Mech. Sci., 17, 747–758, 2026
https://doi.org/10.5194/ms-17-747-2026
© Author(s) 2026. This work is distributed under
the Creative Commons Attribution 4.0 License.

Mechanical Sciences
Open Access

An enhanced neural network-based method for predicting step error in a five-axis finishing machining

Te Ye, Wei Liu, Jiaping Zhang, Jiawei Chen, Jingyang Yu, Yuhang Zhao, and Ziyu Zhang

School of Mechanical Engineering, Suzhou University of Science and Technology, Suzhou 215000, China

Correspondence: Wei Liu (liuwei@usts.edu.cn)

Received: 29 March 2026 – Revised: 19 June 2026 – Accepted: 4 July 2026 – Published: 20 July 2026

Abstract. To address the low efficiency of traditional geometry-based step error calculations in five-axis numerical control (NC) finishing that cannot reuse historical data, this study proposes a step error prediction method based on a SSA-Attention-LSTM (Sparrow search algorithm–attention mechanism–long short-term memory) model. The method employs principal component analysis (PCA) for dimensionality reduction, optimizes LSTM hyperparameters using the Sparrow search algorithm, incorporates an attention mechanism to enhance key feature extraction, and introduces dropout to prevent overfitting. Experimental results on two surface models show that the proposed model achieves an average R^2 of 99.4 %, which is 31 %, 10 %, 5 %, and 6 % higher than that of back propagation (BP), long short-term memory (LSTM), bidirectional gated recurrent unit (BiGRU), and bidirectional long short-term memory (BiLSTM) models. In the blade test case, the model achieves high-precision prediction using only 9480 training samples, with an R^2 of 0.995, demonstrating high prediction accuracy for step error in five-axis finishing tool paths.

1 Introduction

Five-axis computer numerical control (CNC) machining is widely applied in the precision machining of complex free-form surfaces. As shown in Fig. 1a, the cutting path of the tool along the discrete cutter contact (CC) points on the workpiece surface is illustrated, while Fig. 1b shows the step error formed between adjacent cutter location (CL) points. Taking the jth and $(j+1)$th CL points on the ith tool path as an example, the tool holding points are $O_{i,j}$ and $O_{i,j+1}$, respectively. The maximum normal distance $e^{\mathrm{L}}_{\max}$ between the CC point line $P^{\mathrm{CC}}_{i,j}P^{\mathrm{CC}}_{i,j+1}$ and the ideal CC point trajectory $\mathrm{CC}_{i,j}$ is defined as the linear error. Taking the midpoint O_{m} of the line segment $O_{i,j}$ and $O_{i,j+1}$, the maximum normal distance $e^{\mathrm{NL}}_{\max}$ among the tool base circle at O_{m}, the CC point line $P^{\mathrm{CC}}_{i,j}P^{\mathrm{CC}}_{i,j+1}$, and the ideal CC point trajectory, $\mathrm{CC}_{i,j}$ is defined as the nonlinear error, where $P^{\mathrm{CL}}_{\mathrm{m}}$ is the CC point coordinate corresponding to the midpoint of the CL points, while $\boldsymbol{T}_{\mathrm{m}}$ represents the intermediate tool axis vector obtained by averaging and normalizing $\boldsymbol{T}_{i,j}$ and $\boldsymbol{T}_{i,j+1}$.

When the tool completes the five-axis simultaneous motion from $P^{\mathrm{CL}}_{i,j}$ to $P^{\mathrm{CL}}_{i,j+1}$, the tool envelope surface formed by the sweeping cutting edge (i.e., the edge of the tool base circle) constitutes the actual cutting path. The maximum normal distance $e_{\max}$ between this actual cutting path and the ideal CC point trajectory line $\mathrm{CC}_{i,j}$ is defined as the step error, which is a key geometric indicator for controlling machining accuracy.

Currently, step error calculation mainly relies on geometry-based iterative methods. Various approaches have been proposed, such as point-by-point iteration by Liu et al. (2022), dynamic-programming-based tool path planning by Wu et al. (2008), and geometric envelope methods by Zhou et al. (2015). These geometry-based iterative methods require explicit mathematical models, and their computational cost increases sharply with higher-precision requirements. Moreover, each calculation is performed independently without reusing historical experience, resulting in low efficiency when generating high-quality tool paths that require massive step error calculations, which has become a major bottleneck limiting their practical application.

In recent years, machine learning techniques have shown potential in tool path planning due to their ability to learn from data and handle complex nonlinear relationships. Neu-

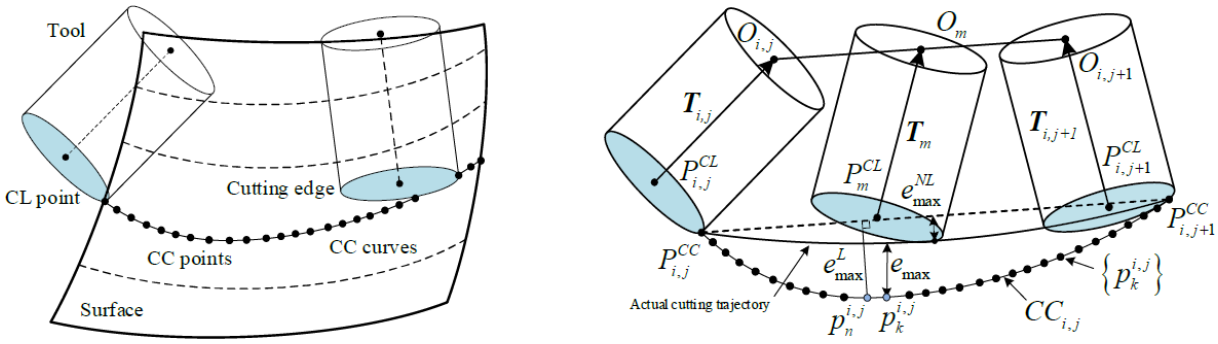


Figure 1. Tool cutting and step error. **(a)** Tool cutting along CC points. **(b)** Step error between two adjacent CL points.

ral networks have been applied to personalized tool path generation, machining error prediction, tool path length prediction, and machining cycle prediction (Eskandari et al., 2013; Geng et al., 2011; Sun et al., 2022). Zhang et al. (2023) proposed using a back propagation (BP) neural network to predict three-axis step error with high accuracy. However, it suffers from issues such as slow computation and susceptibility to overfitting, making it difficult to directly apply to five-axis step error prediction.

Besides the machining field, data-driven methods are also widely applied in dynamic system analysis and intelligent monitoring. Existing studies have established 3D vision systems to realize non-contact detection of equipment (Cascino et al., 2025), and other studies have utilized deep learning to complete mechanical component defect recognition (Zheng et al., 2021). The above cross-domain studies confirm that deep neural networks excel at mining nonlinear features from high-dimensional data, providing a good reference for the step error prediction research in this paper.

To address the above problems, this paper proposes a step error prediction method for five-axis NC finishing based on the SSA-attention-LSTM (Sparrow search algorithm–attention mechanism–long short-term memory) model. By integrating the Sparrow search algorithm (SSA) for hyperparameter optimization and incorporating an attention mechanism to enhance key feature extraction, the model effectively captures the spatiotemporal continuity and nonlinear characteristics of tool motion, thereby improving prediction accuracy and stability while preventing overfitting. In order to use the SSA-attention-LSTM model to predict five-axis finishing step error, this paper conducts research on data preprocessing architecture design, network structure optimization, and multi-task collaborative training mechanisms. The main content is as follows: Sect. 2 defines the step error based on five-axis machining geometric principles. Section 3 designs the sample data architecture. Z-score standardization is applied to eliminate unit differences (Al-Faiz et al., 2018). Principal component analysis (PCA) is used for dimensionality reduction and reconstruction of high-dimensional discrete point sets to compress redundant noise and enhance model generalization capability (Fariz and Basha, 2024). Section 4 constructs the SSA-attention-LSTM collaborative optimization framework. The SSA is introduced to dynamically optimize LSTM hyperparameters (hidden-layer dimensions, learning rate). The attention mechanism is used to weight and focus on key CC and CL features. Dropout (Chen et al., 2024) regularization and the Adam (Kingma and Ba, 2014; Salem et al., 2022) optimizer are combined to suppress overfitting. Section 5 validates the effectiveness and prediction accuracy of the proposed SSA-attention-LSTM model using tool path sample data from two general free-form surfaces and a typical aero-engine blade workpiece, covering both basic geometric test cases and practical industrial scenarios.

2 Sample data design and optimization

This paper employs tool parameters and CC and CL points as core input variables. As the samples contain diverse parameter types (e.g., tool radius, point coordinates), standardization is applied to unify the scales of these parameters. The high-dimensional features of the standardized samples are reconstructed by PCA to reduce the dimensionality, aiming to eliminate interference from redundant features and retain critical ones, thereby suppressing model overfitting.

2.1 Sample data design

Before using the neural network model for step error prediction, extensive samples are required to train the model for ideal prediction accuracy. As derived from the geometric principles of step error, the following parameters are all necessary input variables for calculating five-axis step error: adjacent CC points ($P_{i,j}^{\mathrm{CC}}$, $P_{i,j+1}^{\mathrm{CC}}$), CL points ($P_{i,j}^{\mathrm{CL}}$, $P_{i,j+1}^{\mathrm{CL}}$) and their corresponding tool axis vectors ($\boldsymbol{T}_{i,j}$, $\boldsymbol{T}_{i,j+1}$), the discrete point set $\left\{P_k^{i,j}\right\}$ on the local CC curve$\mathrm{CC}_{i,j}$, and basic tool parameters. These data collectively constitute the input features of the samples, while the step error value is the only output variable. They form the sample data structure shown in Table 1.

Table 1. Sample data required for step error prediction.

Input layer							Output layer	
Tool radius	Fillet radius	Tool length	Tool inclination angle	Discrete points in CC curve	CL points	Tool vectors	Expected value of step error	Calculated value of step error
R	r	L	α	$\left\{P_k^{i,j}\right\}$, $P_{i,j}^{\mathrm{CC}}$, P_{i+1}^{CC}	P_i^{CL}, P_{i+1}^{CL}	$\boldsymbol{T}_{i,j}$, $\boldsymbol{T}_{i,j+1}$	$\hat{y}_t$	y_t

2.2 Sample data optimization

In step error prediction of the five-axis finish machining, original sample data suffer from high-dimensionality, multi-dimensional units, and redundant features, which directly impair the model's computational efficiency and generalization capability. To address this, this section proposes an optimization strategy based on data standardization and PCA. Z-score standardization is applied to eliminate parameter-scale discrepancies, and PCA extracts critical geometric features and compresses redundant information. This approach effectively reduces data complexity, suppresses noise interference, and significantly enhances the model's prediction efficiency and accuracy.

3 Design and optimization of the SSA-attention-LSTM model for step error prediction

To achieve high-precision and high-efficiency prediction of step error for five-axis finishing tool paths, this paper proposes a joint modeling and optimization method that integrates the SSA with an attention-mechanism-based LSTM network, resulting in the SSA-attention-LSTM model. The attention mechanism is introduced to adaptively highlight key CL point features and suppress attenuation and noise in information transmission. Furthermore, the SSA algorithm is integrated to achieve intelligent global optimization of hyperparameters, enhancing the model's generalizability while avoiding local optima caused by complex sample features, thereby significantly improving the model's prediction robustness and accuracy.

3.1 Design of the LSTM model integrated with attention mechanism

Unlike traditional BP neural networks that process independent samples, LSTM, as an improved architecture of recurrent neural networks (RNNs), introduces gating mechanisms (forget gate, input gate, and output gate). The forget gate determines the retention level of important historical CC and CL points, preserving crucial features. The input gate then integrates subsequent CC and CL point features for comprehensive learning of core sample characteristics. Finally, the output gate extracts the final core sample data from the memory cell. This mechanism incorporates historical sample information for step error prediction, combines current outputs with hidden states from previous time steps, and enable perception and transmission of continuous contextual features.

For five-axis finishing tool paths, this mechanism allows the model to incorporate preceding tool path information and model temporal correlations of step error, thus improving prediction accuracy and stability. However, LSTM may insufficiently distinguish feature importance, and early CL point information is prone to attenuation during transmission, reducing prediction accuracy. Therefore, an attention mechanism is introduced to optimize LSTM by dynamically weighting different features according to their importance, thereby achieving superior prediction results.

As shown in Fig. 2, the attention-LSTM model structure comprises five layers: input layer, LSTM layer, attention layer, fully connected (FC) layer, and output layer. The input layer imports preprocessed sequence data into the LSTM layer for model training. It should be emphasized that the lead angle is used as an independent input feature in this study, and all information of the tilt angle is completely contained in the two three-dimensional tool axis vectors. As a complete representation of spatial posture, the tool axis vector can eliminate the inherent periodic non-uniqueness of angle parameters and retain the inherent coupling relationship between posture angles, which is more conducive to network learning.

After learning sample features through the LSTM layer, the hidden states h_t at all time steps are output. Subsequently, an independent attention layer processes these hidden states; this layer calculates corresponding weight coefficients α_t for each hidden state h_t and continuously updates these weights to emphasize the information most critical for prediction. Finally, nonlinear transformation of the weighted feature vector is performed via a sigmoid activation function. Features are mapped through the FC layer, and prediction is completed at the output layer.

The weight coefficient α_t is calculated in Eq. (1). h_t represents the hidden state at the tth time step. $\mathbf{w}$ is a learnable weight matrix. $\boldsymbol{b}$ and $\boldsymbol{u}$ represent the learnable bias vector and weight vector, respectively. e_t represents the attention score at time step t. α_t indicates the relative importance of the hidden state h_t at the tth time step to the current prediction task,

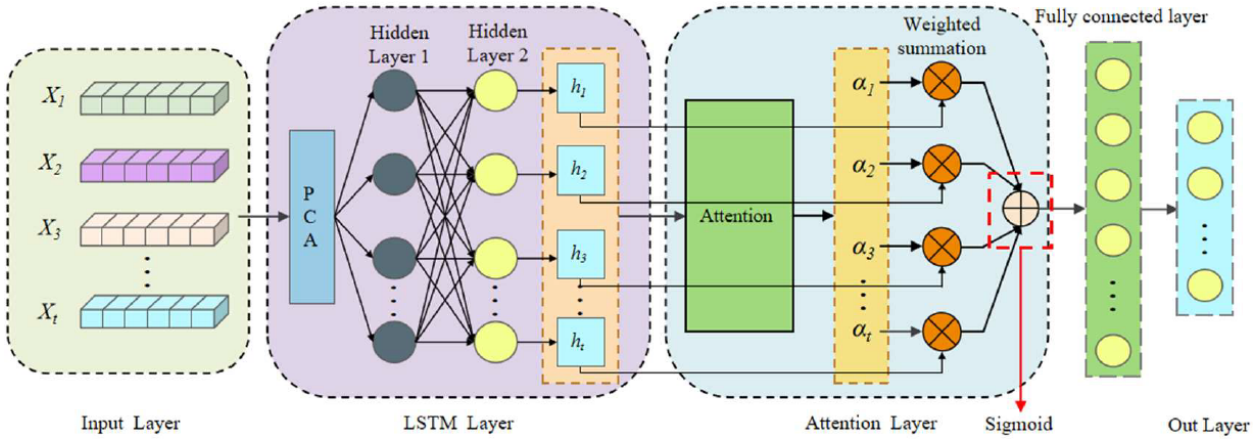


Figure 2. Attention-LSTM structure.

i.e., the weight coefficient.

$$\begin{cases} e_t = \boldsymbol{u}^T \tanh(\mathbf{w}h_t + \boldsymbol{b}) \\ \alpha_t = \frac{\exp(e_t)}{\sum_{j=1}^{T\sum_j} \exp} \end{cases} \tag{1}$$

After calculating the weight coefficients, as shown in Eq. (2), a vector $\boldsymbol{v}$ that integrates global key feature information is obtained by performing a weighted summation of the hidden states across all time steps. This vector encompasses the historical information most critical to the current prediction.

$$\boldsymbol{v} = \sum_{t=1}^{T} \alpha_t h_t \tag{2}$$

Finally, a fully connected layer is set at the output layer. The vector $\boldsymbol{v}$, which integrates global key feature information, is mapped into the final predicted value $\hat{y}$, as shown in Eq. (3). Here, $\mathbf{W}_\mathrm{o}$ represents the output layer weight matrix, $\boldsymbol{b}_\mathrm{o}$ represents the output layer bias vector, and σ represents the Sigmoid activation function.

$$\hat{y} = \sigma\left(\mathbf{W}_\mathrm{o}\boldsymbol{v} + \boldsymbol{b}_\mathrm{o}\right) \tag{3}$$

3.2 Attention-LSTM hyperparameter optimization based on SSA algorithm

In five-axis machining step error prediction tasks, input features within the dataset exhibit strong correlations. Consequently, minor variations in hyperparameters (e.g., LSTM hidden layer dimensions, learning rate) can cause significant fluctuations in prediction accuracy. Furthermore, when machined surface profiles are similar, the distribution of key feature points (CC, CL) may overlap, which can easily trap the training model in local optima. To address this, this section uses the SSA for the collaborative optimization of the LSTM neural network.

The SSA is a swarm intelligence optimization algorithm that simulates the foraging and anti-predation behaviors of sparrows. By mimicking sparrows' food-seeking and danger avoidance behaviors, it balances global and local search, effectively escaping local optima to find the global optimal solution. As shown in Fig. 3, the main steps of the algorithm include setting hyperparameter boundaries, population initialization, producer assignment, scrounger assignment, and scout assignment. After each position update, the fitness value of each individual is calculated. The optimal solution is updated, and the termination condition is assessed based on fitness. The detailed steps are as follows.

3.2.1 Step 1: hyperparameter boundary setting and population initialization

In the initial phase of SSA, the search space is firstly defined, and an initial candidate solution set is generated to establish a start point for subsequent iterative optimization. The population size is N. As shown in Eq. (4), each sparrow's position vector $\boldsymbol{X}_i$ corresponds to a candidate solution, where $\boldsymbol{X}_i = [x_{i1}, x_{i2}, \ldots, x_{iD}]$ represents the position vector of the ith sparrow in the D-dimensional search space. Ub and Lb are the upper- and the lower-bound vectors of the hyperparameters, respectively. The symbol $\odot$ denotes element-wise multiplication, and rand(0, 1) represents a randomly generated vector with a uniform distribution in the interval [0, 1].

$$\boldsymbol{X}_i = \mathrm{Lb} + \mathrm{rand}(0,1) \odot (\mathrm{Ub} - \mathrm{Lb}), \quad i = 1, 2, \ldots, N \tag{4}$$

3.2.2 Step 2: fitness value calculation and producer assignment

1. *Fitness value calculation.*

   The fitness values of the initially selected population are calculated separately. The individual fitness value $f(\boldsymbol{X}_i)$ for $\boldsymbol{X}_i$ is calculated by Eq. (5), where M is the number of validation set samples, y_j is the true value of the jth sample, and $(\boldsymbol{X}_i)$ is the predicted value of the jth

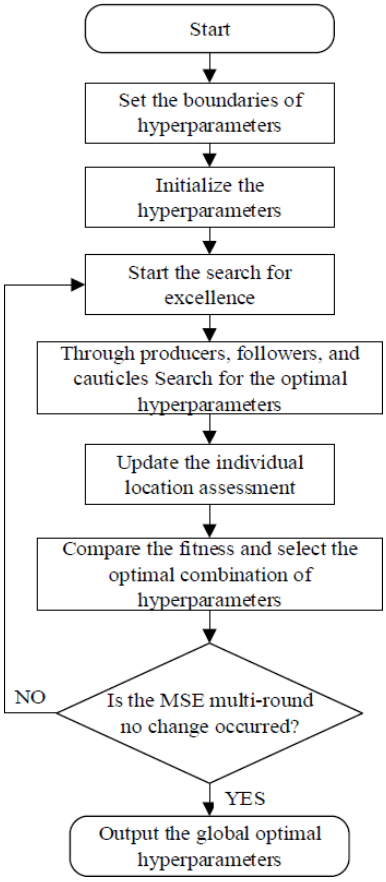


Figure 3. SSA framework process.

sample by the model using hyperparameters $\boldsymbol{X}_i$.

$$f(\boldsymbol{X}_i) = \frac{1}{M}\sum_{j=1}^{M}\left(y_j - \hat{y}_j(\boldsymbol{X}_i)\right)^2 \tag{5}$$

2. *Producer selection.*

   To guide the population to search in better regions, leaders with exploration capability, named "producers", are selected from the current population. The calculated fitness values are sorted in ascending order (since the fitness value is MSE, a smaller value indicates a better solution). As shown in Eq. (6), the top p_n individuals are selected as producers P to guide the optimal hyperparameter search direction, where rank() sorts $f(\boldsymbol{X}_i)$ in ascending order.

$$P = \{\boldsymbol{X}_i \,|\mathrm{rank}(f(\boldsymbol{X}_i)) \le p_n \tag{6}$$

3. *Producer position update.*

   When producers perform global search, both extensive exploration and refined local search are expected. In Eq. (7), $\boldsymbol{X}_i^{t+1}$ reflects the real-time update of the producer's position. t is the current iteration count. T is the maximum number of iterations. α represents a random decay factor. R_2 is the alert value. If R_2 is less than the safety threshold ST, refined local search is performed. Otherwise, extensive global search is conducted. $\boldsymbol{Q} \sim N(0,1)$ is a random vector following the standard normal distribution, and $\boldsymbol{L}$ is an all-one vector.

$$\boldsymbol{X}_i^{t+1} = \begin{cases} \boldsymbol{X}_i^t \odot \exp\left(-\frac{i}{\alpha \cdot T}\right) & R_2 < \mathrm{ST} \\ \boldsymbol{X}_i^t + Q \cdot L & R_2 \ge \mathrm{ST} \end{cases} \tag{7}$$

3.2.3 Step 3: scrounger assignment

During the producers' optimization process, scroungers will follow the producers for local optimization. As shown in Eq. (8), utilizing known high-quality regions and information, the first half of scroungers ($i > pN/2$) move toward the optimal producer direction, while the latter half move away from the worst region to avoid inferior solutions. In Eq. (9), X_{P}^t represents the current optimal producer position. X_{worst}^t represents the current worst position. $\mathbf{A}$ is a random matrix of ± 1, and i is the individual ranking index.

$$\boldsymbol{X}_i^{t+1} = \begin{cases} X_{\mathrm{P}}^t + |\boldsymbol{X}_i^t - X_{\mathrm{P}}^t| \cdot A^+ \cdot L & i > \frac{pN}{2} \\ Q \cdot \exp\left(\frac{X_{\mathrm{worst}}^t - \boldsymbol{X}_i^t}{i^2}\right) & i \le \frac{pN}{2} \end{cases} \tag{8}$$

3.2.4 Step 4: scout assignment

To enhance the algorithm's global search capability and prevent the population from falling into local optima during the optimization process, the SSA algorithm sets a portion of individuals as "scouts". Scouts execute two different position update strategies based on their current status, with the mathematical expression shown in Eq. (9).

Firstly, when the fitness is not optimal (i.e., $f_i \ne f_{\mathrm{g}}$), it indicates that the individual is still in a potentially valuable non-optimal region. In this case, its position update strategy is to move toward the current global best position X_{best}^t. This is achieved by adding a random step size factor $\beta \cdot |\boldsymbol{X}_i^t - X_{\mathrm{best}}^t|$, enabling local refined search and thereby enhancing the algorithm's convergence capability.

Secondly, when an individual's fitness reaches the global optimum (i.e., $f_i = f_{\mathrm{g}}$), it means the individual may be trapped in a local optimum. To break this stagnation and increase the diversity of hyperparameter solutions, its update strategy shifts to an adjustment based on the global worst position X_{worst}^t. Specifically, this is implemented by introducing a proportional term $K \cdot \frac{|\boldsymbol{X}_i^t - X_{\mathrm{worst}}^t|}{f_i - f_w + \epsilon}$, which incorporates a random step size factor K and the fitness difference between the current individual and the worst individual (a minimal constant ϵ is added to prevent the denominator from being zero and ensure numerical stability). This prompts the individual to escape the current optimal region and explore a wider solution space.

Finally, the fitness is recalculated based on the updated positions, and it is checked whether the termination conditions are met.

$$X_i^{t+1} = \begin{cases} X_{\text{best}}^t + \beta \cdot |X_i^t - X_{\text{best}}^t| & f_i \neq f_g \\ X_{\text{best}}^t + K \cdot \frac{|X_i^t - X_{\text{worst}}^t|}{f_i - f_w + \epsilon} & f_i = f_g \end{cases} \quad (9)$$

3.3 Algorithm implementation

After sample construction, preprocessing is performed using Z-score normalization and PCA dimensionality reduction. Temporal features are extracted via LSTM, followed by an attention mechanism to enhance key features. Hyperparameters are optimized using SSA, and the model is trained with the Adam optimizer, early stopping, and dropout to output the predicted step error. The detailed algorithm steps are as follows.

3.3.1 Step 1: data import and standardized preprocessing

Sample data are imported, including tool parameters (radius R, length L, inclination angle α), as well as discrete CC and CL points. The dataset is then batch-normalized by the Z-score standardization method described in Sect. 2.2.1.

3.3.2 Step 2: PCA dimensionality reduction and reconstruction

To eliminate redundant noise in high-dimensional data, PCA as described in Sect. 2.2.2 is applied to the standardized data for dimensionality reduction, and the reduced-dimensional output is generated.

3.3.3 Step 3: configure LSTM layer for temporal feature extraction

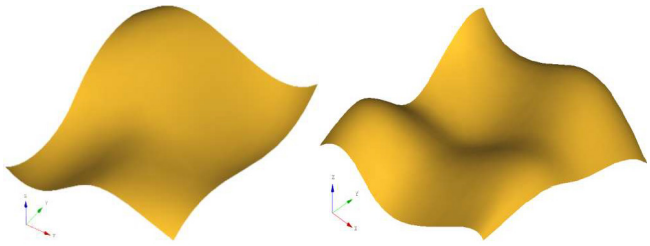
An LSTM network is used to extract continuous features from CC and CL point coordinates, learning the dependency relationships between adjacent CL points.

3.3.4 Step 4: configure attention mechanism layer

Following the method described in Sect. 3.1, an attention module is designed to weight the hidden states output by the LSTM, achieving adaptive weight allocation to reinforce key CC and CL point features.

3.3.5 Step 5: implement SSA for hyperparameter optimization

As outlined in Sect. 3.2, the hyperparameters (e.g., hidden layer dimensions, learning rate) for the attention-LSTM model are initialized. The global exploration and local exploitation mechanisms of SSA are utilized to iteratively optimize the hyperparameter combinations, avoiding local optima and enhancing the model's generalization capability.



(a) Surface S1 (b) Surface S2

Figure 4. Two surfaces.

3.3.6 Step 6: configure Adam optimization and dropout regularization layer

To mitigate issues such as local optima and slow convergence in gradient descent, the Adam optimizer is employed. A dropout algorithm is incorporated before the fully connected output layer to randomly discard outputs from some neurons, suppressing interference from non-critical parameters and reducing the risk of noise-induced overfitting during training.

3.3.7 Step 7: error evaluation and model output

The predicted step error values output by the model are compared with the true values generated by the geometric iterative method (Liu et al., 2012) and the mean squared error (MSE) is calculated to verify the model's prediction accuracy. If the MSE meets the predefined threshold, the model is considered to be optimized, and the predicted step error values are accepted as final outputs. Otherwise, the model parameters are further refined through backpropagation and iterative training until the convergence criterion is satisfied. The optimized model is then deployed for high-precision step error prediction.

4 Case study

4.1 Sample generation

The two surface models S1 and S2 shown in Fig. 4 are imported into the CAM software developed for this study. Isoparameter tool paths are generated with 99 and 100 lines respectively, each containing 150 and 200 CL points, respectively. The machining tool is a flat-end cutter with fillet radius of 0 mm, a radius R of 5 mm, and a length L of 50 mm. The inclination angle α is 15°.

While generating tool paths, six-dimensional features are collected between two adjacent CC points following the sample structure defined in Sect. 2.1, including 4 basic tool parameters, 30 features from 10 discrete CC points (3D coordinates for each point), 3 features from the 3D coordinates of the first CL point, 3 features from the corresponding tool axis vector of the first CL point (represented by x, y, and z val-

Table 2. Parameter and sample statistics table.

Surface	Lines of tool paths	Numbers of discrete CL points per line	Numbers of discrete CC points	Numbers of Samples
1	99	150	10	11 801 (training set), 2950 (test set)
2	100	200	10	15 920 (training set), 3980 (test set)

ues), 3 features from the tool axis vector of the second adjacent CL point, and 3 features from the 3D coordinates of the second CL point. These 46-D features constitute the input of each sample, while the corresponding step error value is the output. The true values of step error are calculated by the geometric method (Liu et al., 2020) during tool path generation. Their ranges for Surface S1 and S2 samples are [−24 µm, −6 µm] and [−4 µ m, −1 µm], respectively. All sample data are constructed according to the sample structure in Table 1 and exported to txt files. The detailed parameter settings and sample statistics for both surfaces are summarized in Table 2. A total of 80 % of the samples are randomly selected to form the training set, and the remaining samples form the test set.

4.2 Algorithm verification and comparison

This section uses the mean squared error (MSE) and coefficient of determination (R^2) in Eq. (10) to evaluate the prediction performance of the model by quantifying the error between the predicted step error values and the true values, where N denotes the total number of test samples, $\hat{y}_t$ represents the predicted step error value (in µm), y_t is the corresponding true step error value (in µm), and $\overline{y}$ denotes the mean value of all true step error values (in µm). A smaller MSE (in µm^2) and an R^2 (dimensionless) closer to 1 indicate higher prediction accuracy.

$$\begin{cases} \mathrm{MSE} = \frac{1}{N}\sum_{t=1}^{N}\left(\hat{y}_t - y_t\right)^2 \\ R^2 = 1 - \frac{\sum_{t=1}^{N}(y_t-\hat{y}_t)^2}{\sum_{t=1}^{N}(y_t-\overline{y})^2} \end{cases} \tag{10}$$

This section introduces three widely used regression models: the BP neural network, BiGRU, neural network and BiLSTM neural network (Liu et al., 2017, 2016), along with an unoptimized LSTM model and the proposed SSA-attention-LSTM model for step error prediction. The prediction results of each model are compared. The SSA algorithm is employed to optimize the neural network hyperparameters: learning rate (lr), number of neurons in hidden layer 1 (lstm1), number of neurons in hidden layer 2 (lstm2), and training epochs. The optimized values are shown in Table 3.

4.2.1 Comparison of SSA-attention-LSTM with mainstream network models

Samples from surfaces S1 and S2 were used to train the BP, LSTM, BiGRU, BiLSTM, and the proposed SSA-attention-LSTM neural network models, respectively. During training, the loss value of the proposed model decreased rapidly within the first 100 epochs and then stabilized, leading to early termination upon meeting the stopping criterion. In terms of computational efficiency, the proposed method demonstrates superior performance, with the geometric iterative method requiring 12.261 s to calculate step errors for 14 850 CL points, while the neural network prediction completes in just 3.6 s. The MSE and R^2 values of the five neural network models are presented in Tables 4 and 5, and the prediction results are illustrated in Figs. 5 and 6.

Table 3. Hyperparameters of SSA-attention-LSTM.

Surface	lr	lstm1	lstm2	epoch
S1	3×10^{-5}	56	73	1351
S2	1×10^{-5}	52	55	776

Table 4. Comparison of five neural network models for Surface S1.

Evaluation index	BP	LSTM	BiGRU	BiLSTM	SSA-attention-LSTM
MSE	6.909	3.465	1.552	2.013	0.197
R^2	0.773	0.886	0.948	0.933	0.993

The experimental results indicate that SSA-attention-LSTM achieves the highest prediction accuracy. The BP neural network performs the worst in predicting the step error for both surfaces S1 and S2 due to its simple structure and lack of sequence modeling capability, demonstrating its inadequacy for five-axis step error prediction. The LSTM model shows better prediction accuracy than the BP network. However, its performance is inferior to the other three models because its gating mechanism is sensitive to long-sequence features and because its hyperparameters are not optimized. While BiGRU and BiLSTM enhance temporal prediction capability through their bidirectional structures, the absence of an attention mechanism leads to insufficient focus on key features.

Table 5. Comparison of five neural network models for Surface S2.

Evaluation index	BP	LSTM	BiGRU	BiLSTM	SSA-attention-LSTM
MSE	0.206	0.056	0.035	0.038	0.003
R^2	0.741	0.928	0.944	0.941	0.995

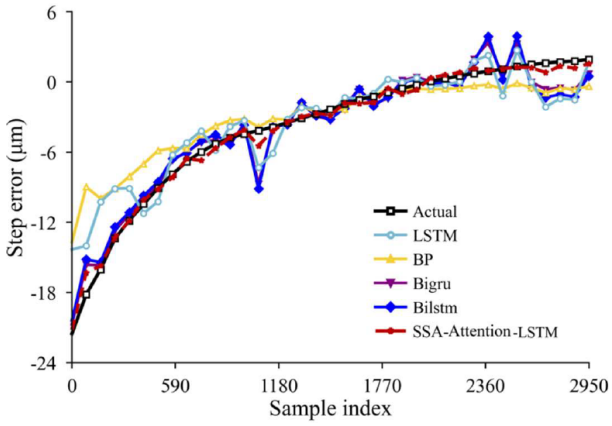


Figure 5. Step error prediction results of Surface S1.

4.2.2 Analysis of relative and absolute errors in SSA-attention-LSTM predictions

To understand the prediction accuracy of SSA-attention-LSTM in greater detail, the absolute and relative errors between its predicted values and the true values are analyzed. As shown in Tables 6 and 7, for Surface S1, 51.8 % of the absolute errors are less than 0.1 µm, and 98.1 % are less than 0.4 µm, with a mean error of 0.12 µm. For Surface S2, as illustrated in Fig. 8, the absolute values of the true step error for many samples are less than 1 µm. Here, 97.49 % of the absolute errors are less than 0.1 µm, and the remaining absolute errors are all less than 0.3 µm, resulting in a mean error of 0.02 µm.

The relative error is the ratio of the absolute error to the true value. When the step error values are small, even minor absolute errors can result in large relative errors. The relative errors between the predicted values of SSA-attention-LSTM and the true values are shown in Tables 8 and 9.

For Surface S1, 82.03 % of the relative errors are less than 15 %. Although 17.97 % of the samples have relative errors greater than 15 %, 97.3 % of these samples have absolute errors less than 0.4 µm. For Surface S2, although 36.61 % of the samples have relative errors greater than 15 %, the absolute errors of all of these samples are less than 0.1 µm. The detailed distributions of relative and absolute errors are illustrated in Fig. 9, demonstrating the high prediction accuracy of SSA-attention-LSTM.

To verify the effectiveness and accuracy of the SSA-attention-LSTM model in predicting step errors for blades,

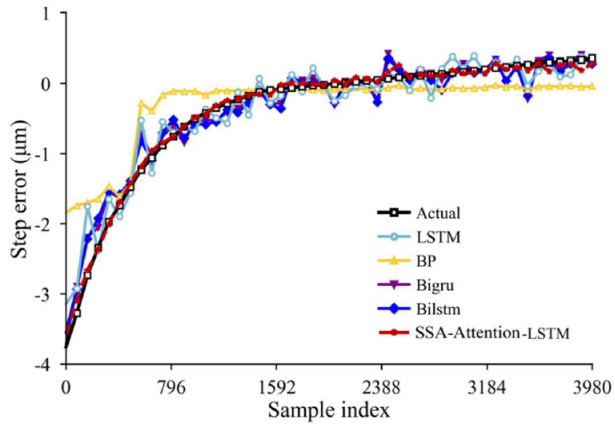


Figure 6. Step error prediction results of Surface S2.

which differ from free-form surfaces, the blade shown in Fig. 8 was selected as the test object. A filleted-end cutter with a radius of 3 mm, a length of 50 mm, and a fillet radius of 1.5 mm was adopted, with a lead angle of 15°. The finishing spiral tool path generation method (Liu et al., 2026) was used to plan 31 rows of blade tool paths. The blade surface was divided into four regions: leading edge, suction surface, posterior margin, and compressed air surface. The leading and posterior margins of the blade not only exhibited significantly higher absolute curvature values than the suction and compressed air surfaces but also showed more drastic curvature variations. Although the machining strokes along the tool path direction in these two regions were shorter, denser CC point arrangements had to be adopted to accurately characterize their complex surface profiles and ensure subsequent machining accuracy. The numbers of CC points for each tool path in the four regions were 79, 79, 79, and 79 for the training set and 159, 119, 159, and 119 for the test set, respectively. Based on the geometric parameters of all tool paths and their corresponding step error data, 9480 training samples and 16680 test samples were finally constructed.

In the blade test, the model structure and hyperparameters completely adopted the optimal configuration from Table 3, which was consistent with the sample experiments on free-form surfaces S1 and S2. PCA dimensionality reduction was applied to eliminate the redundant information of tool parameters and CC point coordinates in the original tool path data, which improved the model training efficiency by approximately 37 % and simultaneously suppressed the overfitting risk to further enhance the prediction accuracy. The results are shown in Table 10 and Fig. 7.

The model was trained based on the training set, and its prediction performance was evaluated using an independent test set. In the blade step error prediction task, this study also verified whether PCA dimensionality reduction would lose the key high-frequency geometric information affecting the curvature variation of the blade surface. Taking this blade model as the research object, the prediction perfor-

Table 6. Absolute errors of step error values predicted by SSA-attention-LSTM for Surface S1.

Interval	< 0.1 µm	0.1–0.2 µm	0.2–0.3 µm	0.3–0.4 µm	> 0.4 µm
Number	1528	898	352	116	56
Proportion	51.80 %	30.44 %	11.93 %	3.93 %	1.90 %

Table 7. Absolute errors of step error values predicted by SSA-attention-LSTM for Surface S2.

Interval	< 0.1 µm	0.1–0.2 µm	0.2–0.3 µm	0.3–0.4 µm	> 0.4 µm
Number	3880	83	17	0	0
Proportion	97.49 %	2.09 %	0.43 %	0	0

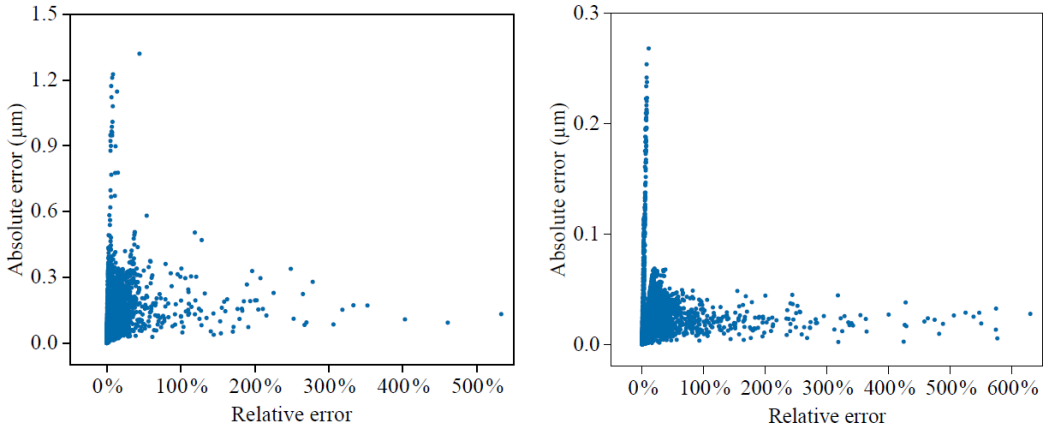


Figure 7. Scatterplot of relative absolute errors of Surface S1 and S2. **(a)** Surface S1. **(b)** Surface S2.

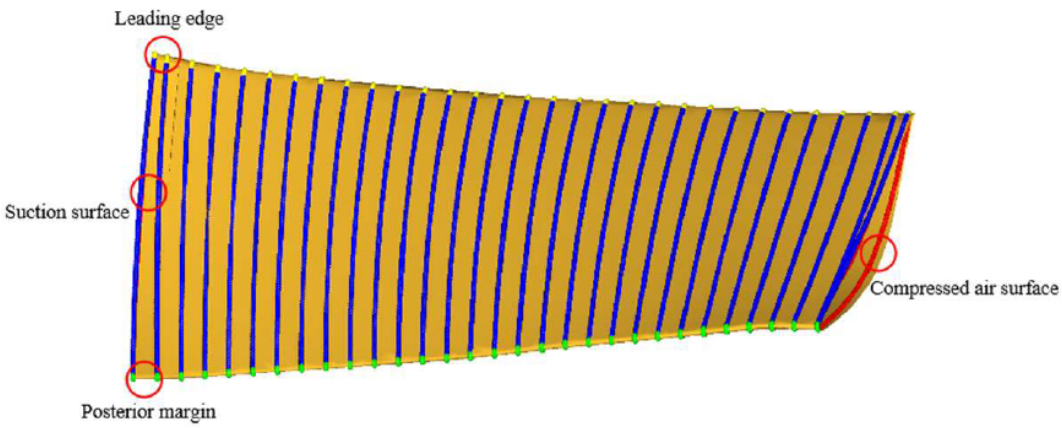


Figure 8. A 3D model of the blade.

mances of the model using all original high-dimensional features and those after PCA dimensionality reduction were compared. The results showed that PCA dimensionality reduction did not lose key geometric information; instead, it further improved the model's prediction accuracy and efficiency by filtering out redundant coordinate noise in the original tool path data. The SSA-attention-LSTM model with PCA preprocessing finally obtained the following prediction results: MSE $= 0.084 \times 10^{-3}$ µm^2, root mean squared error (RMSE) $= 9.145 \times 10^{-3}$ µm, $R^2 = 0.995$, mean absolute error (MAE) $= 7.39 \times 10^{-3}$ µm, and mean relative error (MRE) $= 0.69$ %. All five indicators indicated high prediction accuracy. These indicators can effectively ensure the objectivity of evaluation results under the micro-error magnitude.

Table 8. Relative errors of step error values predicted by SSA-attention-LSTM for Surface S1.

Interval	< 1 %	1 %–5 %	5 %–10 %	10 %–15 %	> 15 %
Number	655	1037	475	1765	530
Proportion	22.20 %	35.15 %	16.10 %	59.83 %	17.97 %

Table 9. Relative errors of step error values predicted by SSA-attention-LSTM for Surface S2.

Interval	< 1 %	1 %–5 %	5 %–10 %	10 %–15 %	> 15 %
Number	507	859	507	2016	1457
Proportion	12.74 %	21.58 %	12.74 %	50.65 %	36.61 %

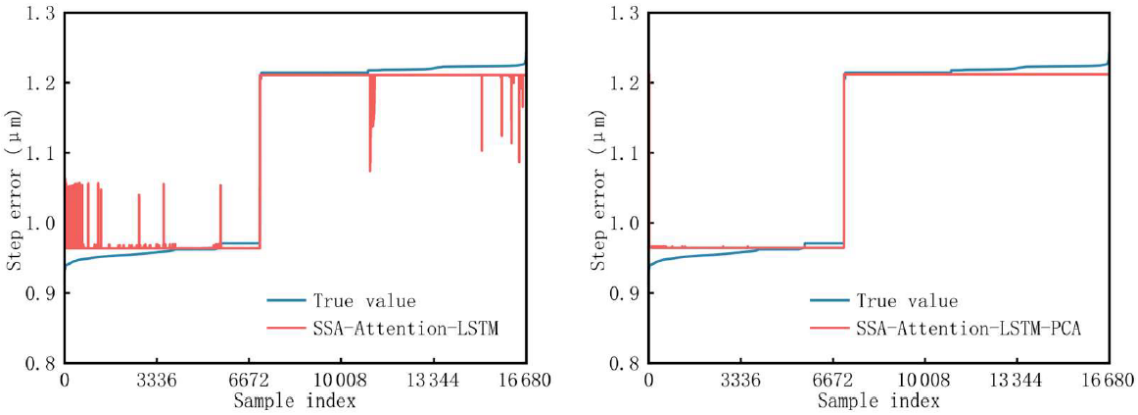


Figure 9. Prediction results of blade. **(a)** SSA-attention-LSTM. **(b)** SSA-attention-LSTM-PCA.

Table 10. Ablation study of blade prediction.

SSA-attention-LSTM	PCA	MSE	RMSE	R^2	Total time (s)
✓	–	0.261×10^{-3}	16.147×10^{-3}	0.985	56.37
✓	✓	0.084×10^{-3}	9.145×10^{-3}	0.995	35.49

In terms of computational efficiency, during the generation of the blade sample set, the geometric iterative method took a total of 67.56 s to calculate the step errors of all 9480 training samples; the neural network model in this paper took 33.16 s to train using 9480 training samples and then took 2.33 s to predict 16 680 test samples, with the entire process requiring only 35.49 s, fully verifying the efficiency advantage of the model in practical machining scenarios.

5 Conclusion

In the field of five-axis machining step error calculation, traditional geometry-based iterative algorithms require repeated iterations to solve the maximum distance between the tool envelope surface and the cutter contact (CC) curve. Their computational load increases exponentially with higher-precision requirements, and historical computation experiences cannot be reused. The research team previously implemented BP neural networks to predict step error in three-axis machining. However, BP models are constrained by single-task learning frameworks and shallow network architectures, making it difficult to effectively capture the high-dimensional nonlinear characteristics of five-axis machining step error, resulting in insufficient prediction accuracy and limited generalization capability. This paper proposes a collaborative optimization framework based on SSA-attention-LSTM. The SSA dynamically optimizes LSTM hyperparameters to avoid local optima and accelerate model convergence. An attention mechanism is introduced to achieve weighted focusing on key CC and CL point features, enhancing the core geometric information in tool paths. Combined with PCA dimensionality reduction preprocessing to eliminate redundant co-

ordinate interference, the model's robustness is significantly improved. This method breaks through the inherent pattern of iterative solution based on geometric relationships, enabling efficient calculation of five-axis step error using neural networks. Compared to single-task models such as LSTM and BP, its hyperparameter optimization mechanism and feature focusing capability are better aligned with the high-dimensional nonlinear characteristics of complex surfaces. In predictions for surfaces S1 and S2, R^2 values increased to 0.993 and 0.995, respectively, with absolute errors controlled within the micrometer range. In the blade test case, high-precision prediction was achieved with only 9480 training samples, and PCA dimensionality reduction preprocessing improved the training efficiency by approximately 37 %, with an R^2 of 0.995, which verified the feasibility and effectiveness of the model and provided a theoretical and algorithmic foundation for the development of CAM software.

Future research will focus on constructing a transfer learning framework to adapt the trained model to new surfaces, achieving high-precision prediction results with minimal training samples for the neural network. On this basis, future research will further investigate the introduction of physical factors in the machining process, such as insufficient tool stiffness, tool wear, machine tool vibration, and thermal deformation, to enable the network to accurately predict the actual errors of parts after machining.

Data availability. The datasets generated and analyzed during this study are not publicly available as they belong to ongoing research projects of the research group and involve proprietary geometric parameters related to blade machining. The datasets are available from the corresponding author upon reasonable request.

Author contributions. Te Ye: conceptualization, methodology, software, writing (original draft). Wei Liu: supervision, funding acquisition. Jiaping Zhang: validation. Jiawei Chen: data curation. Jingyang Yu: visualization. Yuhang Zhao: writing (review and editing). Ziyu Zhang: investigation.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. The authors bear the ultimate responsibility for providing appropriate place names. Views expressed in the text are those of the authors and do not necessarily reflect the views of the publisher.

Financial support. This research has been supported by the Ministry of Industry and Information Technology of the People's Republic of China, Great Project of Ministry of Industrialization and Information of China (grant no. HK202403640) and the Natural Science Foundation of Jiangsu Province, Basic Research Program of Jiangsu Province (grant no. BK20210865).

Review statement. This paper was edited by Jeong Hoon Ko and reviewed by Mircea Viorel DRĂGOI and one anonymous referee.

References

Al-Faiz, M. Z., Ibrahim, A. A., and Hadi, S. M.: The effect of Z-Score standardization (normalization) on binary input due the speed of learning in back-propagation neural network, Iraqi Journal of Information and Communication Technology, 1, 42–48, 2018.

Cascino, A., Delle Monache, S., Lanzillo, L., Mazzeo, F., Nencioni, L., Nicolella, A., Strano, S., and Terzo, M.: Development and Experimental Testing of a 3D Vision System for Railway Freight Wagon Monitoring, Appl. Sci., 15, 11547, https://doi.org/10.3390/app152111547, 2025.

Chen, T., Wang, L., Liu, Y., and Chen, C. P.: Double-kernel based Bayesian approximation broad learning system with dropout, Neurocomputing, 610, 128533, https://doi.org/10.1016/j.neucom.2024.128533, 2024.

Eskandari, S., Arezoo, B., and Abdullah, A.: Positional, geometrical, and thermal errors compensation by tool path modification using three methods of regression, neural networks, and fuzzy logic, Int. J. Adv. Manuf. Tech., 65, 1635–1649, 2013.

Fariz, T. N. and Basha, S. S.: Enhancing solar radiation predictions through COA optimized neural networks and PCA dimensionality reduction, Energy Reports, 12, 341–359, 2024.

Geng, L., Zhang, Y. F., and Fuh, J. H.: A neural network based approach to 5-axis tool-path length estimation for optimal multi-cutter selection, Computer-Aided Design and Applications, 8, 301–313, 2011.

Kingma, D. P. and Ba, J. A.: A method for stochastic optimization, arXiv [preprint], https://doi.org/10.48550/arXiv.1412.6980, 2014.

Liu, W., Zhou, L. S., and An, L. L.: Constant scallop-height tool path generation for three-axis discrete data points machining, Int. J. Adv. Manuf. Tech., 63, 137–146, 2012.

Liu, W., Zhang, J. W., Cao, Z. Y., Zhu, S. M., and Yuan, T. J.: Direct 5-axis tool posture local collision-free area generation for point clouds, Int. J. Adv. Manuf. Tech., 86, 2055–2067, 2016.

Liu, W., Zhang, J. W., Zhu, S. M., Zhang, C. C., and Yuan, T. J.: Efficient tool posture global collision-free area generation for 5-axis point clouds machining, Int. J. Adv. Manuf. Technol., 88, 1013–1023, 2017.

Liu, W., Zhu, S. M., Huang, T., and Zhou, C.: An efficient iso-scallop tool path generation method for three-axis scattered point cloud machining, Int. J. Adv. Manuf. Tech., 107, 3471–3483, 2020.

Liu, W., Fan, L. Y., Zhu, Q. X., Zhu, S. M., Wang, T. L., and Tang, F.: Five-axis iso-error numerical control tool path generation for

flat-end tool machining sculptured surface, Int. J. Adv. Manuf. Tech., 119, 7503–7516, 2022.

Liu, W., Kang, J., Zhang, J. P., Zhang, Z. Y., and Li, P. F.: Five-axis numerical control helical tool path generation for bullnose milling cutters finishing machining blades, Int. J. Adv. Manuf. Tech., 1–17, https://doi.org/10.1007/s00170-026-17429-7, 2026.

Salem, H., Kabeel, A. E., El-Said, E. M., and Elzeki, O. M.: Predictive modelling for solar power-driven hybrid desalination system using artificial neural network regression with Adam optimization, Desalination, 522, 115411, https://doi.org/10.1016/j.desal.2021.115411, 2022.

Sun, C., Dominguez-Caballero, J., Ward, R., Ayvar-Soberanis, S., and Curtis, D.: Machining cycle time prediction: Data-driven modelling of machine tool feedrate behavior with neural networks, Robot. Com.-Int. Manuf., 75, 102293, https://doi.org/10.1016/j.rcim.2021.102293, 2022.

Wu, P. H., Li, Y. W., and Chu, C. H.: Optimized tool path generation based on dynamic programming for five-axis flank milling of rule surface, Int. J. Mach. Tool. Manu., 48, 1224–1233, 2008.

Zhang, Z. Y., Liu, W., Li, P. F., Zhang, J. P., and Fan, L. Y.: A numerical control machining tool path step error prediction method based on BP neural network, Sci. Rep., 13, 16347, https://doi.org/10.1038/s41598-023-43617-6, 2023.

Zheng, D., Li, L., Zheng, S., Chai, X., Zhao, S., Tong, Q., Wang, J., and Guo, L.: A Defect Detection Method for Rail Surface and Fasteners Based on Deep Convolutional Neural Network, Comput. Intel. Neurosc., 2021, 2565500, https://doi.org/10.1155/2021/2565500, 2021.

Zhou, Y., Chen, Z. C., and Yang, X.: An accurate, efficient envelope approach to modeling the geometric deviation of the machined surface for a specific five-axis CNC machine tool, Int. J. Mach. Tool. Manu., 95, 67–77, 2015.