[\boldsymbol{y}_d(s)]}{T_s^3}\right|, \left|\frac{D^3[\boldsymbol{s}]}{T_s^3}\right| \leq J_{max} \\
&\hat{\boldsymbol{\epsilon}}(s) + \delta\hat{\boldsymbol{\epsilon}}(s) \leq E_{max} \\
&\hat{\boldsymbol{\tau}}(s) + \delta\hat{\boldsymbol{\tau}}(s) \leq T_{max}
\end{aligned} \tag{19}$$

where k denotes the local time step within the current window, $\mathbf{1}$ is a column vector whose values are all equal to one, L is the total length of the current cutting pass, T_s is the sampling time, D is the difference operator used to compute time derivatives, while $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\tau}}$ are the concatenated vectors of contour error and spindle torque for all sampled locations s_j^*, i.e.,

$$\hat{\boldsymbol{\epsilon}} = \begin{bmatrix} \hat{\epsilon_0} & \hat{\epsilon_1} & \dots & \hat{\epsilon_N} \end{bmatrix}^T \tag{20}$$

$$\hat{\boldsymbol{\tau}} = \begin{bmatrix} \hat{\tau_0} & \hat{\tau_1} & \dots & \hat{\tau_N} \end{bmatrix}^T \tag{21}$$

where ϵ_j and τ_j are defined in Equations (15) and (16).

To reduce the size of the optimization problem and to ensure continuity of the optimized results across windows without explicitly including equality constraints, $\boldsymbol{s}$ in Equation (19) is further expressed as a linear combination of B-spline basis functions [13,23], as shown below:

$$\boldsymbol{s} = \boldsymbol{\Psi}_{dn,c}\boldsymbol{p}_c + \boldsymbol{\Psi}_{dn,p}\boldsymbol{p}_p \tag{22}$$

where $\boldsymbol{p}_c$ is a vector consisting of the control point coefficients within the current window that are to be optimized by solving Equation (19), $\boldsymbol{p}_p$ is the vector of coefficients from the past window that have residual effects on the current window, while $\boldsymbol{\Psi}_{dn,c}$ and $\boldsymbol{\Psi}_{dn,p}$ are the corresponding B-spline basis function matrices for down-sampling. Lastly, to prevent the window-based optimization from becoming too greedy, we update only the first half ($N_w/2$ time steps) of the optimized results $\boldsymbol{s}_{opt}$ after solving and reconstructing it from Equation (19) and Equation (22), respectively.

It is worth noting that some terms, such as the time-derivative terms in the x- and y-reference frames, are nonlinear in s due to the nonlinear mapping from s to x and y. Hence, these terms need

 doi:10.20944/preprints202603.2368.v1

to be linearized about a predefined equilibrium $\boldsymbol{s}_{eq}$. For example, $\frac{D^2[\boldsymbol{x}_d(s)]}{T_s^2}$ is linearized by (with all multiplication being element-wise):

$$\begin{aligned}\frac{D^2[\boldsymbol{x}_d(s)]}{T_s^2} &= \frac{d^2\boldsymbol{x}}{ds^2}\dot{\boldsymbol{s}}^2 + \frac{d\boldsymbol{x}}{ds}\ddot{\boldsymbol{s}} \\ &\approx \frac{d^2x}{ds^2}\dot{s}^2 + \frac{dx}{ds}\ddot{s}\Big|_{\boldsymbol{s}=\boldsymbol{s}_{eq}} \\ &+ \left(\frac{d^3x}{ds^3}\dot{s}^2 + \frac{d^2x}{ds^2}\ddot{s}\right)\Big|_{\boldsymbol{s}=\boldsymbol{s}_{eq}} (\boldsymbol{s}-\boldsymbol{s}_{eq}) \\ &+ \left(2\frac{d^2x}{ds^2}\right)\Big|_{\boldsymbol{s}=\boldsymbol{s}_{eq}} (\dot{\boldsymbol{s}}-\dot{\boldsymbol{s}}_{eq}) \\ &+ \left(\frac{dx}{ds}\right)\Big|_{\boldsymbol{s}=\boldsymbol{s}_{eq}} (\ddot{\boldsymbol{s}}-\ddot{\boldsymbol{s}}_{eq})\end{aligned} \tag{23}$$

where notably $\dot{s}$ and $\ddot{s}$ are also functions of s and thus need to be included in the multivariate linearization.

A similar procedure is repeated for higher-order derivatives, $\frac{D^3[\boldsymbol{x}_d(s)]}{T_s^3}$, and the derivatives for y reference, $\frac{D^2[\boldsymbol{y}_d(s)]}{T_s^2}$ and $\frac{D^3[\boldsymbol{y}_d(s)]}{T_s^3}$.

In addition, for contour error $\hat{\boldsymbol{\epsilon}}(s)$ and spindle torque $\hat{\boldsymbol{\tau}}(s)$ predictions, since $\boldsymbol{s}$ is not known prior to FO and $\dot{s}$ appears in the feature vector, as described in Equations (17) and (18), $\dot{\boldsymbol{s}}$ also needs to be approximated by the time derivative of the equilibrium $\dot{\boldsymbol{s}}_{eq}$. For the feedrate at location j, $\dot{s}(s_j)$ is approximated by linearly interpolating $\dot{\boldsymbol{s}}$ among $\boldsymbol{s}_{eq}$ and finding the corresponding values at s_j, i.e.,

$$\begin{aligned}\dot{s}(s_j) =& \frac{s_{eq,j^+} - s_j}{s_{eq,j^+} - s_{eq,j^-}} \cdot \dot{s}(s_{eq,j^-}) \\ &+ \frac{s_j - s_{eq,j^-}}{s_{eq,j^+} - s_{eq,j^-}} \cdot \dot{s}(s_{eq,j^+})\end{aligned} \tag{24}$$

where s_{eq,j^+} and s_{eq,j^-} are the two closest neighboring equilibrium points in $\boldsymbol{s}_{eq}$ that surround the sampled location s_j. Lastly, for the credible intervals of the contour error $\delta\hat{\boldsymbol{\epsilon}}(s)$ and spindle torque $\delta\hat{\boldsymbol{\tau}}(s)$, because they involve the square root of $\sigma^2_{\epsilon_j,1}$ and $\sigma^2_{\tau_j,1}$ and depend on $\dot{s}$, these terms also need to be linearized and approximated using a similar approach as in Equations (23) and (24).

Once all linearization steps are complete, sequential linear programming (SLP) is adopted to solve the FO problem [33]. SLP is performed by first computing an initial equilibrium trajectory $\boldsymbol{s}_{eq}$ using a trapezoidal acceleration profile with conservative kinematic limits. The conservative kinematic limits are selected to help ensure that the constraints are satisfied (though they cannot guarantee this). Then, all constraints are linearized or approximated with respect to the predetermined $\boldsymbol{s}_{eq}$. Next, linear programming is performed to obtain an optimized solution $\boldsymbol{s}_{opt}$ to the linearized problem. Lastly, the optimized solution $\boldsymbol{s}_{opt}$ is used as the new equilibrium $\boldsymbol{s}_{eq}$ for the next iterations and this process is repeated until convergence between $\boldsymbol{s}_{opt}$ and $\boldsymbol{s}_{eq}$, which can be quantified by the root mean squares of their difference. Note that, unlike the authors' prior work [31] (which did not make use of a windowed SLP), no constraint on the step size is imposed since the window length is relatively small and large deviations between $\boldsymbol{s}_{opt}$ and $\boldsymbol{s}_{eq}$ are not expected.

3.2. Generation of Backup Solution

The window-based FO problem, as presented in Equation (19), is not guaranteed to be feasible. Infeasibility can occur due to the optimization strategy or the model's predictions. For the optimization strategy, linearization distorts the boundary of the feasible region defined by the original nonlinear

 doi:10.20944/preprints202603.2368.v1

constraints, which can potentially eliminate the feasible regions. In rare cases, the window-based strategy can still result in greedy optimization, where a solution cannot be found based on results from the prior window, even with window overlap. For model prediction, in the early stage of training, the pass-to-pass learning model may not be sufficiently confident to return predictions whose credible intervals fall within the constraints. All of these possibilities motivate the introduction of backup solutions to ensure the functioning and robustness of the overall FO framework. In addition, the backup solution can also serve as the initial equilibrium solution for the SLP algorithm.

The idea of the backup solution is to generate a trajectory analytically, using a trapezoidal acceleration profile with conservative path-level kinematic constraints F_{con}, A_{con}, and J_{con}, and without considering the remaining nonlinear constraints, including axis-level kinematic constraints, contour error constraints, or spindle torque constraints. While multiple methods are available to construct the backup solution, this section presents the one used in the experimental validation in Section 4.

In this paper, the backup solutions are categorized into two types: (1) reaching the predefined constant feedrate F_{con}, or (2) reaching the predefined end point S_F with zero feedrate and zero acceleration, as shown in Figures 4 and 5 respectively, depending on the remaining distance to the end of the path.

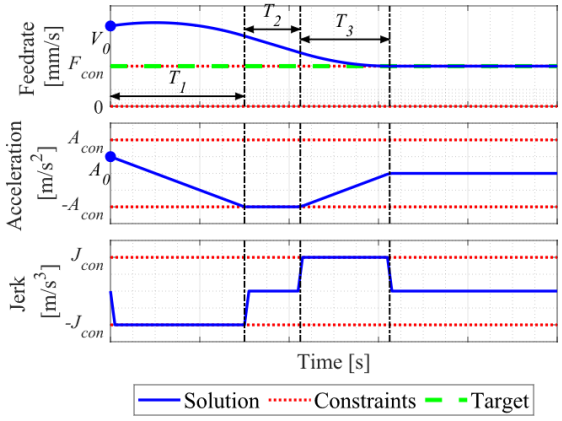

Figure 4. An example of the backup solution to reach a predefined constant feedrate. It consists of three sections: increasing deceleration, maximum deceleration, and decreasing deceleration.

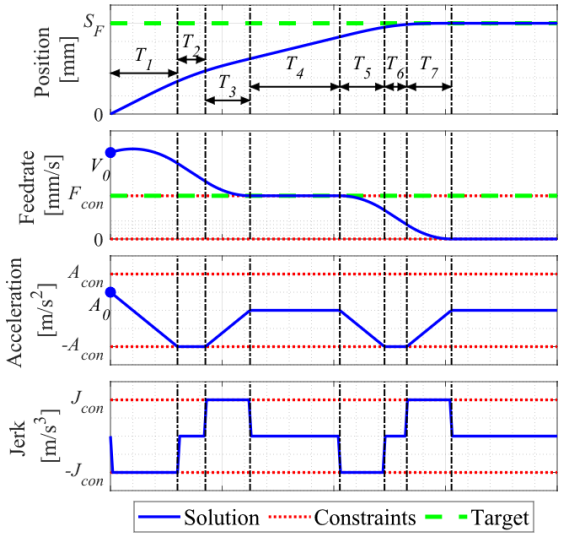

Figure 5. An example of the backup solution to reach a predefined end point with zero feedrate and zero acceleration. It first reaches the predefined feedrate, then maintain the constant feedrate for a specified duration, and finally decelerates to zero feedrate.

 doi:10.20944/preprints202603.2368.v1

For the first type of backup solution, i.e., reaching a constant feedrate, assume that the initial feedrate and acceleration are F_0 and A_0. As shown in Figure 4, the trapezoidal motion profile is composed of three sections – increasing deceleration, maximum deceleration, and decreasing deceleration. In the first section, the jerk is set to its maximum limit J_{con} to reduce the acceleration until the maximum deceleration magnitude A_{con} is reached. Then, in the second section, the maximum deceleration A_{con} is maintained to reduce the feedrate as quickly as possible. Lastly, the jerk is again set to its limit J_{con} to drive the acceleration to zero while driving the feedrate to the desired value F_{con}. In this scenario, the time duration for each section is calculated as follows:

$$\begin{aligned} T_1 &= \frac{A_{con} + A_0}{J_{con}} \\ T_2 &= \frac{F_0 - F_{con}}{A_{con}} + \frac{A_0^2 - 2A_{con}^2}{2A_{con}J_{con}} \\ T_3 &= \frac{A_{con}}{J_{con}} \end{aligned} \tag{25}$$

Then, the acceleration, feedrate, and position profiles with respect to time t for each section can be calculated by

$$\begin{aligned} A(t) &= A(t_{p-1}) + J_p\bar{t}_p \\ F(t) &= F(t_{p-1}) + A(t_{p-1})\bar{t}_p + \frac{1}{2}J_p\bar{t}_p^2 \\ S(t) &= S(t_{p-1}) + F(t_{p-1})\bar{t}_p + \frac{1}{2}A(t_{p-1})\bar{t}_p^2 + \frac{1}{6}J_p\bar{t}_p^3 \end{aligned} \tag{26}$$

where t_{p-1} denotes time stamp at the beginning of section p, and $\bar{t}_p$ is the relative time stamp with respect to t_{p-1} during section p, i.e.,

$$\begin{aligned} t_{p-1} &= \sum_{q=1}^{p-1} T_q \\ \bar{t}_p &= t - t_{p-1}, \quad t \in (t_{p-1}, t_p] \end{aligned} \tag{27}$$

However, exceptions to this motion profile may occur. For example, when the initial feedrate or initial acceleration is so high that there is a risk of violating the original feedrate constraint V_{max}, which can adversely affect cutting, then a jerk magnitude exceeding J_{con} may be applied to avoid the violation. Similarly, when the initial feedrate is close to zero, and the acceleration is too low that it may cause backward motion, then a jerk magnitude exceeding J_{con} may also be applied to prevent the reversal.

For the second type of backup solution, i.e., reaching a fixed endpoint, the motion profile is constructed based on the first-type backup solution. More specifically, the motion profile first drives the feedrate to the predefined value F_{con} using the motion profile described in Equation (25), with the corresponding travel distance recorded as S_{con}. Then, it maintains the constant feedrate for a time duration of T_4, which is calculated by

$$T_4 = \frac{S_F - S_{con}}{V_{con}} - \frac{V_{con}}{2A_{con}} - \frac{A_{con}}{2J_{con}} \tag{28}$$

 doi:10.20944/preprints202603.2368.v1

Finally, the motion profile, by utilizing the first-type backup solution again, aims to drive the feedrate and acceleration to zero. The corresponding time durations for the increasing deceleration, maximum deceleration, and decreasing deceleration sections are given as

$$\begin{aligned} T_5 &= \frac{A_{con}}{J_{con}} \\ T_6 &= \frac{V_{con}}{A_{con}} - \frac{A_{con}}{J_{con}} \\ T_7 &= \frac{A_{con}}{J_{con}} \end{aligned} \tag{29}$$

Exceptions may also be encountered here. For example, a major exception to the second-type backup solution occurs when the sum of the travel distances during T_1–T_3 and T_5–T_7 exceeds S_F, making T_4 negative. In this scenario, the motion profile first drives the feedrate and acceleration to zero, i.e., following Equation (25) but replacing V_{con} with 0, with the corresponding traveled distance recorded as S_{stop}. Then, a conventional S-curve motion profile is appended to compensate for the difference between S_{stop} and S_F.

Once the time-domain backup solution is obtained, it is then fitted using B-spline basis functions as formulated in Equation (23). However, fitting only with respect to position level will incur large oscillations in higher-order derivatives, such as acceleration and jerk, while fitting only with respect to jerk will result in steady-state bias in position and feedrate, as shown in Figure 6. Therefore, to resolve this issue, the backup solution is fitted at multiple levels, including position, velocity, acceleration, and jerk levels, i.e.,

$$\begin{aligned} \boldsymbol{p}_{c,0} &= \boldsymbol{\Psi}_{dn,c}^{\dagger}\left(\boldsymbol{s}_{backup} - \boldsymbol{\Psi}_{dn,p}\boldsymbol{p}_p\right) \\ \boldsymbol{p}_{c,1} &= \dot{\boldsymbol{\Psi}}_{dn,c}^{\dagger}\left(\dot{\boldsymbol{s}}_{backup} - \dot{\boldsymbol{\Psi}}_{dn,p}\boldsymbol{p}_p\right) \\ \boldsymbol{p}_{c,2} &= \ddot{\boldsymbol{\Psi}}_{dn,c}^{\dagger}\left(\ddot{\boldsymbol{s}}_{backup} - \ddot{\boldsymbol{\Psi}}_{dn,p}\boldsymbol{p}_p\right) \\ \boldsymbol{p}_{c,3} &= \dddot{\boldsymbol{\Psi}}_{dn,c}^{\dagger}\left(\dddot{\boldsymbol{s}}_{backup} - \dddot{\boldsymbol{\Psi}}_{dn,p}\boldsymbol{p}_p\right) \end{aligned} \tag{30}$$

where $\dot{\boldsymbol{\Psi}}$, $\ddot{\boldsymbol{\Psi}}$, and $\dddot{\boldsymbol{\Psi}}$ are the time derivatives of the B-spline basis function matrices.

Then, the fitting results based on lower order terms are filtered with a low-pass filter, while those fitted based on higher order terms are filtered with a high-pass filter, and the filtered results are combined by

$$\boldsymbol{p}_c = G_0 \cdot \boldsymbol{p}_{c,0} + G_1 \cdot \boldsymbol{p}_{c,1} + G_2 \cdot \boldsymbol{p}_{c,2} + G_3 \cdot \boldsymbol{p}_{c,3} \tag{31}$$

where G_0, G_1, G_2, and G_3 must sum to 1. In this paper, without loss of generality, they are designed as

$$\begin{aligned} G_0(s) &= \frac{\omega_c^3}{s^3 + 3\omega_c s^2 + 3\omega_c^2 s + \omega_c^3} \\ G_1(s) &= \frac{3\omega_c^2 s}{s^3 + 3\omega_c s^2 + 3\omega_c^2 s + \omega_c^3} \\ G_2(s) &= \frac{3\omega_c s^2}{s^3 + 3\omega_c s^2 + 3\omega_c^2 s + \omega_c^3} \\ G_3(s) &= \frac{s^3}{s^3 + 3\omega_c s^2 + 3\omega_c^2 s + \omega_c^3} \end{aligned} \tag{32}$$

where ω_c is the design variable, indicating the cutoff frequency.

The overall framework of the iterative feedrate optimization is summarized in Figure 7.

 doi:10.20944/preprints202603.2368.v1

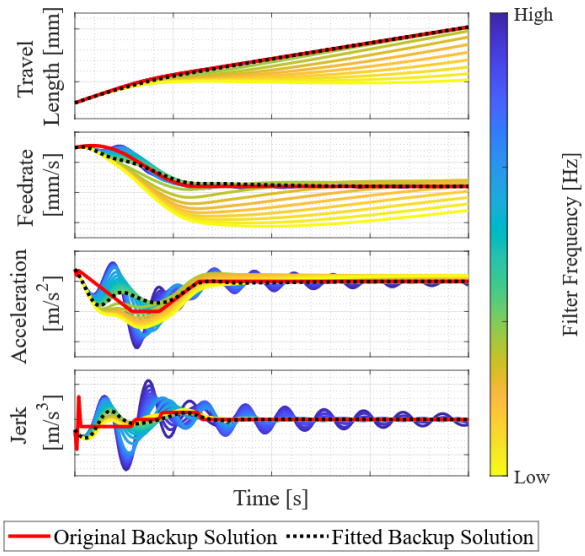

Figure 6. An example of backup solution filtering with different filter cutoff frequencies. Higher cutoff frequencies tend to rely more on the fitting of position and feedrate, causing larger oscillations in acceleration and jerk, while lower cutoff frequencies tend to prioritize acceleration and jerk fitting, causing steady-state error in position and feedrate.

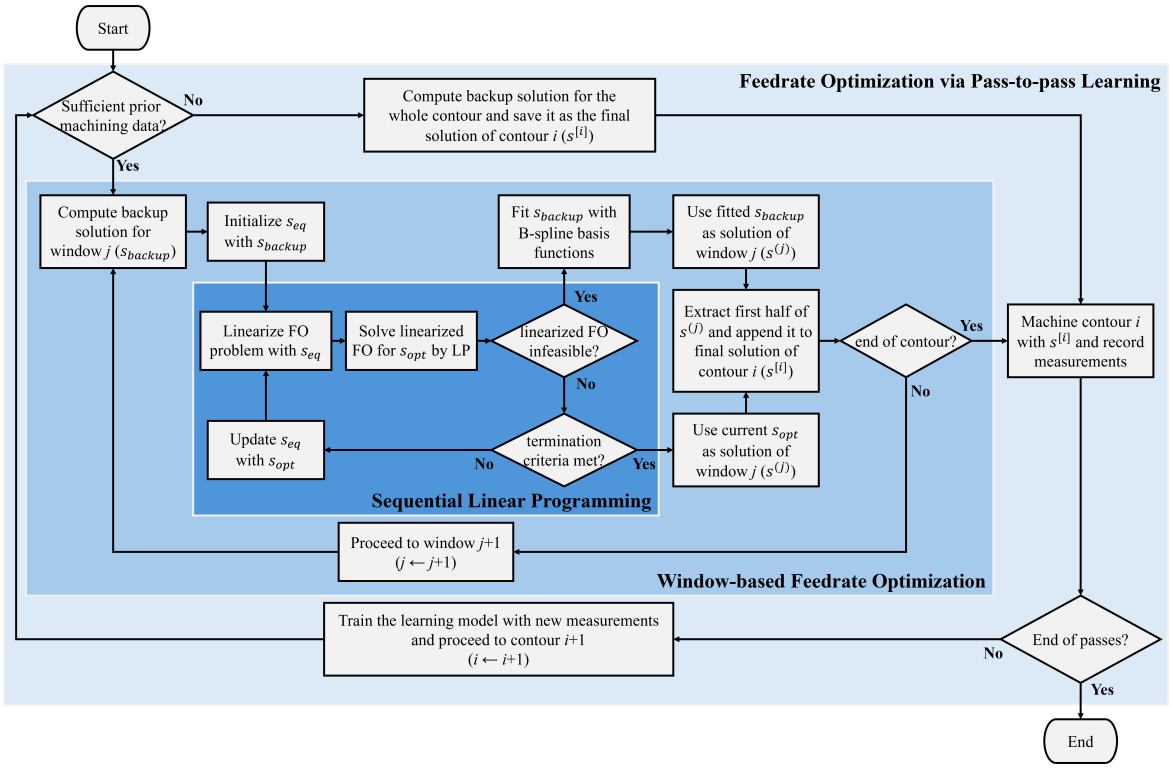

Figure 7. The overall framework of the proposed feedrate optimization via pass-to-pass learning. It consists of several iteration loops. The sequential linear programming (SLP) loop iteratively solves the linearized problem to approach the solution of the nonlinear FO problem. The window-based FO loop iteratively finds the optimal feedrate for each overlapping window. The pass-to-pass learning loop iteratively updates the learning model and applies to next machining task.

4. Experimental Validation

To validate the proposed framework for FO via pass-to-pass learning, experimental case studies are conducted on a 3-axis desktop CNC milling machine tool to contour-machine three different prismatic parts. The experiments utilize the 2.5D milling process, in which the tool follows a 2D

 doi:10.20944/preprints202603.2368.v1

motion profile consisting of multiple passes with a fixed depth of cut, then proceeds to the next z-level after each level is completed. The method proposed in this paper is further compared with the one proposed in the authors' prior work [31], i.e., FO via part-to-part learning, with a modification to its FO framework that applies a window-based strategy.

4.1. Experimental Setup

The experimental case studies are performed on a Carbide 3D Nomad 3 Desktop CNC Mill, as used in [28,29,31]. The desktop CNC machine is actuated by stepper motors for the three motion axes, and the in-plane (x- and y-axis) positions are measured by two externally attached Renishaw RKLC20-S optical linear encoders with 5 μm resolution. In addition, to enable higher cutting force and obtain spindle torque measurements, the spindle motor is replaced with an Aerotech BMS60 brushless motor, which has a torque constant of 0.20 $\mathrm{Nm/A_{rms}}$ and is equipped with a rotary encoder. Finally, the type of endmill used throughout the case studies is a Carbide 3D #102 .125" flat-end cutter, while the material of the workpiece is plastic (Acetal Resin).

To command the machine with a customized trajectory profile, a dSPACE DS1007 Real-Time Interface Platform running at 1 kHz is used, along with a dSPACE DS5203 FPGA Board and a DS3002 Incremental Encoder Interface Board, to bypass the original controller board on the Nomad 3. The stepper motors are then driven by DRV8825 stepper motor drivers at 40 kHz, and the spindle motor is driven by Aerotech's Soloist CP10 servo driver. Furthermore, an integral controller with $K_i = 50$ is added to compensate the steady-state error of the stepper motors, while the spindle motor are controlled by a PI controller with $K_{vp} = 0.01667\ \frac{\mathrm{A}}{\mathrm{rev/s}}$ and $K_{vi} = 1.4710\ \frac{\mathrm{A}}{\mathrm{rev}}$ and an acceleration feedforward $K_{af} = 8.7965 \cdot 10^{-5}\ \frac{\mathrm{A}}{\mathrm{rev/s^2}}$, which generates a current reference for the Soloist servo driver to track. The experimental setup is summarized in Figure 8.

4.2. Experimental Tasks

The three different geometries used in the experimental case studies are an airfoil shape, a tensile test specimen, and a spur gear, as shown in Figure 9. For each prismatic part, the part height is 4.5 mm. With a fixed cut depth of 1.5 mm, the machine repeats an identical in-plane cutting contour at three different z-levels, denoted as Z-1.5, Z-3.0, and Z-4.5. Each z-level consists of six non-identical iso-contour cutting passes – two roughing passes, two semi-finishing passes, and two finishing passes.

For all passes, the kinematic constraints are set as

$$\begin{aligned} V_{max} &= 15\ \mathrm{mm/s} \\ A_{max} &= 500\ \mathrm{mm/s^2} \\ J_{max} &= 25\ \mathrm{m/s^3} \end{aligned} \tag{33}$$

while the spindle torque constraint is set as

$$T_{max} = 0.075\ \mathrm{Nm} \tag{34}$$

which is equivalent to a voltage constraint of $V_{max} = 0.375$ V given the torque constant of the servo motor and the signal-current ratio of 1V/1A in the Soloist software setting. For the contour error constraints, however, the limits differ across cutting passes. For the roughing passes, since machining accuracy is less relevant to part quality, the error tolerance is set higher. Conversely, during the finishing stage, the error tolerance is the most stringent. The contour error constraints are given as follows:

$$\begin{aligned} E_{max} &= 45\ \mu\mathrm{m} \quad && \text{(for roughing)} \\ E_{max} &= 30\ \mu\mathrm{m} \quad && \text{(for semi-finishing)} \\ E_{max} &= 15\ \mu\mathrm{m} \quad && \text{(for finishing)} \end{aligned} \tag{35}$$

 doi:10.20944/preprints202603.2368.v1

For FO via pass-to-pass learning, the main objective is to learn the cutting behavior from one pass to the next, even if the passes are non-identical. In practice, to initialize the pass-to-pass learning model, the first z-level (Z-1.5) uses a conservative feedrate (40% of the kinematic constraints defined in Equation (33) throughout all passes to learn the geometric components of the model. Then, the first pass of the Z-3.0 level uses a slightly higher conservative feedrate, i.e., 45% of the kinematics constraints defined in Equation (33), to learn the speed component of the model. Afterwards, for all remaining passes, the pass-to-pass learning model is used to predict the contour error and spindle torque, as described in Section 2, which are then applied to the FO framework described in Section 3.

In contrast, the part-to-part learning model presented in [31] can only learn the cutting behavior from identical cutting geometry. Therefore, in this case study of contour machining of prismatic parts, the part-to-part learning model learns the cutting behavior independently for each pass between successive z-levels. (Therefore, it is hereafter referred to as level-to-level learning). Accordingly, both Z-1.5 and Z-3.0 levels have to use conservative feedrates, i.e., 40% and 45% of the kinematics constraints defined in Equation (33), respectively, to initialize the level-to-level learning model. In other words, only at the final z-level (Z-4.5) can the contour error and the spindle torque for each individual pass be predicted by the level-to-level learning models that are trained on the corresponding identical passes from the previous z-levels.

For the parameter settings of the pass-to-pass learning and level-to-level learning models, as well as the FO algorithm, the trajectory sampling time is $T_s = 1$ ms. To down-sample the trajectory using Equation (22), fifth-order B-spline basis functions are used, with the control point frequency set to 100 Hz. The sample locations s_j^*, $j = 0, 1, \ldots, N$, with $N = 101$, are selected to be 101 uniformly distributed points on the innermost contour. For the augmented training sample, a total of 501 uniformly distributed points are selected and assigned to individual models based on the intervals presented in Equation (12). For fair comparison, both the pass-to-pass learning and level-to-level learning models have identical sample points and augmentation settings. The window size for the overlapping window is set to be $N_w = 200$. The stopping criteria for the SLP are either $\text{rms}(\boldsymbol{s}_{opt} - \boldsymbol{s}_{eq}) \leq 10^{-6}$, the number of iteration exceeding 20 iterations, or a backup solution being used due to infeasibility. The end of a contour is defined as the fact that the last point of $\boldsymbol{s}_{opt}$ is greater than $1 - \epsilon_s$, with the numerical tolerance $\epsilon_s = 10^{-6}$. Finally, for the backup solution, the conservative kinematic constraints to generate the backup solution are chosen to be 40% of the kinematic constraints defined in Equation (33). The cutoff frequency for the filters used for the fitting of backup solution is set to 10 Hz, with the aim of mitigating any oscillations within the 0.1 s update window.

4.3. Experimental Results

The experimental results of the FO via pass-to-pass learning and that via level-to-level learning are respectively presented in Figures 10 and 11 for the airfoil shape, in Figures 12 and 13 for the tensile test specimen, and in Figures 14 and 15 for the spur gear.

As shown in Figures 10, 12, and 14, after initialization, the pass-to-pass learning model can predict the contour error and spindle torque reasonably accurately (albeit with large credible intervals) starting from the second pass of Z-3.0 level. However, one can notice that the spindle torque prediction is usually slightly higher than the measurements in the first few iterations. This inaccuracy and the resultant loss of model confidence lead to frequent switching to the backup solution. In addition, feedrate oscillations are observed, potentially because the SLP cannot converge within the iteration limit. Therefore, SLP is terminated by reaching the maximum number of iterations, yielding sub-optimal results. However, these imperfections are quickly mitigated after a few more iterations of machining, data acquisition, and training. At the final z-level (Z-4.5), the obtained feedrates are close to optimal. Notice that the optimal feedrate for the roughing passes slightly decreases along the straight line or concave arc segments due to a larger tool engagement angle, while the optimal feedrate for the finishing passes also drops during the high-curvature portions and at motion reversal points due to the larger servo error and quadrant glitches.

 doi:10.20944/preprints202603.2368.v1

On the other hand, for level-to-level learning, as shown in Figures 11, 13, and 15, after initialization using two z-levels, the contour error and spindle torque predictions at the final z-level (Z-4.5) still exhibit very large credible intervals, especially for contour error. This lack of confidence does not significantly affect FO for the roughing passes, given the looser contour error constraints. However, it significantly affects the finishing passes, where the optimized feedrate is oscillatory. Moreover, the predicted spindle torque is typically underestimated for the roughing passes at Z-4.5 level, which also implies that, with fewer training data per model, the learning model may overfit the training data and cannot extrapolate well.

Although pass-to-pass learning demonstrates superior performance compared to level-to-level learning in terms of learning speed and the rate of reaching optimality, its main deficiency is that it may still be affected by heterogeneity in cutting geometries across different passes. For the case studies of machining the tensile test specimen or the spur gear, as shown in Figures 12, 13, 14, 15, the optimized feedrate is affected by the geometric differences across passes. For example, in roughing passes, feedrate drops occur more often in pass-to-pass learning than in level-to-level learning. This is likely due to the fact that the spindle torque ripples or spikes learned in the finishing passes are carried over into the prediction of the roughing passes. The results of the spur gear case study, i.e., Figures 14 and 15, further show frequent switching to the backup solution at certain locations during the finishing passes. These locations correspond to the roots of the gear teeth, where the engagement angle increases sharply and cannot be captured solely by the radial depth of cut. Both deficiencies suggest that the learning model could be improved to mitigate these effects.

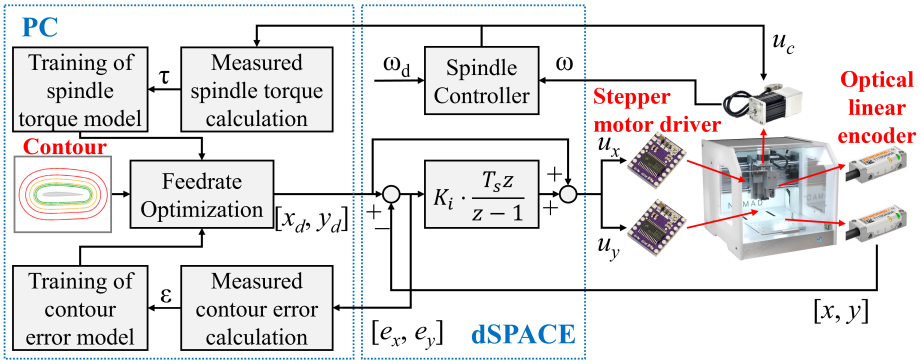

Figure 8. The experimental setup used for the case studies. It consists of two learning and one FO modules, performed on a PC, and two motor control loops for servo and spindle, performed on dSPACE.

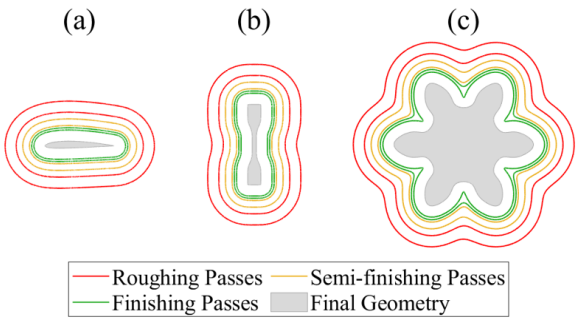

Figure 9. The part geometries and the cutting passes used in the experimental case studies for the machining of (a) an airfoil shape, (b) a tensile test specimen, and (c) a spur gear.

 doi:10.20944/preprints202603.2368.v1

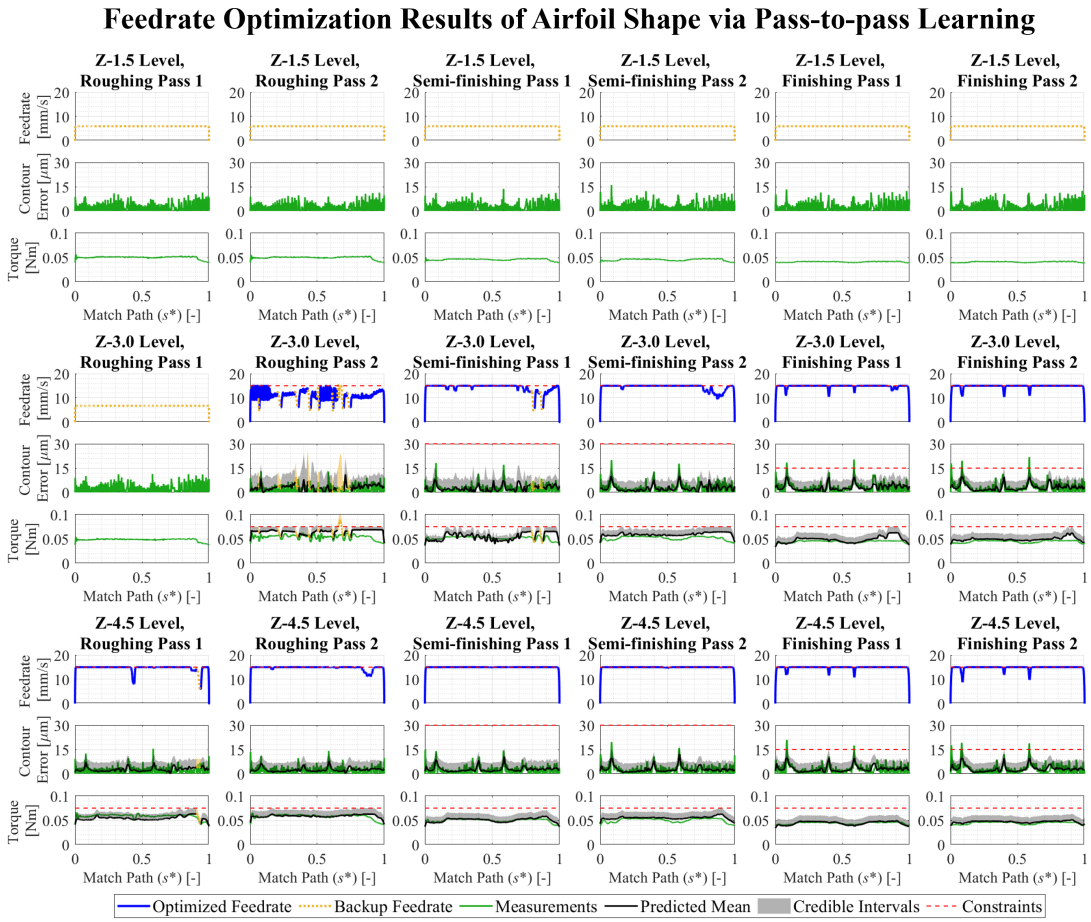

Figure 10. The experimental results of machining airfoil shape based on feedrate optimization via pass-to-pass learning. After initialization, the FO via pass-to-pass learning can return close-to-optimal feedrates after only a few passes (Note: the contour error constraints for the roughing passes is 45μm, which is above the plot's axis limit.)

 doi:10.20944/preprints202603.2368.v1

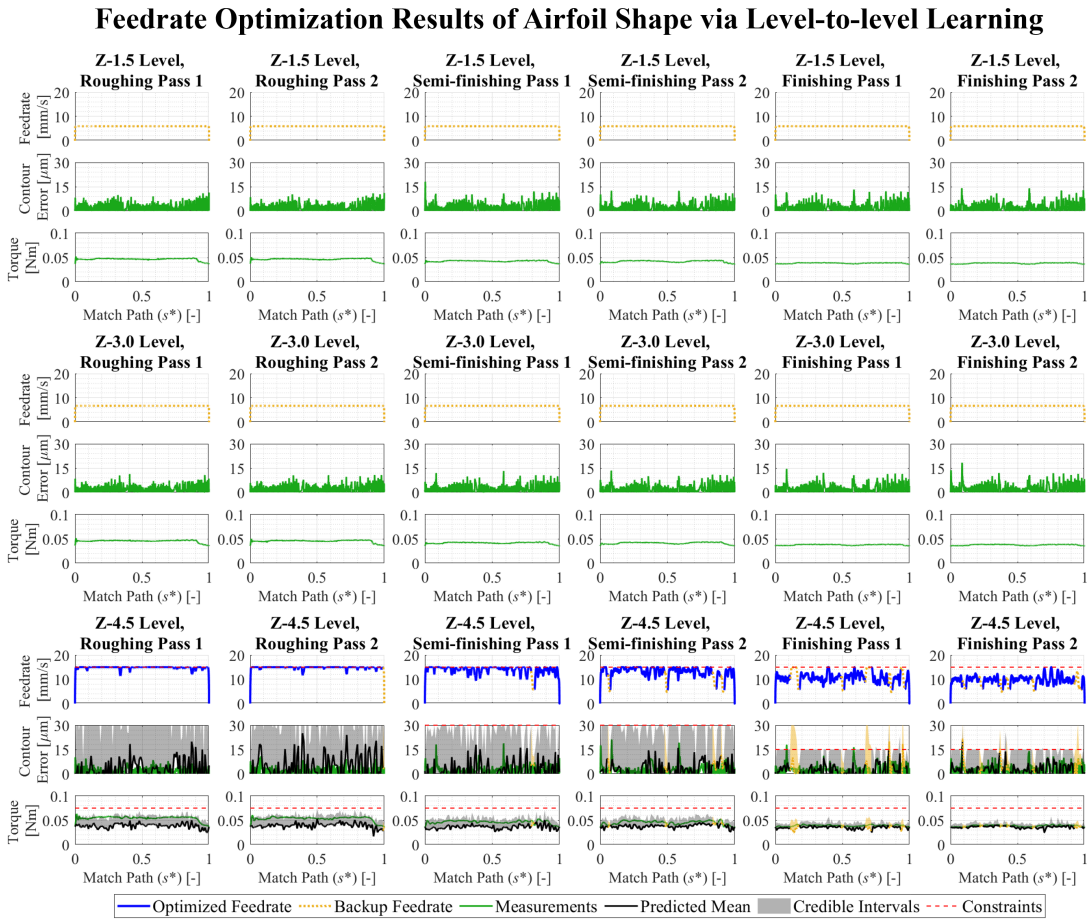

Figure 11. The experimental results of machining airfoil shape based on feedrate optimization via level-to-level learning. After initialization with two z-levels, level-to-level learning still cannot achieve sufficient confidence level and return optimal feedrates. (Note: the contour error constraints for the roughing passes is 45μm, which is above the plot's axis limit.)

 doi:10.20944/preprints202603.2368.v1

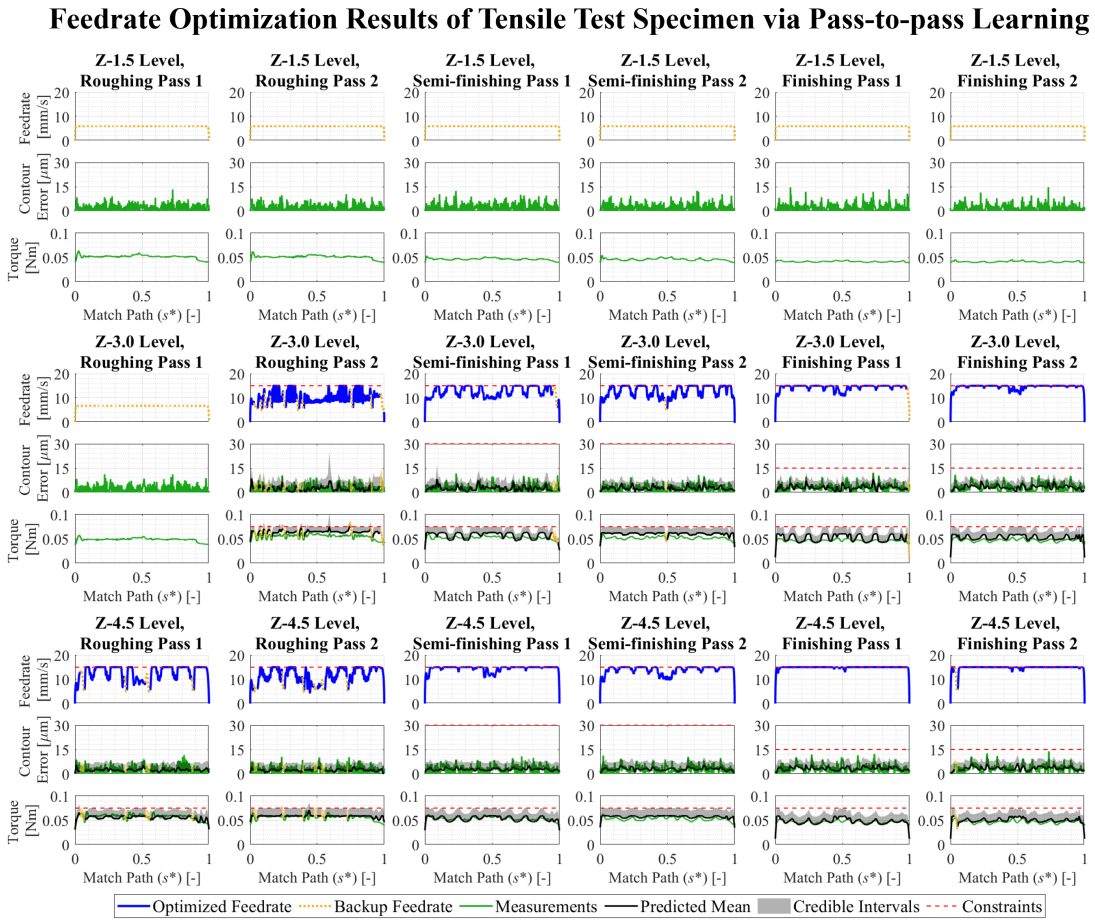

Figure 12. The experimental results of machining tensile test specimen based on feedrate optimization via pass-to-pass learning. This case study shows that the pass-to-pass learning can be affected by the geometric differences between non-identical passes. (Note: the contour error constraints for the roughing passes is 45μm, which is above the plot's axis limit.)

 doi:10.20944/preprints202603.2368.v1

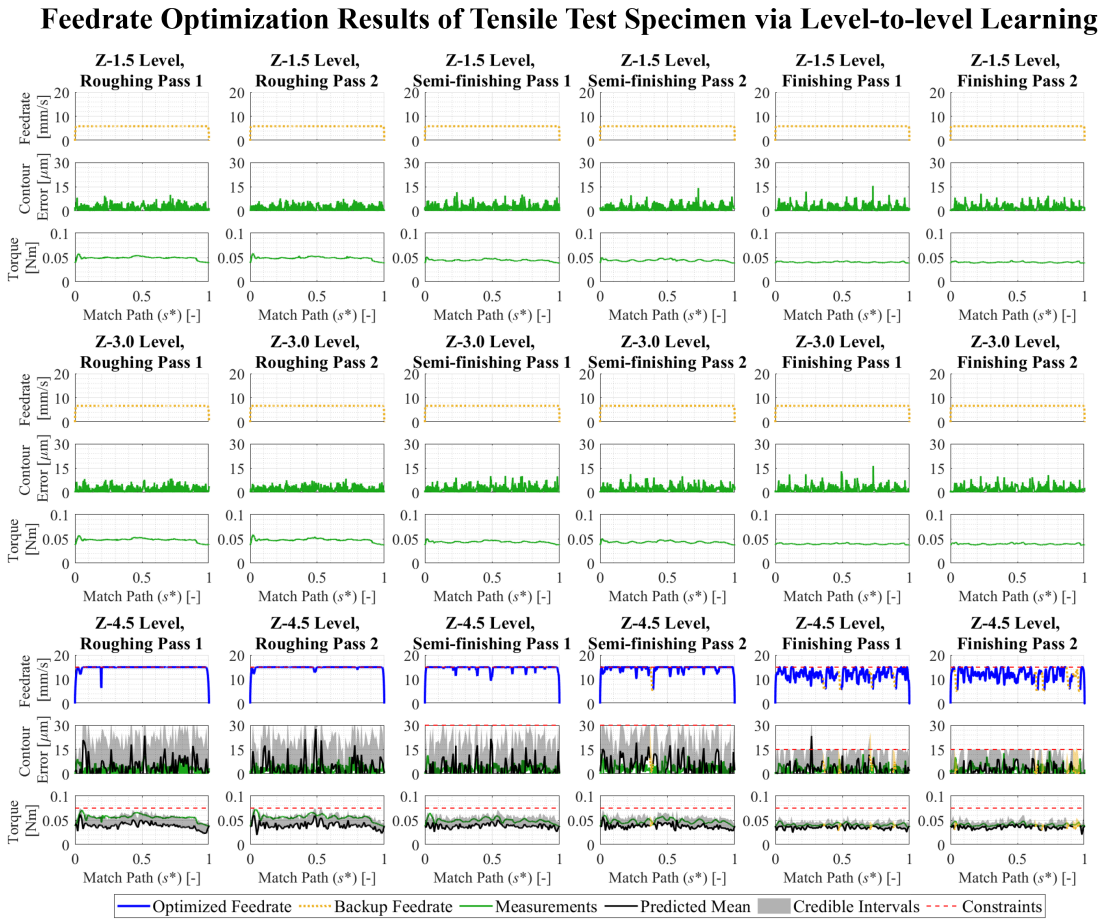

Figure 13. The experimental results of machining tensile test specimen based on feedrate optimization via level-to-level learning. It shows that level-to-level learning does not result in frequent drops in roughing passes due to geometric differences across passes. (Note: the contour error constraints for the roughing passes is 45μm, which is above the plot's axis limit.)

 doi:10.20944/preprints202603.2368.v1

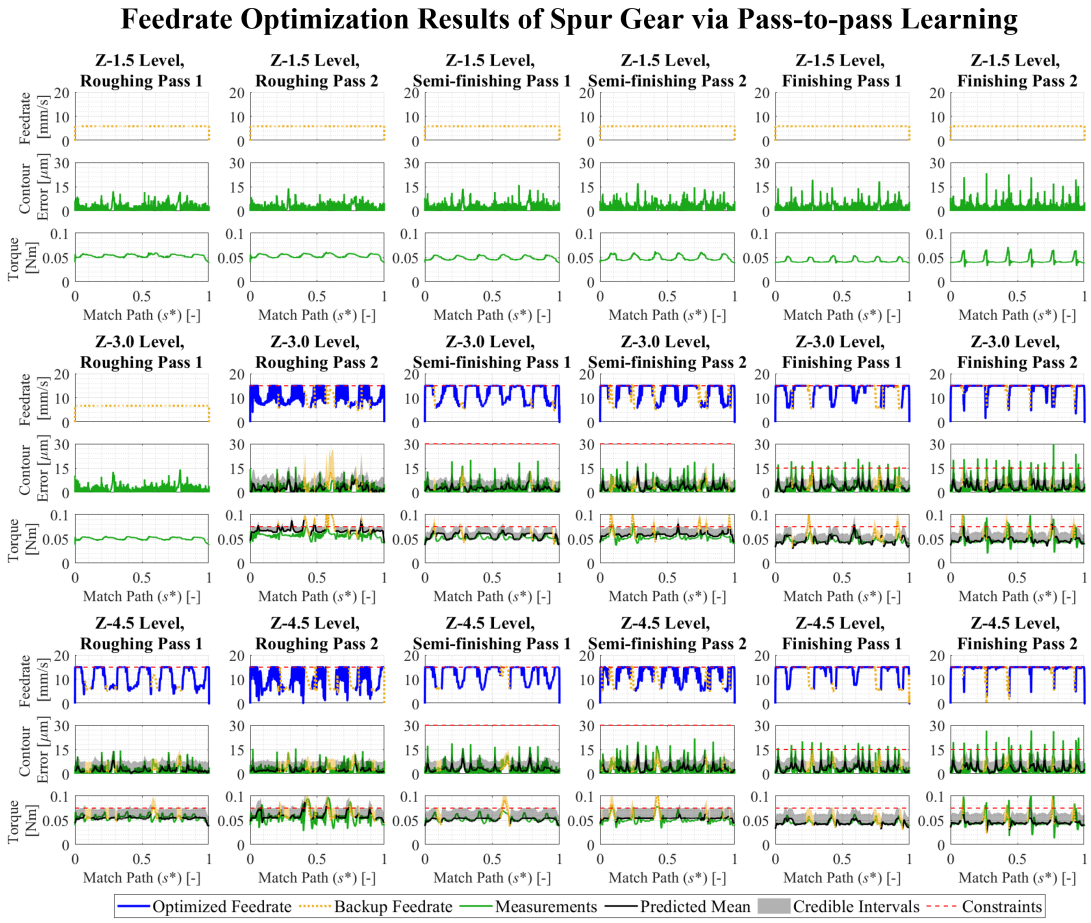

Figure 14. The experimental results of machining spur gear based on feedrate optimization via pass-to-pass learning. It clearly shows that the spindle torque predictions for roughing passes are affected by the training of finishing passes in pass-to-pass learning. (Note: the contour error constraints for the roughing passes is 45μm, which is above the plot's axis limit.)

 doi:10.20944/preprints202603.2368.v1

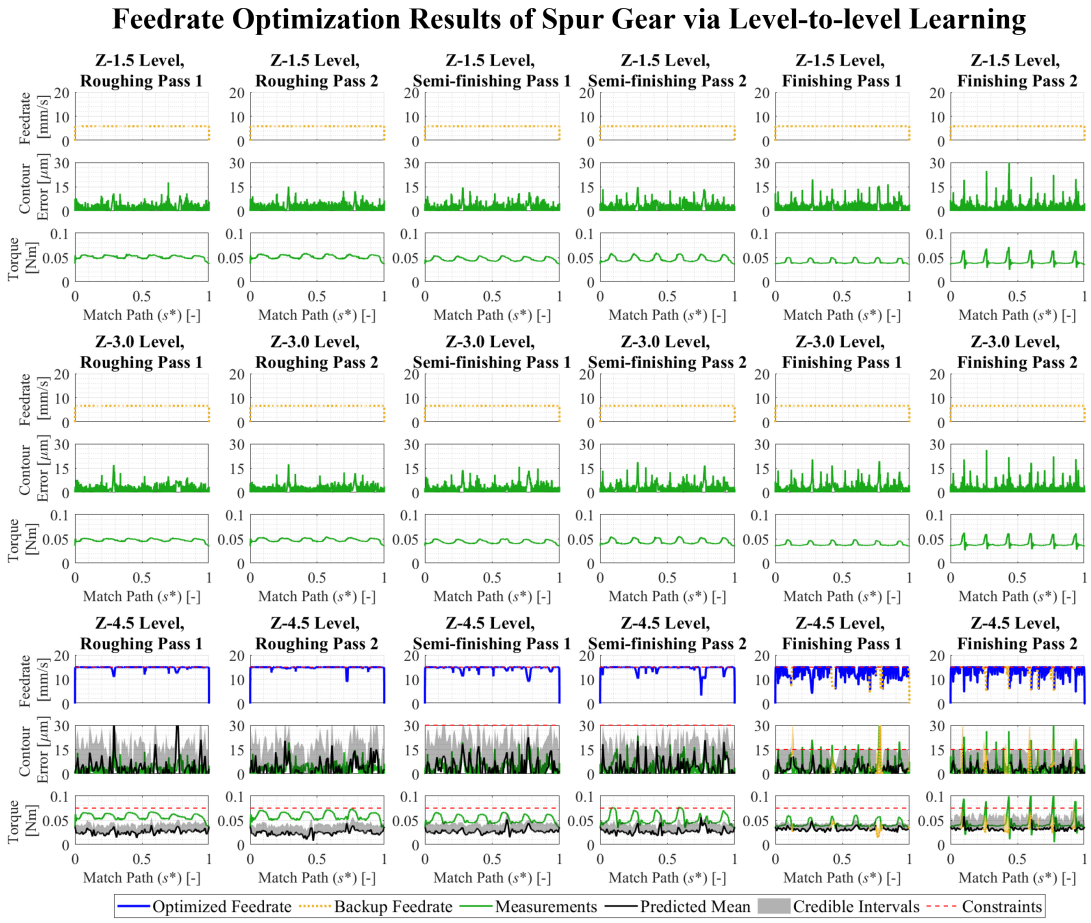

Figure 15. The experimental results of machining spur gear based on feedrate optimization via level-to-level learning. The frequent switching to backup solutions during the finishing passes suggests that the learning model could be improved. (Note: the contour error constraints for the roughing passes is 45μm, which is above the plot's axis limit.)

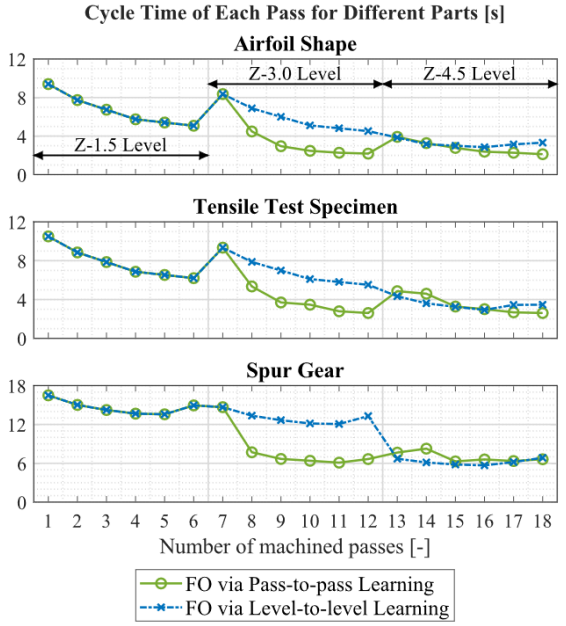

Figure 16. Individual cycle times of each pass using FO via pass-to-pass learning or via level-to-level learning. It is shown that FO via pass-to-pass learning can significantly reduce the cycle time starting Z-3.0 level, whereas that via level-to-level learning requires more iterations of training and does not reduce cycle time until Z-4.5 level.

 doi:10.20944/preprints202603.2368.v1

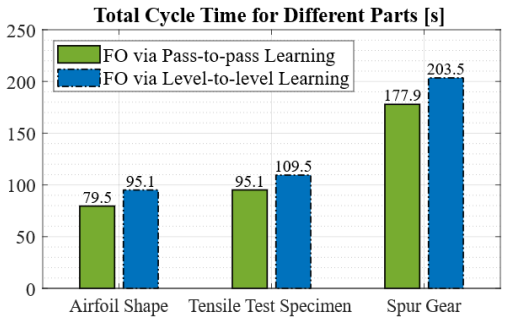

Figure 17. Total cycle time to machine the parts using FO via pass-to-pass learning or via level-to-level learning.

In addition, among all case studies, one can also find that the peaks of the contour error are generally underestimated. This is likely due to the sampling procedure in Equation (3) and the augmentation of the training samples introduced in Equation (12). Regarding the sampling problem, the peaks of the error may not always be selected as sample points and therefore may be excluded from training. Even when error peaks are selected, the augmented training samples from the neighboring locations, which have lower values, can also weaken the model's ability to learn those peaks, despite increasing the number of training samples. This illustrates a trade-off between learning accuracy and training speed.

Overall, the cycle times of machining each pass for different part geometries are summarized in Figure 16, while the total times to machine the whole parts are summarized in Figure 17. The machining time results show that FO via pass-to-pass learning can yield faster machining compared to the level-to-level learning, with a 16.4%, 13.1%, and 12.6% improvement of the total machining time to machine an airfoil shape, a tensile test specimen, and a spur gear respectively, thanks to its flexibility of training on non-identical passes. Note that the proposed framework, which formulates feedrate optimization as an SLP problem, requires substantially more computation time to generate the optimized feedrate profile compared to the resulting cycle time savings, leading to overall longer total time (cycle time + computational time) for FO via pass-to-pass or level-to-level learning than using conservative feedrates across all passes. This deficiency, however, may be alleviated with greater computational resources.

5. Conclusions

This paper proposes a feedrate optimization framework using pass-to-pass learning for contour machining tasks. The pass-to-pass learning is enabled by two Bayesian linear regression models and a contour matching technique, while the FO framework is based on window-based approach and sequential linear programming. The proposed framework works as follows: The contour matching algorithm first identifies sections of each pass that have similar geometry features, then two Bayesian linear regression models respectively learn the contour error and the spindle torque based on the data collected from previous passes, and finally the feedrate optimizer iteratively updates the feedrate based on the prediction from the pass-to-pass learning model.

Experiments of 2.5D contour machining for three different prismatic parts are performed on a 3-axis desktop CNC milling machine. The results show that the proposed FO via pass-to-pass learning can return close-to-optimal feedrates shortly after initialization, culminating in up to 16.4% improvement in productivity relative to an equivalent learning-based FO algorithm for identical toolpaths [31]. However, the interference from geometric differences across passes can increase the prediction uncertainty (i.e., larger credible intervals) of the pass-to-pass learning model and thereby compromise the FO. In addition, the computational time of the FO also adversely affects the total time required for the machining process. Therefore, future work will include improving the efficiency of the FO framework and developing more complex learning model structures to better handle the geometric differences. Other future directions include accounting for tool wear and changes in machine behavior during operations, incorporating physics-based models and feedforward controller to further boost

 doi:10.20944/preprints202603.2368.v1

feedrate, and extending pass-to-pass learning to machine-to-machine learning, where learning occurs across a fleet of similar CNC machine tools.

Author Contributions: Cheng-Hao Chou – Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. Milad Azvar – Conceptualization, Investigation, Methodology, Software, Writing – review & editing. Chenhui Shao – Conceptualization, Methodology, Writing – review & editing. Chinedum Okwudire – Conceptualization, Funding acquisition, Methodology, Project administration, Resources, Supervision, Writing – original draft, Writing – review & editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the National Science Foundation grant #2054715.

Conflicts of Interest: The authors have no known financial or non-financial interests to disclose.

References

1. Bristow, D.; Tharayil, M.; Alleyne, A. A survey of iterative learning control. *IEEE Control Systems Magazine* **2006**, *26*, 96–114. https://doi.org/10.1109/MCS.2006.1636313.
2. Lo, C.C.; Hsiao, C.Y. CNC machine tool interpolator with path compensation for repeated contour machining. *Computer-Aided Design* **1998**, *30*, 55–62. https://doi.org/10.1016/S0010-4485(97)00053-5.
3. Barton, K.L.; Alleyne, A.G. A Cross-Coupled Iterative Learning Control Design for Precision Motion Control. *IEEE Transactions on Control Systems Technology* **2008**, *16*, 1218–1231. https://doi.org/10.1109/TCST.2008.919433.
4. Hendrawan, Y.M.; Simba, K.R.; Uchiyama, N. Iterative learning based trajectory generation for machine tool feed drive systems. *Robotics and Computer-Integrated Manufacturing* **2018**, *51*, 230–237. https://doi.org/10.1016/j.rcim.2017.12.009.
5. Hendrawan, Y.M.; Farrage, A.; Uchiyama, N. Iterative NC program modification and energy saving for a CNC machine tool feed drive system with linear motors. *The International Journal of Advanced Manufacturing Technology* **2019**, *102*, 3543–3562. https://doi.org/10.1007/s00170-019-03390-1.
6. Chen, S.L.; Hsieh, S.M. Iterative learning contouring control: Theory and application to biaxial systems. *Mechatronics* **2023**, *89*, 102932. https://doi.org/10.1016/j.mechatronics.2022.102932.
7. Bahtiyar, K.; Sencer, B.; Beudaert, X. Data-driven feedforward control of inertial dampers for accuracy improvement. *CIRP Annals* **2024**, *73*, 317–320. https://doi.org/10.1016/j.cirp.2024.04.007.
8. Bolder, J.; Oomen, T. Rational Basis Functions in Iterative Learning Control—With Experimental Verification on a Motion System. *IEEE Transactions on Control Systems Technology* **2015**, *23*, 722–729. https://doi.org/10.1109/TCST.2014.2327578.
9. Tsurumoto, K.; Ohnishi, W.; Koseki, T.; van Haren, M.; Oomen, T. Integrated Rational Feedforward in Frequency-Domain Iterative Learning Control for Highly Task-Flexible Motion Control. *IEEE/ASME Transactions on Mechatronics* **2024**, *29*, 3010–3018. https://doi.org/10.1109/TMECH.2024.3400252.
10. Dumanli, A.; Sencer, B. Pre-compensation of servo tracking errors through data-based reference trajectory modification. *CIRP Annals* **2019**, *68*, 397–400. https://doi.org/10.1016/j.cirp.2019.03.017.
11. Altintas, Y.; Erkorkmaz, K. Feedrate Optimization for Spline Interpolation In High Speed Machine Tools. *CIRP Annals* **2003**, *52*, 297–302. https://doi.org/10.1016/S0007-8506(07)60588-5.
12. Dong, J.; Ferreira, P.; Stori, J. Feed-rate optimization with jerk constraints for generating minimum-time trajectories. *International Journal of Machine Tools and Manufacture* **2007**, *47*, 1941–1955. https://doi.org/10.1016/j.ijmachtools.2007.03.006.
13. Erkorkmaz, K.; Chen, Q.G.C.; Zhao, M.Y.; Beudaert, X.; Gao, X.S. Linear programming and windowing based feedrate optimization for spline toolpaths. *CIRP Annals* **2017**, *66*, 393–396. https://doi.org/10.1016/j.cirp.2017.04.058.
14. Bharathi, A.; Dong, J. Feedrate optimization for smooth minimum-time trajectory generation with higher order constraints. *The International Journal of Advanced Manufacturing Technology* **2016**, *82*, 1029–1040. https://doi.org/10.1007/s00170-015-7447-x.
15. Bosetti, P.; Bertolazzi, E. Feed-rate and trajectory optimization for CNC machine tools. *Robotics and Computer-Integrated Manufacturing* **2014**, *30*, 667–677. https://doi.org/10.1016/j.rcim.2014.03.009.
16. Lu, T.C.; Chen, S.L. Novel Feedrate Optimization Method for NURBS Tool Paths Under Various Constraints. *IEEE Access* **2022**, *10*, 3192–3205. https://doi.org/10.1109/ACCESS.2021.3138049.

 doi:10.20944/preprints202603.2368.v1

17. Fan, W.; Gao, X.S.; Lee, C.H.; Zhang, K.; Zhang, Q. Time-optimal interpolation for five-axis CNC machining along parametric tool path based on linear programming. *The International Journal of Advanced Manufacturing Technology* **2013**, *69*, 1373–1388. https://doi.org/10.1007/s00170-013-5083-x.
18. Sencer, B.; Dumanli, A.; Yamada, Y. Spline interpolation with optimal frequency spectrum for vibration avoidance. *CIRP Annals* **2018**, *67*, 377–380. https://doi.org/10.1016/j.cirp.2018.03.002.
19. Chen, M.; Xu, J.; Sun, Y. Adaptive feedrate planning for continuous parametric tool path with confined contour error and axis jerks. *The International Journal of Advanced Manufacturing Technology* **2017**, *89*, 1113–1125. https://doi.org/10.1007/s00170-016-9021-6.
20. Dong, J.; Stori, J.A. Optimal Feed-Rate Scheduling for High-Speed Contouring. *Journal of Manufacturing Science and Engineering* **2007**, *129*, 63–76. https://doi.org/10.1115/1.2280549.
21. DiCola, K.N.; Chen, C.Q.G.; Engin, S.; Erkorkmaz, K. Two-stage LP/NLP feedrate optimization for spline toolpaths. *CIRP Journal of Manufacturing Science and Technology* **2025**, *60*, 122–137. https://doi.org/10.1016/j.cirpj.2025.04.005.
22. Balula, S.; Liao-McPherson, D.; Rupenyan, A.; Lygeros, J. Data-driven reference trajectory optimization for precision motion systems. *Control Engineering Practice* **2024**, *144*, 105834. https://doi.org/10.1016/j.conengprac.2023.105834.
23. Kim, H.; Okwudire, C.E. Simultaneous servo error pre-compensation and feedrate optimization with tolerance constraints using linear programming. *The International Journal of Advanced Manufacturing Technology* **2020**, *109*, 809–821. https://doi.org/10.1007/s00170-020-05651-w.
24. Oh, J.; Sim, B.; Lee, W.; Choi, S.; Lee, W. Model-based feed rate optimization for cycle time reduction in milling. *Journal of Manufacturing Processes* **2023**, *94*, 289–296. https://doi.org/10.1016/j.jmapro.2023.03.033.
25. Erkorkmaz, K.; Layegh, S.E.; Lazoglu, I.; Erdim, H. Feedrate optimization for freeform milling considering constraints from the feed drive system and process mechanics. *CIRP Annals* **2013**, *62*, 395–398. https://doi.org/10.1016/j.cirp.2013.03.084.
26. Mansour, S.Z.; Seethaler, R. Feedrate Optimization for Computer Numerically Controlled Machine Tools Using Modeled and Measured Process Constraints. *Journal of Manufacturing Science and Engineering* **2016**, *139*, 011012. https://doi.org/10.1115/1.4033933.
27. Chang, Y.C.; Chen, C.W.; Tsao, T.C. Near Time-Optimal Real-Time Path Following Under Error Tolerance and System Constraints. *Journal of Dynamic Systems, Measurement, and Control* **2018**, *140*, 071004. https://doi.org/10.1115/1.4038651.
28. Kim, H.; Okwudire, C.E. Intelligent feedrate optimization using a physics-based and data-driven digital twin. *CIRP Annals* **2023**, *72*, 325–328. https://doi.org/10.1016/j.cirp.2023.04.063.
29. Kim, H.; Kontar, R.A.; Okwudire, C.E. Intelligent Feedrate Optimization Using an Uncertainty-Aware Digital Twin Within a Model Predictive Control Framework. *IEEE Access* **2024**, *12*, 49947–49961. https://doi.org/10.1109/ACCESS.2024.3384471.
30. Rattunde, L.; Laptev, I.; Klenske, E.D.; Möhring, H.C. Safe optimization for feedrate scheduling of power-constrained milling processes by using Gaussian processes. *Procedia CIRP* **2021**, *99*, 127–132. https://doi.org/10.1016/j.procir.2021.03.020.
31. Chou, C.H.; Shao, C.; Okwudire, C.E. Feedrate optimization based on part-to-part learning in repeated machining. *CIRP Annals* **2025**, *74*, 569–573. https://doi.org/10.1016/j.cirp.2025.04.043.
32. Murphy, K.P. *Machine Learning : A Probabilistic Perspective*; MIT press: Cambridge, UNITED STATES, 2012.
33. Kim, H.; Okwudire, C.E. Accurate and computationally efficient approach for simultaneous feedrate optimization and servo error pre-compensation of long toolpaths—with application to a 3D printer. *The International Journal of Advanced Manufacturing Technology* **2021**, *115*, 2069–2082. https://doi.org/10.1007/s00170-021-07200-5.