scientific reports

OPEN

Combined contour error control method for five-axis machine tools based on digital twin

Liuquan Wang[1,2], Ruijie Yang[1,2], Shisheng Lv[1,3], Zhiqi Yang[1,4], Shuwei Xin[1,5], Yanqiang Liu[1,2]✉ & Qiang Liu[1,2]✉

Contour error is a critical factor influencing machining quality. This paper proposes a combined contour error control method for five-axis machine tools based on digital twin. The proposed method combines pre-compensation implemented in digital twin with feedback control in the real-time controller. After obtaining the tool path input, the digital twin performs interpolation and applies model predictive pre-compensation control to the interpolated commands to control modeled errors. The pre-compensated commands and interpolation data are sent to the real-time controller where contour error is estimated and controlled in each control cycle through feedback control to control unmodeled errors. Using the S-shaped curve as the test case, the maximum tool tip position contour error is reduced by 77.78%, with an average reduction of 83.90%. The maximum tool orientation contour error decreased by 79.05%, with an average reduction of 86.66%. The experimental results demonstrate that the proposed method significantly reduces tool contour error.

Keywords Contour error control, Digital twin, Model predictive pre-compensation control, Five-axis machine tool

Five-axis machine tools are widely used for machining complex parts. The limited dynamic bandwidth of servo axes leads to tracking errors during motion. These tracking errors are kinematically transferred to the tool of five-axis machine tools and result in tool deviations from the intended tool path. These deviations are defined as contour error.

Improving the bandwidth of each axis can reduce contour error indirectly but cannot address the dynamic performance mismatches between axes. Therefore, direct control of contour error is currently the focus of research, which can be divided into two directions: contour error estimation (CEE) and contour error control (CEC).

For CEE, since real-time controllers operate in discrete systems, discrete interpolation points are typically used to approximate the ideal curve for contour error estimation. Yang et al.[1] proposed a fast iterative method that approximates the ideal contour as a straight line. Yang et al.[2] used three interpolation points to fit an intended arc for calculating the contour error. Chen et al.[3] conducted a comparative analysis of the computational efficiency of various approximation methods, including straight lines, arcs, and their linear combinations. Pi et al.[4] used Ferguson curves to refit the interpolation points. Liu et al.[5] proposed a CEE method based on Taylor series expansion and the Frenet-Serret frame to consider the effects of curvature and torsion. Yang et al.[6] identified high-curvature regions and modified the contour error vector to avoid abrupt changes. Wang et al.[7] proposed an iterative method with Aitken acceleration. Li et al.[8] employed a moving window method to compare reference positions and search for the optimal solution. Ghaffari et al.[9] proposed a dynamic CEE algorithm based on the Newton update algorithm.

For CEC, typical control methods include feedback control, pre-compensation control, and predictive control[10]. Control structures of these three methods are illustrated in Fig. 1.

Feedback control calculates the current contour error in each control cycle, decomposes the contour error into each axis, and compensates it by adjusting the control inputs of each axis in the next cycle. Lu et al.[11] designed a single-neuron adaptive cross-coupled controller, which dynamically adjusts its weights through

[1]School of Mechanical Engineering and Automation, Beihang University, Beijing 100191, China. [2]Jiangxi Research Institute of Beihang University, Nanchang 330096, China. [3]Beijing Engineering Technological Research Center of High-Efficient and Green CNC Machining Process and Equipment, Beijing 100191, China. [4]Research and Application Center of Advanced CNC Machining Technology, State Administration of Science, Technology and Industry for National Defense, Beijing 100191, China. [5]Jiangxi Key Laboratory of High-end CNC Machine Tools, Nanchang 330096, China. ✉email: liuyanqiang@buaa.edu.cn; qliusmea@buaa.edu.cn

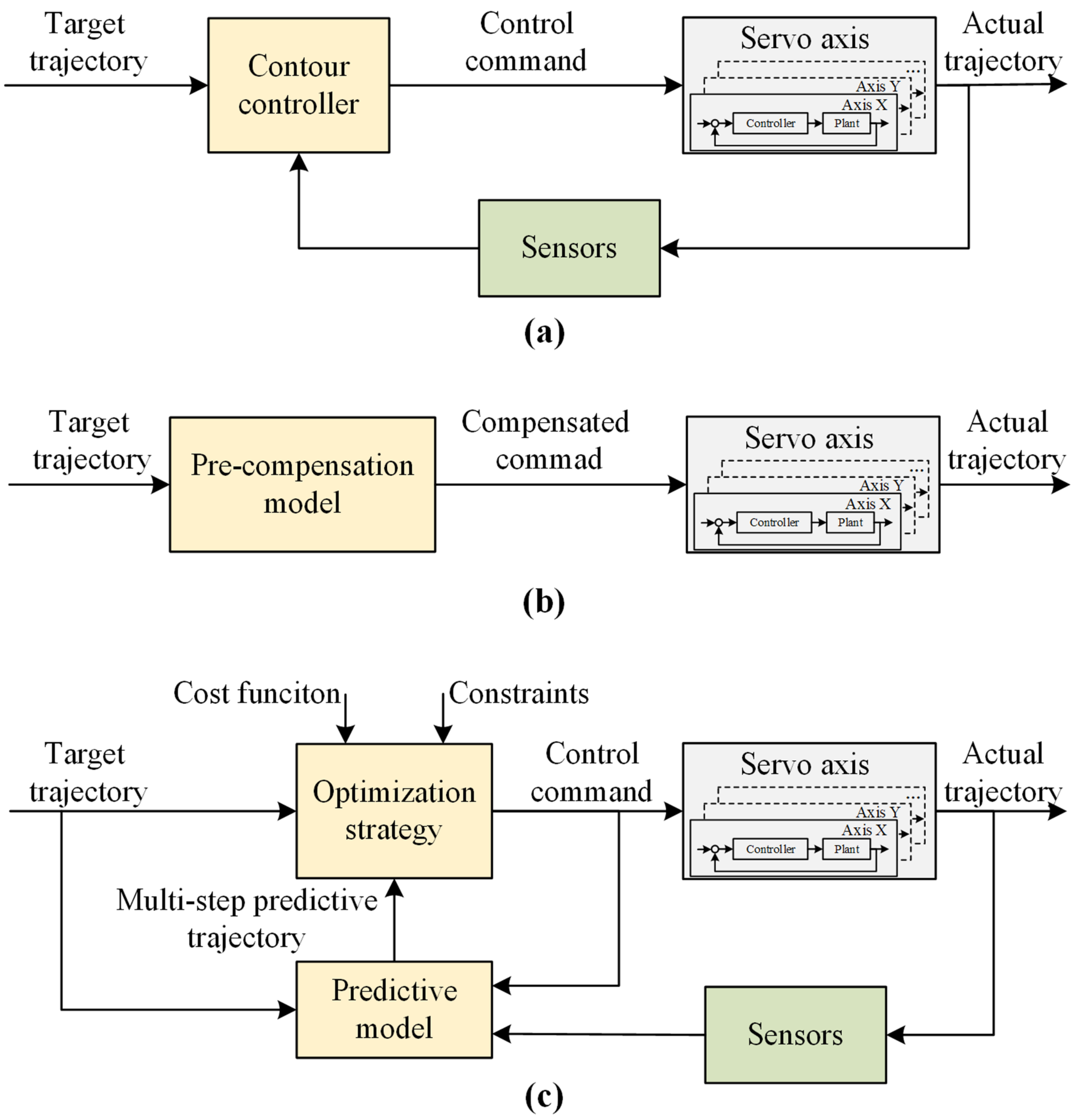


Fig. 1. Three contour error control methods: (**a**) feedback control; (**b**) pre-compensation control; (**c**) predictive control.

a single-neuron learning algorithm to enhance control precision. Zhao et al.[12] proposed a controller that integrates feedforward, feedback, and cross-coupled control. Feedback control is simple to implement and can handle the effects of unknown disturbances. However, feedback control always lags the contour error. To address these limitations, researchers have investigated pre-compensation control. Pre-compensation control optimizes control commands in advance through simulation or modeling, enabling compensation before contour errors occur. Pre-compensation control does not rely on the real-time state of the system, so it can be completed in offline conditions. Xu et al.[13] proposed a 1DCNN-BiLSTM-Attention model to predict and compensate contour errors. Wang et al.[14] proposed an iterative pre-compensation method based on a predictive model. Xiao et al.[15] proposed a model predictive contour error pre-compensation method. Wang et al.[16] analyzed the composition of contour errors in three-axis machine tools and designed a pre-compensation controller. Pi et al.[17] further considered the influence of cutting force disturbances. Lyu et al.[18] and Duong et al.[19] achieved pre-compensation by offline adjustment of servo control gains. Although pre-compensation method can control contour error in advance, its effectiveness depends on the model accuracy which is not easy to fulfill. Recently, predictive control methods such as MPC[20–22] and generalized predictive control (GPC)[5,23] have been applied to contour error control. Based on the current state of the machine tool, predictive control solves multi-objective optimization problems by predicting future states over multiple steps, thereby obtaining the optimal control commands for the current cycle. Compared with pre-compensation control, predictive control compensates based on the system's current actual state, so it can compensate for errors caused by unmodeled disturbances. Compared with feedback control, predictive control compensates through multi-step prediction and rolling optimization, so the compensation does not lag the occurrence of errors. However, predictive control involves numerous matrix calculations, which result in a significant computational load on the real-time controller. The application of MPC and GPC in contour error control is mainly focused on two-axis or three-axis platforms. Therefore, it is necessary to consider combining those methods to fully leverage the advantages of each.

As a digital mirror of the real world, digital twin technology has obtained significant attention in recent years[24]. Many scholars have explored the application of digital twin in CNC machine tools for commissioning[25,26], virtual machining[27,28], optimization[29–32] and monitoring[33,34]. However, in the above research, digital twin primarily optimizes the machining process by adjusting feedrate override and few studies have explored combining digital twins with real-time controllers for contour error control. Digital twin can synchronously update with physical machine tools[35,36] and handle greater computational loads through cloud computing service[37], which meets the requirements of model precision and computation capability of MPC. Therefore, developing a combined contour error control method based on the digital twin effectively meets the requirements for both control accuracy and computational load.

This paper presents a combined contour error control method based on the digital twin. After obtaining the tool path input, the digital twin performs interpolation and applies model predictive pre-compensation control (MPCC) to the interpolated commands to control modeled errors. The pre-compensated commands and interpolation data are sent to the real-time controller which estimates and controls contour error in each control cycle by feedback control to control unmodeled errors.

The remaining sections of this paper are organized as follows: "Principle of proposed method" introduces the principle of the proposed method. Section "Digital twin for contour error control" describes the software architecture of the digital twin, as well as the servo dynamic model and the machine kinematics model. Section "Combined contour error control method" presents the offline model predictive pre-compensation control and the online feedback control method. Finally, "Experiment" demonstrates the experimental results and discusses their implications.

Principle of proposed method

The combined contour error control method for five-axis machine tools based on digital twin is shown in Fig. 2. The contour error control is divided into two steps: offline pre-compensation implemented in the digital twin and online control executed in the real-time controller.

The NURBS toolpath and processing information generated by CAM are input into the digital twin. The NURBS direct interpolator of the digital twin processes the tool path to generate interpolation commands. Subsequently, the simulation module, which includes the machine tool kinematics model and servo dynamics model, executes simulations of the interpolation commands. Meanwhile, the pre-compensation module simultaneously compensates for the interpolation commands and ultimately generates the pre-compensated commands. Then the interpolation commands, the pre-compensated commands, and other required data are sent to the real-time controller via the Beckhoff Automation Device Specification (ADS) protocol.

In the real-time controller, the online controller estimates the contour error in each control cycle and generates compensation commands. The resulting commands are then transmitted via the industrial ethernet to the servo drives. The real-time data (axis position, axis velocity, axis current, etc.) sampled at each interpolation cycle is also transmitted to the digital twin adapter via EtherCAT and formatted into structured data. Then the formatted data is sent to the database as machining history data for storage via ethernet. The machine status data is transmitted to the digital twin monitoring module via ADS every 100 ms. The digital twin periodically reads machining historical data from the database, analyzes machining performance and updates simulation model parameters through parameter identification module. If the machining quality still fails to meet requirements, the feedrate and toolpath can be further optimized using methods from the reference[30,38–40] to enhance the accuracy of subsequent machining processes.

Digital twin for contour error control

Architecture of the digital twin

The functional architecture of the digital twin developed in this paper is shown in Fig. 3. The software was developed using C + + on the Visual Studio 2019 platform, with the visualization interface implemented using Qt 5.

Servo dynamic model

Currently, the servo feed axis commonly employs proportional-proportional-integral (PPI) control. The control model of servo axis is illustrated in Fig. 4.

Where K_p is the proportional gain coefficient of the position loop. K_v is the proportional gain coefficient of the velocity loop. K_iv is the integral gain coefficient of the velocity loop. K_t *is the torque constant of motor.* J_m is the equivalent mass. C_m is the equivalent damping coefficient and r_g is the transmission coefficient. T_load is the external load torque including Coulomb friction f_c. p_cmd is the command input position, p_act is the actual response position, $\omega_\mathrm{\ act}$ is the actual angular velocity, o_act is the output of the velocity loop integrator. Let the

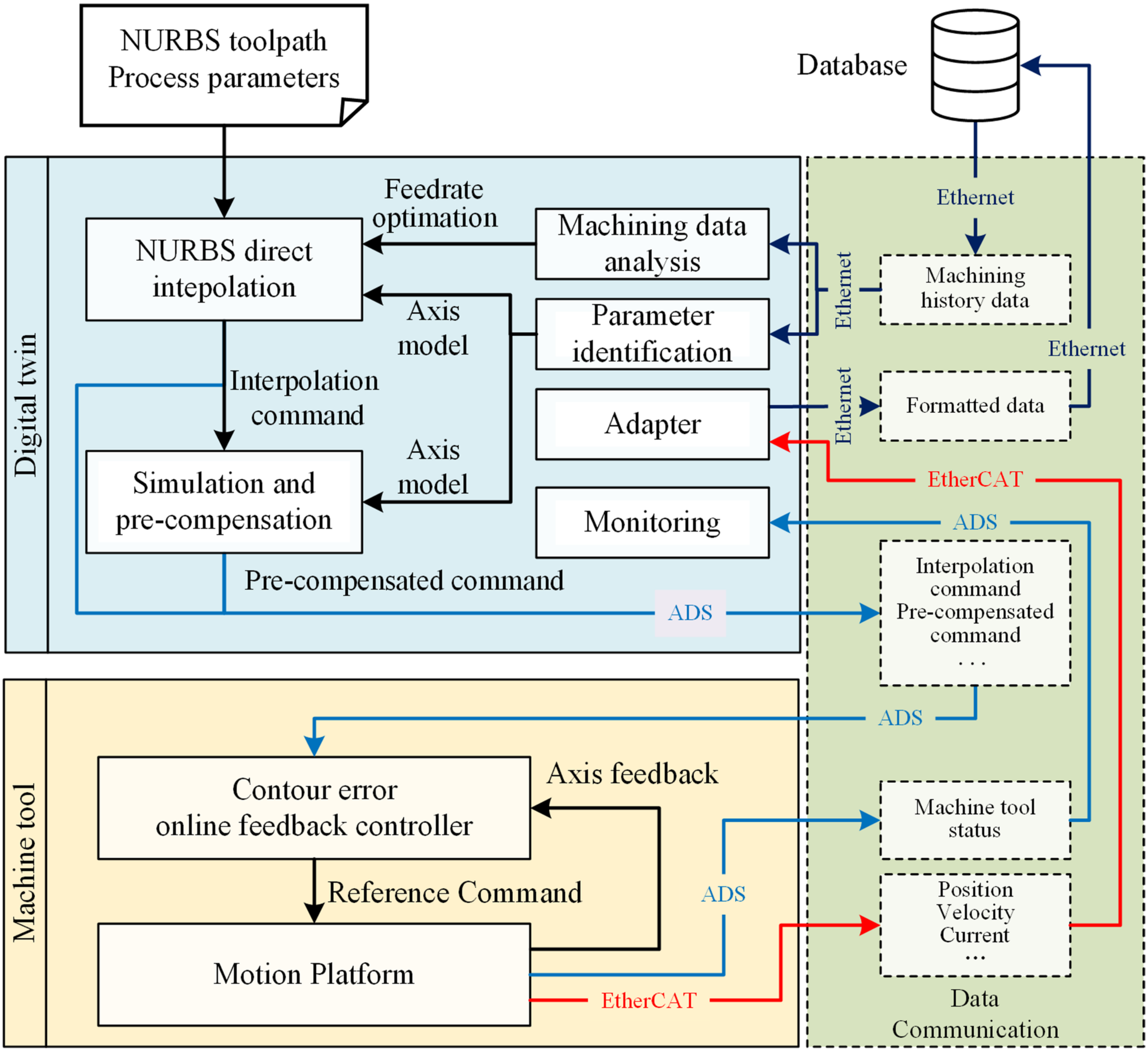


Fig. 2. Schematic diagram of combined contour error method for five-axis machine tools based on digital twin.

state vector be $\boldsymbol{x} = \begin{bmatrix} p_{\mathrm{act}} & \omega_{\ \mathrm{act}} & o_{\mathrm{act}} \end{bmatrix}^{\mathrm{T}}$, output vector be $\boldsymbol{y} = [p_{\mathrm{act}}]$, input vector be $\boldsymbol{u} = [p_{\mathrm{cmd}}]$ and disturbance vector be $\boldsymbol{d} = [T_{\mathrm{load}}]$, the above system can be described as Eq. (1).

$$\begin{cases} \dot{\boldsymbol{x}} = \boldsymbol{A}_{\mathrm{c}}x + \boldsymbol{B}_{\mathrm{c}}u + \boldsymbol{F}_{\mathrm{c}}d \\ y = Cx \end{cases} \tag{1}$$

where

$$\boldsymbol{A}_{\mathrm{c}} = \begin{bmatrix} 0 & r_{\mathrm{g}} & 0 \\ -\frac{K_{\mathrm{p}}K_{\mathrm{v}}K_{\mathrm{t}}}{J_{\mathrm{m}}} & -\frac{K_{\mathrm{v}}K_{\mathrm{t}}+C_{\mathrm{m}}}{J_{\mathrm{m}}} & \frac{K_{\mathrm{t}}}{J_{\mathrm{m}}} \\ -K_{\mathrm{p}}K_{\mathrm{iv}} & -K_{\mathrm{iv}} & 0 \end{bmatrix}, \boldsymbol{B}_{\mathrm{c}} = \begin{bmatrix} 0 \\ \frac{K_{\mathrm{p}}K_{\mathrm{v}}K_{\mathrm{t}}}{J_{\mathrm{m}}} \\ K_{\mathrm{p}}K_{\mathrm{iv}} \end{bmatrix}, \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, \boldsymbol{F}_{\mathrm{c}} = \begin{bmatrix} 0 \\ -\frac{1}{J_{\mathrm{m}}} \\ 0 \end{bmatrix}$$

Assuming that $\boldsymbol{u}$ and $\boldsymbol{d}$ remain constant in an interpolation cycle, the system described by Eq. (1) can be discretized using a zero-order hold (ZOH). The resulting discrete system can be expressed as Eq. (2):

$$\begin{cases} x\left(k+1\right) = Ax\left(k\right) + Bu\left(k\right) + Fd\left(k\right) \\ y\left(k\right) = Cx\left(k\right) \end{cases} \tag{2}$$

where

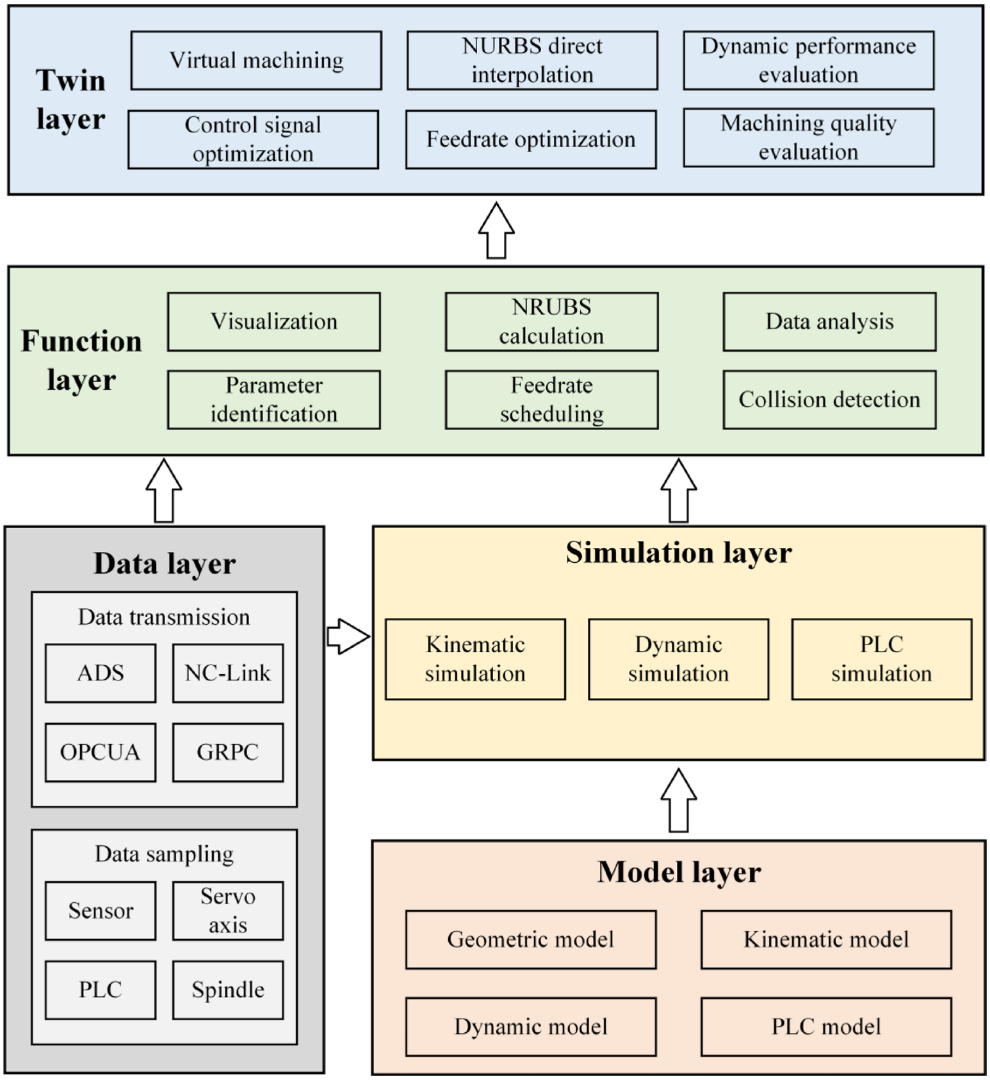


Fig. 3. Architecture of the digital twin.

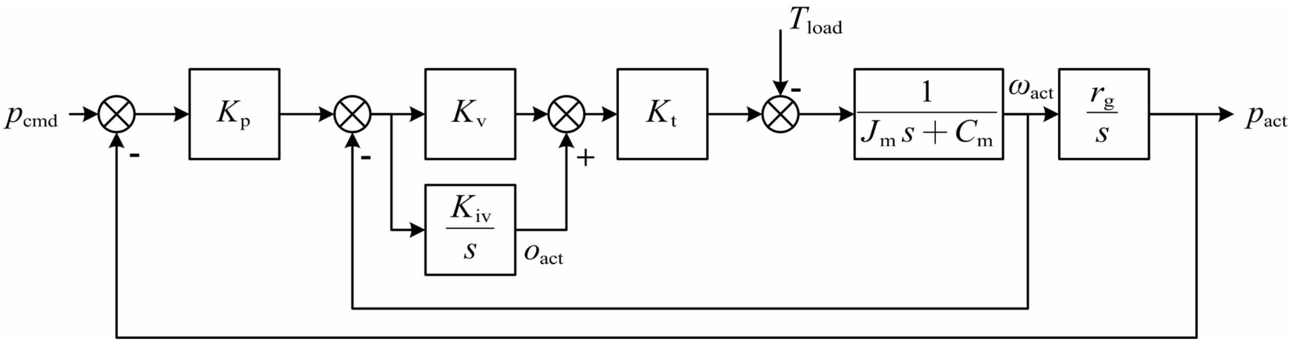


Fig. 4. Servo dynamic model.

$$\boldsymbol{A}=\mathrm{e}^{\boldsymbol{A}_{\mathrm{c}q}T_s},\boldsymbol{B}=\left(\int_0^{T_s}\mathrm{e}^{\boldsymbol{A}_{\mathrm{c}}t}\mathrm{d}t\right)\boldsymbol{B}_{\mathrm{c}},\boldsymbol{F}=\left(\int_0^{T_s}\mathrm{e}^{\boldsymbol{A}_{\mathrm{c}}t}\mathrm{d}t\right)\boldsymbol{F}_{\mathrm{c}}$$

The position loop proportional gain coefficient, velocity loop proportional gain coefficient, velocity loop integral gain coefficient, and torque constant in the above model can typically be obtained directly from the servo driver. Parameters such as mass, viscous friction, and Coulomb friction can be identified using the methods described in references[4,41].

Machine tool kinematic model

As illustrated in Fig. 5, this paper uses an AC head-type CNC machine tool as an example. The forward and inverse kinematics transformations are established using the method described in reference[42].

Let the tool tip coordinates be denoted as $[p_{\mathrm{t}x},p_{\mathrm{t}y},p_{\mathrm{t}z}]^{\mathrm{T}}$, the tool orientation vector as $[O_{\mathrm{t}i},O_{\mathrm{t}j},O_{\mathrm{t}k}]^{\mathrm{T}}$. and servo axis coordinates as $[p_{\mathrm{m}x},p_{\mathrm{m}y},p_{\mathrm{m}z},p_{\mathrm{m}a},p_{\mathrm{m}c}]^{\mathrm{T}}$ The forward kinematics transformation is expressed in Eq. (3).

$$\left\{\begin{array}{c} p_{\mathrm{t}x}=p_{\mathrm{m}x}-x_0+Lsinp_{\mathrm{ma}}sinp_{\mathrm{mc}}\\ p_{\mathrm{t}y}=p_{\mathrm{m}y}-y_0-Lsinp_{\mathrm{ma}}cosp_{\mathrm{mc}}\\ p_{\mathrm{t}z}=p_{\mathrm{m}z}-z_0+Lcosp_{\mathrm{ma}}\\ O_{\mathrm{t}i}=-sin\left(p_{\mathrm{ma}}-a_0\right)sin\left(p_{\mathrm{mc}}-c_0\right)\\ O_{\mathrm{t}j}=sin\left(p_{\mathrm{ma}}-a_0\right)cos\left(p_{\mathrm{mc}}-c_0\right)\\ O_{\mathrm{t}k}=-cos\left(p_{\mathrm{ma}}-a_0\right)\end{array}\right. \tag{3}$$

where x_0,y_0,z_0,a_0,c_0 are the zero offsets from the workpiece coordinate system (WCS) to the machine coordinate system (MCS). L is the tool length and in this paper $L=75$ mm.

From Eq. (3), the inverse kinematics equations can be obtained as Eq. (4).

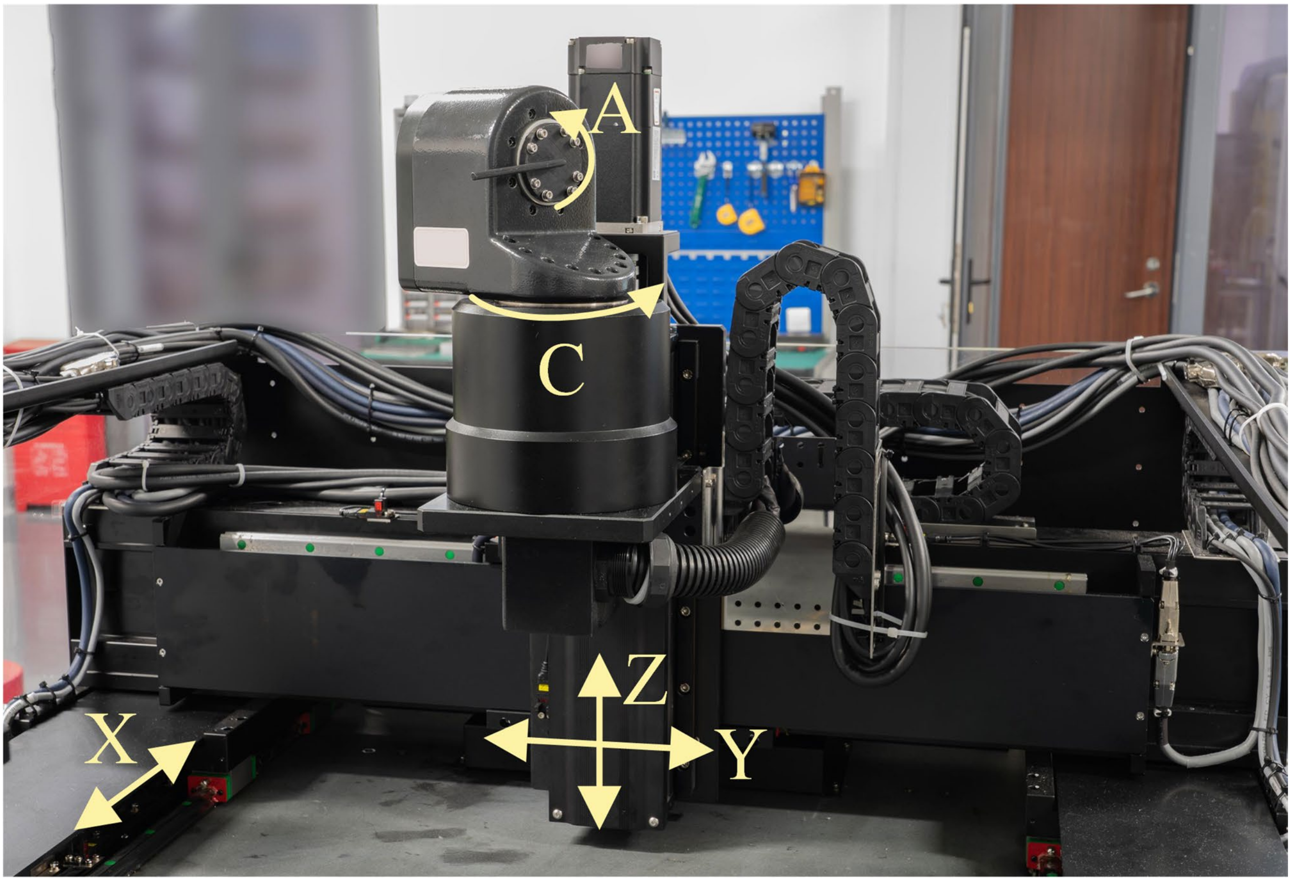


Fig. 5. AC head-type motion platform.

$$\begin{cases} p_{\mathrm{ma}} = arccos\left(-O_{\mathrm{t}k}\right) \\ p_{\mathrm{mc}} = arctan\left(-\frac{O_{\mathrm{t}i}}{O_{\mathrm{t}j}}\right) \\ p_{\mathrm{m}x} = p_{\mathrm{t}x} + x_0 - Lsinp_{\mathrm{ma}}sinp_{\mathrm{mc}} \\ p_{\mathrm{m}y} = p_{\mathrm{t}y} + y_0 + Lsinp_{\mathrm{ma}}cosp_{\mathrm{mc}} \\ p_{\mathrm{m}z} = p_{\mathrm{t}z} + z_0 - Lcosp_{\mathrm{ma}} \end{cases} \tag{4}$$

Combined contour error control method

Definition of contour error

As shown in Fig. 6, $\boldsymbol{C}\left(u\right)$ is the ideal tool tip position path, and $\boldsymbol{P}_{\mathrm{rt}}, \boldsymbol{O}_{\mathrm{rt}}$ are the ideal interpolation tool tip position (TTP) and tool orientation (TORI) vectors. $\boldsymbol{P}_{\mathrm{at}}\left(k\right), \boldsymbol{O}_{\mathrm{at}}\left(k\right)$ represent the actual TTP and TORI at the k-th cycle. The five-axis contour error encompasses both TTP error and TORI error. In current research, contour error is often defined as the shortest distance or normal distance from the actual position to the ideal trajectory. The point on the ideal trajectory closest to the actual position is identified as the foot point $\boldsymbol{P}_{\mathrm{f}}$, with the corresponding tool orientation being $\boldsymbol{O}_{\mathrm{f}}$. Considering computational stability and efficiency, this paper employs the iterative method proposed in reference[1] to find $\boldsymbol{P}_{\mathrm{f}}$.

Since the interpolation cycle is typically very short, the command trajectory between two adjacent interpolation points can be approximated as a straight line. At this time, $\boldsymbol{P}_{\mathrm{f}}$ lies on the line $\overrightarrow{\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)\boldsymbol{P}_{\mathrm{rt}}\left(i\right)}$ formed by the two adjacent interpolation points $\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right), \boldsymbol{P}_{\mathrm{rt}}\left(i\right)$, and can be expressed as

$$\boldsymbol{P}_{\mathrm{f}} = \boldsymbol{P}_{\mathrm{rt}}\left(i-1\right) - h_i\left(\boldsymbol{P}_{\mathrm{rt}}\left(i\right) - \boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)\right) \tag{5}$$

where

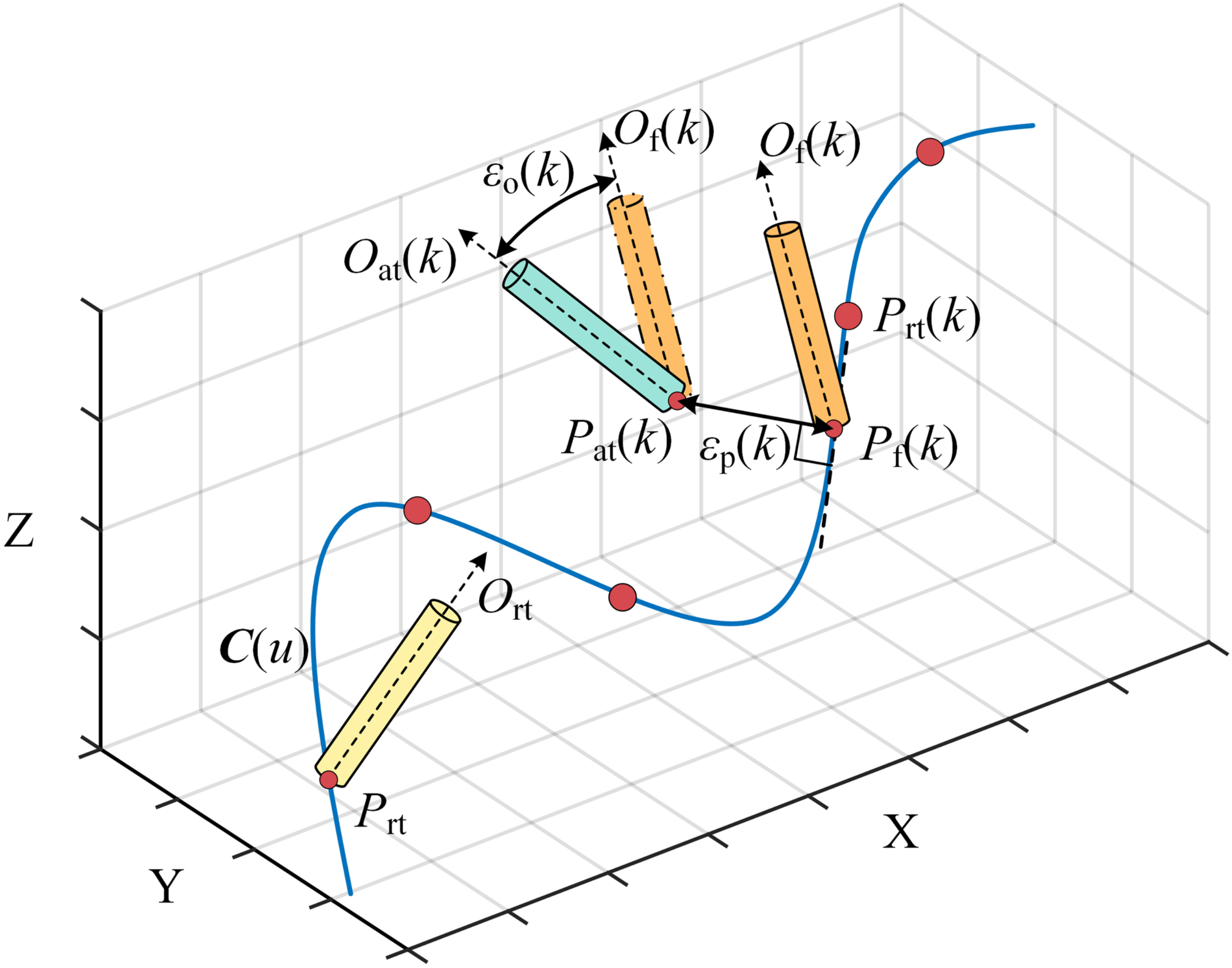


Fig. 6. Schematic diagram of contour error.

$$h_i = \frac{\overrightarrow{\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)\boldsymbol{P}_{\mathrm{f}}}}{\overrightarrow{\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)\boldsymbol{P}_{\mathrm{rt}}\left(i\right)}}$$

From the geometric relationship, it is known that $\overrightarrow{\boldsymbol{P}_{\mathrm{at}}\left(k\right)\boldsymbol{P}_{\mathrm{f}}} \perp \overrightarrow{\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)\boldsymbol{P}_{\mathrm{rt}}\left(i\right)}$, i.e., $\left(\overrightarrow{\boldsymbol{P}_{\mathrm{at}}\left(k\right)\boldsymbol{P}_{\mathrm{f}}}\right)^{\mathrm{T}} \cdot \left(\overrightarrow{\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)\boldsymbol{P}_{\mathrm{rt}}\left(i\right)}\right) = 0$. Substituting into Eq. (5), we have

$$h_i = -\frac{\left(\boldsymbol{P}_{\mathrm{rt}}(i-1)-\boldsymbol{P}_{\mathrm{at}}(k)\right)^{\mathrm{T}}\left(\boldsymbol{P}_{\mathrm{rt}}(i)-\boldsymbol{P}_{\mathrm{rt}}(i-1)\right)}{\left(\boldsymbol{P}_{\mathrm{rt}}(i)-\boldsymbol{P}_{\mathrm{rt}}(i-1)\right)^{\mathrm{T}}\left(\boldsymbol{P}_{\mathrm{rt}}(i)-\boldsymbol{P}_{\mathrm{rt}}(i-1)\right)} \tag{6}$$

During the search process, the value of h_i can be categorized into five cases, as illustrated in Fig. 7. When $0 \leq h_i \leq 1$, it indicates that $\boldsymbol{P}_{\mathrm{f}}$ lies between $\boldsymbol{P}_{\mathrm{rt}}\left(i-1\right), \boldsymbol{P}_{\mathrm{rt}}\left(i\right)$. If $h_i < 0$ occurs, the search is restarted forward to determine whether $\boldsymbol{P}_{\mathrm{f}}$ is located between $\boldsymbol{P}_{\mathrm{rt}}\left(i-2\right), \boldsymbol{P}_{\mathrm{rt}}\left(i-1\right)$, and this iteration continues until the situations shown in Fig. 7(a) or (d) are satisfied. Conversely, if $h_i > 1$, the search proceeds backward until the conditions shown in Fig. 7(a) or (e) are satisfied. The process can start from the current cycle command point and the previous cycle command point, $\boldsymbol{P}_{\mathrm{rt}}\left(k\right)$ and $\boldsymbol{P}_{\mathrm{rt}}\left(k-1\right)$.

After identifying the interval where $\boldsymbol{P}_{\mathrm{f}}$ lies, the value of $\boldsymbol{P}_{\mathrm{f}}$ can be determined by Eq. (5). The tool orientation vector $\boldsymbol{O}_{\mathrm{f}}$ corresponding to $\boldsymbol{P}_{\mathrm{f}}$ can be obtained through spherical linear interpolation of the tool orientation at the two interpolation points in the interval, as shown in Eq. (7)

$$\boldsymbol{O}_{\mathrm{f}} = \frac{\sin((1-h_i)\theta)}{\sin\theta}\boldsymbol{O}_{\mathrm{rt}}\left(i-1\right) + \frac{\sin(h_i\theta)}{\sin\theta}\boldsymbol{O}_{\mathrm{rt}}\left(i\right) \tag{7}$$

where θ is the angle between $\boldsymbol{O}_{\mathrm{rt}}\left(i-1\right)$ and $\boldsymbol{O}_{\mathrm{rt}}\left(i\right)$.

After obtaining $\boldsymbol{P}_{\mathrm{f}}$ and $\boldsymbol{O}_{\mathrm{f}}$, the TTP contour error and TORI contour error can be calculated by Eq. (8)

$$\begin{cases} \varepsilon_{\mathrm{p}} = \left\|\boldsymbol{P}_{\mathrm{at}} - \boldsymbol{P}_{\mathrm{f}}\right\| \\ \varepsilon_{\mathrm{o}} = arccos\frac{\boldsymbol{O}_{\mathrm{f}} \cdot \boldsymbol{O}_{\mathrm{at}}}{\left\|\boldsymbol{O}_{\mathrm{f}}\right\| \cdot \left\|\boldsymbol{O}_{\mathrm{at}}\right\|} \end{cases} \tag{8}$$

Contour error model predictive pre-compensation control

The principle of model predictive pre-compensation control is shown in Fig. 8. In the digital twin, the predictive controller reads the interpolation commands from the interpolator, optimizes axis commands via predictive control and then outputs the pre-compensated commands for the real-time controller.

As analyzed in the previous section, the tool contour error is not equivalent to the tool tracking error. It depends not only on the command point but also on the characteristics of the nearby curve. And calculating the contour error often requires information from several adjacent command points and iterative computations, making it impossible to express it as a linear combination of the position of each axis. However, since the tool command posture must lie on the ideal contour, the tracking error of the current point $\boldsymbol{P}_{\mathrm{at}}\left(k\right)$ relative to its corresponding command point $\boldsymbol{P}_{\mathrm{rt}}\left(k\right)$ serves as the upper bound of its contour error. Therefore, reducing the tool tracking error remains beneficial for minimizing the tool contour error. Additionally, the tool tracking error can be conveniently linearized to facilitate pre-compensation control.

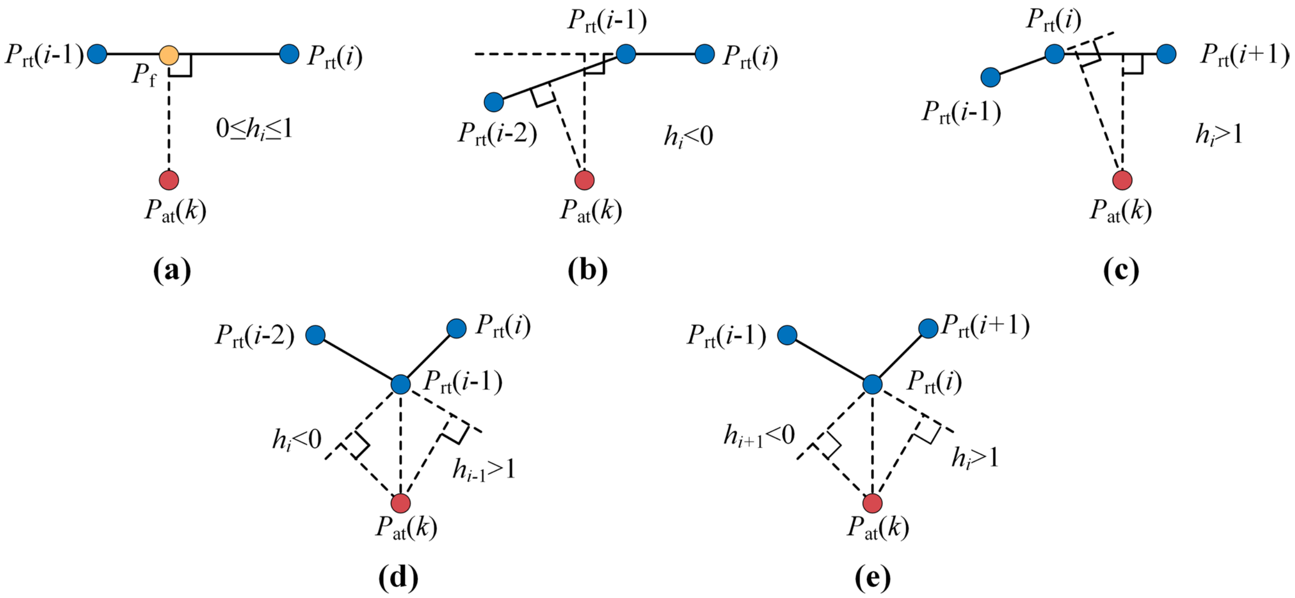
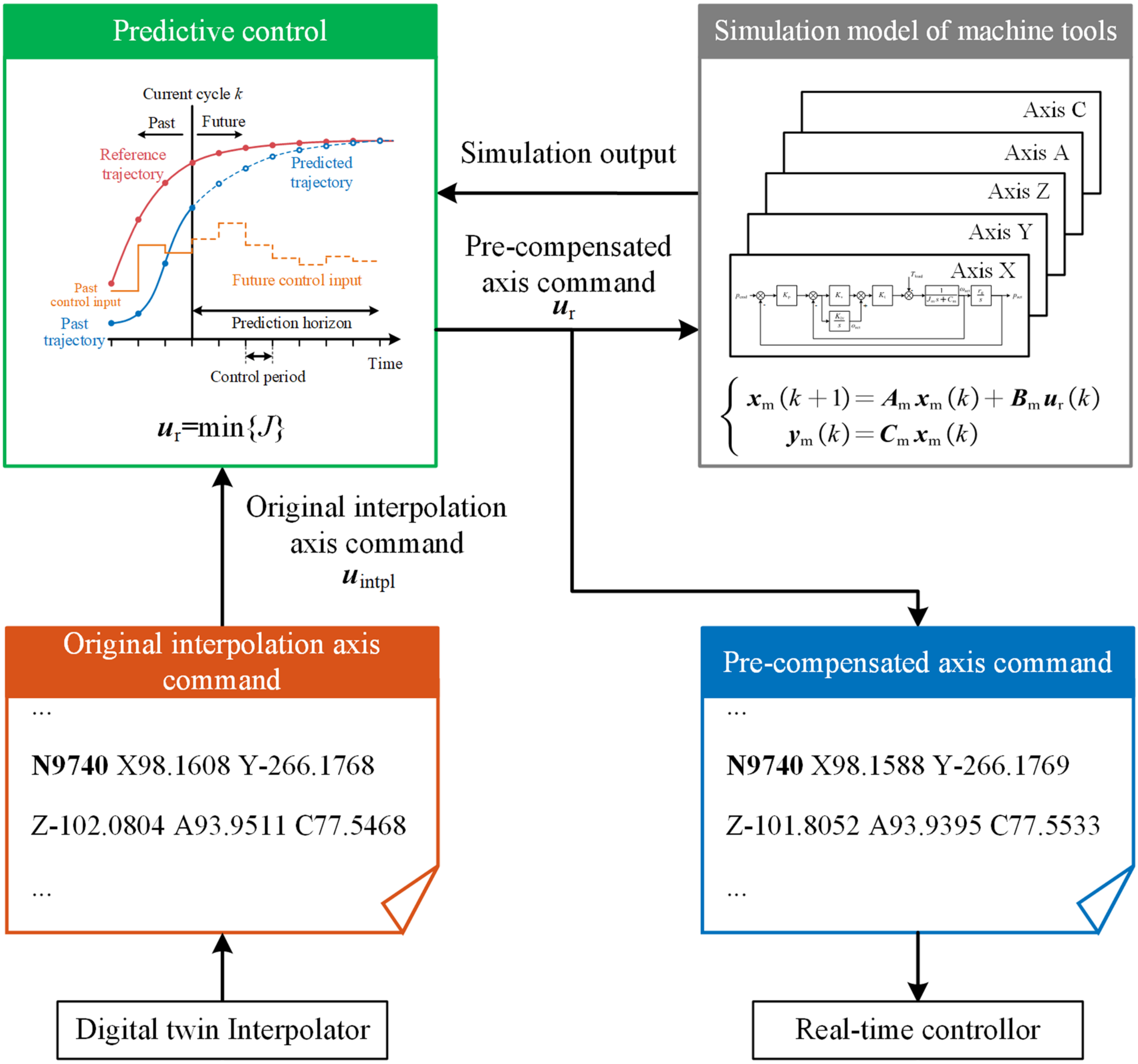


Fig. 7. Different situations when searching for the foot point.


Predictive control
Current cycle k
Past
Future
Reference trajectory
Predicted trajectory
Past control input
Future control input
Past trajectory
Prediction horizon
Control period
Time
u_r=min{J}
Simulation model of machine tools
Axis C
Axis A
Axis Z
Axis Y
Axis X
$\begin{cases} x_m(k+1) = A_m x_m(k) + B_m u_r(k) \\ y_m(k) = C_m x_m(k) \end{cases}$
Simulation output
Pre-compensated axis command u_r
Original interpolation axis command u_{intpl}
Original interpolation axis command
Pre-compensated axis command
Digital twin Interpolator
Real-time controllor

```
...
N9740 X98.1608 Y-266.1768
Z-102.0804 A93.9511 C77.5468
...
```

```
...
N9740 X98.1588 Y-266.1769
Z-101.8052 A93.9395 C77.5533
...
```

Fig. 8. Principle of model predictive pre-compensation control.

Assuming that the disturbance changes slowly compared to the sampling control cycle, the incremental form can be used to eliminate its influence. Define $\Delta \boldsymbol{u}_q(k) = \boldsymbol{u}_q(k) - \boldsymbol{u}_q(k-1)\,, \Delta \boldsymbol{x}_q(k) = \boldsymbol{x}_q(k) - \boldsymbol{x}_q(k-1)\,,\ \Delta \boldsymbol{y}_q(k) = \boldsymbol{y}_q(k) - \boldsymbol{y}_q(k-1)\,, (q = X, Y, Z, A, C)$. Then Eq. (2) can be written as Eq. (9):

$$\begin{cases} \Delta \boldsymbol{x}_q(k+1) = \boldsymbol{A}_q \Delta \boldsymbol{x}_q(k) + \boldsymbol{B}_q \Delta \boldsymbol{u}_q(k) \\ \Delta \boldsymbol{y}_q(k) = \boldsymbol{C}_q \Delta \boldsymbol{x}_q(k) \end{cases} \tag{9}$$

For a machine tool with five coordinate axes(X/Y/Z/A/C), let its state be $\boldsymbol{x}_{\mathrm{m}} = \left[\boldsymbol{x}_X^{\mathrm{T}}\ \boldsymbol{x}_Y^{\mathrm{T}}\ \boldsymbol{x}_Z^{\mathrm{T}}\ \boldsymbol{x}_A^{\mathrm{T}}\ \boldsymbol{x}_C^{\mathrm{T}}\right]^{\mathrm{T}}$, the output be the positions of each coordinate axis, i.e., $\boldsymbol{y}_{\mathrm{m}} = \left[\boldsymbol{y}_X^{\mathrm{T}}\ \boldsymbol{y}_Y^{\mathrm{T}}\ \boldsymbol{y}_Z^{\mathrm{T}}\ \boldsymbol{y}_A^{\mathrm{T}}\ \boldsymbol{y}_C^{\mathrm{T}}\right]^{\mathrm{T}}$, and the input be the command positions of each axis, i.e., $\boldsymbol{u}_{\mathrm{m}} = \left[\boldsymbol{u}_X^{\mathrm{T}}\ \boldsymbol{u}_Y^{\mathrm{T}}\ \boldsymbol{u}_Z^{\mathrm{T}}\ \boldsymbol{u}_A^{\mathrm{T}}\ \boldsymbol{u}_C^{\mathrm{T}}\right]^{\mathrm{T}}$. Then, expressing the machine state in incremental form, we have

$$\begin{cases} \Delta \boldsymbol{x}_{\mathrm{m}}(k+1) = \boldsymbol{A}_{\mathrm{m}} \Delta \boldsymbol{x}_{\mathrm{m}}(k) + \boldsymbol{B}_{\mathrm{m}} \Delta \boldsymbol{u}_{\mathrm{m}}(k) \\ \Delta \boldsymbol{y}_{\mathrm{m}}(k) = \boldsymbol{C}_{\mathrm{m}} \Delta \boldsymbol{x}_{\mathrm{m}}(k) \end{cases} \tag{10}$$

where

$$\boldsymbol{A}_{\mathrm{m}} = \mathrm{diag}\left(\boldsymbol{A}_X, \boldsymbol{A}_Y, \boldsymbol{A}_Z, \boldsymbol{A}_A, \boldsymbol{A}_C\right)$$
$$\boldsymbol{B}_{\mathrm{m}} = \mathrm{diag}\left(\boldsymbol{B}_X, \boldsymbol{B}_Y, \boldsymbol{B}_Z, \boldsymbol{B}_A, \boldsymbol{B}_C\right)$$
$$\boldsymbol{C}_{\mathrm{m}} = \mathrm{diag}\left(\boldsymbol{C}_X, \boldsymbol{C}_Y, \boldsymbol{C}_Z, \boldsymbol{C}_A, \boldsymbol{C}_C\right)$$

Since the rotational axes generally rotate at a slower speed, it is assumed that the angular increment within an interpolation cycle is very small. Under this assumption, the approximations $\sin\left(\theta + \varDelta\theta\right) \approx \sin\theta + \varDelta\theta\cos\theta$ and $\cos\left(\theta + \varDelta\theta\right) = \cos\theta - \varDelta\theta\sin\theta$ can be applied. Taking $p_{\mathrm{t}x}$ in Eq. (3) as an example, we obtain

$$\begin{aligned}\varDelta p_{\mathrm{t}x}(k) \approx p_{\mathrm{t}x}(k) - p_{\mathrm{t}x}(k-1) &= \varDelta p_{\mathrm{m}x} + L\left(\sin\left(p_{\mathrm{m}a} + \varDelta p_{\mathrm{m}a}\right)\sin\left(p_{\mathrm{m}c} + \varDelta p_{\mathrm{m}c}\right) - \sin p_{\mathrm{m}a}\sin p_{\mathrm{m}c}\right) \\ &= \varDelta p_{\mathrm{m}x} + L\left(\varDelta p_{\mathrm{m}a}\cos p_{\mathrm{m}a}\sin p_{\mathrm{m}c} + \varDelta p_{\mathrm{m}c}\sin p_{\mathrm{m}a}\cos p_{\mathrm{m}c} + \varDelta p_{\mathrm{m}a}\varDelta p_{\mathrm{m}c}\cos p_{\mathrm{m}a}\cos p_{\mathrm{m}c}\right)\end{aligned} \quad \# \qquad (11)$$

Since the angular increment is very small, $\varDelta p_{\mathrm{m}a}\varDelta p_{\mathrm{m}c}$ can be regarded as higher-order infinitesimal quantities relative to the angular increment. Therefore, Eq. (11) can be approximated as Eq. (12).

$$\varDelta p_{\mathrm{t}x}(k) \approx \varDelta p_{\mathrm{m}x} + L\left(\varDelta p_{\mathrm{m}a}\cos p_{\mathrm{m}a}\sin p_{\mathrm{m}c} + \varDelta p_{\mathrm{m}c}\sin p_{\mathrm{m}a}\cos p_{\mathrm{m}c}\right) \qquad (12)$$

Equation (3) can be approximated and expressed in incremental form as shown in Eq. (13).

$$\begin{cases} \varDelta p_{\mathrm{t}x} = \varDelta p_{\mathrm{m}x} + L\left(\varDelta p_{\mathrm{m}a}\cos p_{\mathrm{m}a}\sin p_{\mathrm{m}c} + \varDelta p_{\mathrm{m}c}\sin p_{\mathrm{m}a}\cos p_{\mathrm{m}c}\right) \\ \varDelta p_{\mathrm{t}y} = \varDelta p_{\mathrm{m}x} - L\left(\varDelta p_{\mathrm{m}a}\cos p_{\mathrm{m}a}\cos p_{\mathrm{m}c} - \varDelta p_{\mathrm{m}c}\sin p_{\mathrm{m}a}\sin p_{\mathrm{m}c}\right) \\ \varDelta p_{\mathrm{t}z} = \varDelta p_{\mathrm{m}z} - L\varDelta p_{\mathrm{m}a} sinp_{\mathrm{m}a} \\ \varDelta O_{\mathrm{t}i} = -\varDelta p_{\mathrm{m}a} cosp_{\mathrm{m}a} sinp_{\mathrm{m}c} - \varDelta p_{\mathrm{m}c} sinp_{\mathrm{m}a} cosp_{\mathrm{m}c} \\ \varDelta O_{\mathrm{t}j} = \varDelta p_{\mathrm{m}a} cosp_{\mathrm{m}a} cosp_{\mathrm{m}c} - \varDelta p_{\mathrm{m}c} sinp_{\mathrm{m}a} sinp_{\mathrm{m}c} \\ \varDelta O_{\mathrm{t}k} = \varDelta p_{\mathrm{m}a} sinp_{\mathrm{m}a} \end{cases} \qquad (13)$$

Equation (13) can be written as Eq. (14):

$$\varDelta \boldsymbol{y}_{\mathrm{t}} = \begin{bmatrix} \varDelta p_{\mathrm{t}x} \\ \varDelta p_{\mathrm{t}y} \\ \varDelta p_{\mathrm{t}z} \\ \varDelta O_{\mathrm{t}i} \\ \varDelta O_{\mathrm{t}j} \\ \varDelta O_{\mathrm{t}k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_{3\times 3} & -L\boldsymbol{T}_{\mathrm{R}} \\ 0_{3\times 3} & \boldsymbol{T}_{\mathrm{R}} \end{bmatrix} \begin{bmatrix} \varDelta p_{\mathrm{m}x} \\ \varDelta p_{\mathrm{m}y} \\ \varDelta p_{\mathrm{m}z} \\ \varDelta p_{\mathrm{m}a} \\ \varDelta p_{\mathrm{m}c} \end{bmatrix} = \boldsymbol{C}_{\mathrm{t}}\varDelta \boldsymbol{x}_{\mathrm{m}} \qquad (14)$$

where

$$\boldsymbol{T}_{\mathrm{R}} = \begin{bmatrix} -\cos p_{\mathrm{m}a}\sin p_{\mathrm{m}c} & -\sin p_{\mathrm{m}a}\cos p_{\mathrm{m}c} \\ \cos p_{\mathrm{m}a}\cos p_{\mathrm{m}c} & -\sin p_{\mathrm{m}a}\sin p_{\mathrm{m}c} \\ \sin p_{\mathrm{m}a} & 0 \end{bmatrix}$$

Then the incremental form of the system can be written as Eq. (15).

$$\begin{cases} \varDelta \boldsymbol{x}_{\mathrm{m}}(k+1) = \boldsymbol{A}_{\mathrm{m}}\varDelta \boldsymbol{x}_{\mathrm{m}}(k) + \boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{\mathrm{m}}(k) \\ \varDelta \boldsymbol{y}_{\mathrm{m}}(k) = \boldsymbol{C}_{\mathrm{m}}\varDelta \boldsymbol{x}_{\mathrm{m}}(k) \\ \varDelta \boldsymbol{y}_{\mathrm{t}}(k) = \boldsymbol{C}_{\mathrm{t}}\varDelta \boldsymbol{x}_{\mathrm{m}}(k) \end{cases} \qquad (15)$$

This paper employs MPC method to implement pre-compensation control. By looking ahead N steps and using the incremental form, the system state can be expressed as Eq. (16).

$$\begin{cases} \varDelta \boldsymbol{x}_{\mathrm{m}}(k+1) = \boldsymbol{A}_{\mathrm{m}}\varDelta \boldsymbol{x}_{\mathrm{m}}(k) + \boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{m}(k) \\ \varDelta \boldsymbol{x}_{\mathrm{m}}(k+2) = \boldsymbol{A}_{\mathrm{m}}^2\varDelta \boldsymbol{x}_{\mathrm{m}}(k) + \boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{\mathrm{m}}(k) + \boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{\mathrm{m}}(k+1) \\ \vdots \\ \varDelta \boldsymbol{x}_{\mathrm{m}}(k+N) = \boldsymbol{A}_{\mathrm{m}}^N\varDelta \boldsymbol{x}_{\mathrm{m}}(k) + \boldsymbol{A}_{\mathrm{m}}^{N-1}\boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{\mathrm{m}}(k) + \cdots \\ +\boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{\mathrm{m}}(k+N-2) + \boldsymbol{B}_{\mathrm{m}}\varDelta \boldsymbol{u}_{\mathrm{m}}(k+N-1) \end{cases} \qquad (16)$$

Let the N-step ahead prediction state be defined as $\boldsymbol{X}_{\mathrm{m}}(k+N) = \begin{bmatrix} \boldsymbol{x}_{\mathrm{m}}^{\mathrm{T}}(k+1) & \boldsymbol{x}_{\mathrm{m}}^{\mathrm{T}}(k+2) & \cdots & \boldsymbol{x}_{\mathrm{m}}^{\mathrm{T}}(k+M) \end{bmatrix}^{\mathrm{T}}$, and the N-step ahead control command as $\boldsymbol{U}_{\mathrm{m}}(k+N) = \begin{bmatrix} \boldsymbol{u}_{\mathrm{m}}^{\mathrm{T}}(k+1) & \boldsymbol{u}_{\mathrm{m}}^{\mathrm{T}}(k+2) & \cdots & \boldsymbol{u}_{\mathrm{m}}^{\mathrm{T}}(k+M) \end{bmatrix}^{\mathrm{T}}$. Then we have

$$\varDelta \boldsymbol{X}_{\mathrm{m}}(k+N) = \boldsymbol{A}_{\mathrm{fx}}\varDelta \boldsymbol{x}_{\mathrm{m}}(k) + \boldsymbol{B}_{\mathrm{fx}}\varDelta \boldsymbol{U}_{\mathrm{m}}(k+N) \qquad (17)$$

where

$$\boldsymbol{A}_{\mathrm{fx}} = \begin{bmatrix} \boldsymbol{A}_{\mathrm{m}} \\ \boldsymbol{A}_{\mathrm{m}}^2 \\ \vdots \\ \boldsymbol{A}_{\mathrm{m}}^N \end{bmatrix}, \boldsymbol{B}_{\mathrm{fx}} = \begin{bmatrix} \boldsymbol{B}_{\mathrm{m}} & 0 & \cdots & 0 \\ \boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} + \boldsymbol{B}_{\mathrm{m}} & \boldsymbol{B}_{\mathrm{m}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=0}^{N-1}\boldsymbol{A}_{\mathrm{m}}^i\boldsymbol{B}_{\mathrm{m}} & \sum_{i=0}^{N-2}\boldsymbol{A}_{\mathrm{m}}^i\boldsymbol{B}_{\mathrm{m}} & \cdots & \boldsymbol{B}_{\mathrm{m}} \end{bmatrix}$$

The *N*-step ahead output incremental state can be described as Eq. (18):

$$\left\{ \begin{array}{c} \varDelta \boldsymbol{Y}_{\mathrm{m}}\left(k+N\right)=\boldsymbol{A}_{\mathrm{fy}}\varDelta \boldsymbol{x}_{\mathrm{m}}\left(k\right)+\boldsymbol{B}_{\mathrm{fy}}\varDelta \boldsymbol{U}_{\mathrm{m}}\left(k+N\right) \\ \varDelta \boldsymbol{Y}_{\mathrm{t}}\left(k+N\right)=\boldsymbol{A}_{\mathrm{ft}}\varDelta \boldsymbol{x}_{\mathrm{m}}\left(k\right)+\boldsymbol{B}_{\mathrm{ft}}\varDelta \boldsymbol{U}_{\mathrm{m}}\left(k+N\right) \end{array} \right. \tag{18}$$

where

$$\boldsymbol{A}_{\mathrm{fy}}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}} \\ \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{2} \\ \vdots \\ \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{N} \end{bmatrix}, \boldsymbol{B}_{\mathrm{fx}}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} & 0 & \cdots & 0 \\ \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}}+\boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} & \boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=0}^{N-1}\boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \sum_{i=0}^{N-2}\boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \cdots & \boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} \end{bmatrix}$$

$$\boldsymbol{A}_{\mathrm{ft}}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}} \\ \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{2} \\ \vdots \\ \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{N} \end{bmatrix}, \boldsymbol{B}_{\mathrm{ft}}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} & 0 & \cdots & 0 \\ \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}}+\boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} & \boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=0}^{N-1}\boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \sum_{i=0}^{N-2}\boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \cdots & \boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} \end{bmatrix}$$

The *N*-step ahead output states are:

$$\left\{ \begin{array}{c} \boldsymbol{y}_{\mathrm{m}}\left(k+j\right)=\boldsymbol{y}_{\mathrm{m}}\left(k\right)+\sum_{i=1}^{j}\varDelta \boldsymbol{y}_{\mathrm{m}}\left(k+i\right) \\ \boldsymbol{y}_{\mathrm{t}}\left(k+j\right)=\boldsymbol{y}_{\mathrm{t}}\left(k\right)+\sum_{i=1}^{j}\varDelta \boldsymbol{y}_{\mathrm{t}}\left(k+i\right) \end{array} \right. \tag{19}$$

Therefore, we have

$$\left\{ \begin{array}{c} \boldsymbol{Y}_{\mathrm{m}}\left(k+N\right)=\boldsymbol{H}_{\mathrm{m}}\boldsymbol{y}_{\mathrm{m}}\left(k\right)+\boldsymbol{A}_{\mathrm{Fy}}\varDelta \boldsymbol{x}_{\mathrm{m}}\left(k\right)+\boldsymbol{B}_{\mathrm{Fy}}\varDelta \boldsymbol{U}_{\mathrm{m}}\left(k+N\right) \\ \boldsymbol{Y}_{\mathrm{t}}\left(k+N\right)=\boldsymbol{H}_{\mathrm{t}}\boldsymbol{y}_{\mathrm{t}}\left(k\right)+\boldsymbol{A}_{\mathrm{Ft}}\varDelta \boldsymbol{x}_{\mathrm{m}}\left(k\right)+\boldsymbol{B}_{\mathrm{Ft}}\varDelta \boldsymbol{U}_{\mathrm{m}}\left(k+N\right) \end{array} \right. \tag{20}$$

where

$$\{\boldsymbol{H}_{\mathrm{m}}\}_{5N\times 5N}=\mathrm{diag}\left(\boldsymbol{I}_{5\times 5},\boldsymbol{I}_{5\times 5},\cdots,\boldsymbol{I}_{5\times 5}\right),\{\boldsymbol{H}_{\mathrm{t}}\}_{6N\times 6N}=\mathrm{diag}\left(\boldsymbol{I}_{6\times 6},\boldsymbol{I}_{6\times 6},\cdots,\boldsymbol{I}_{6\times 6}\right)$$

$$\boldsymbol{A}_{\mathrm{F}y}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}} \\ \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{2}+\boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}} \\ \vdots \\ \sum_{i=1}^{N}\boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{i} \end{bmatrix}, \boldsymbol{B}_{\mathrm{F}y}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} & 0 & \cdots & 0 \\ \boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}}+\boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} & \boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{N-1}\boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \sum_{i=1}^{N-2}\boldsymbol{C}_{\mathrm{m}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \cdots & \boldsymbol{C}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}} \end{bmatrix}$$

$$\boldsymbol{A}_{\mathrm{F}t}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}} \\ \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{2}+\boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}} \\ \vdots \\ \sum_{i=1}^{N}\boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{i} \end{bmatrix}, \boldsymbol{B}_{\mathrm{F}t}=\begin{bmatrix} \boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} & 0 & \cdots & 0 \\ \boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}\boldsymbol{B}_{\mathrm{m}}+\boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} & \boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{N-1}\boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \sum_{i=1}^{N-2}\boldsymbol{C}_{\mathrm{t}}\boldsymbol{A}_{\mathrm{m}}^{i}\boldsymbol{B}_{\mathrm{m}} & \cdots & \boldsymbol{C}_{\mathrm{t}}\boldsymbol{B}_{\mathrm{m}} \end{bmatrix}$$

Define pre-compensation increment $\boldsymbol{v}(k)_{5N\times 1}$, the final reference command is defined as $\boldsymbol{u}_{\mathrm{r}}\left(k\right)=\boldsymbol{u}_{\mathrm{m}}\left(k\right)+\boldsymbol{v}\left(k\right)$. Equation (20) can be rewritten as:

$$\left\{ \begin{array}{c} \boldsymbol{Y}_{\mathrm{m}}\left(k+N\right)=\boldsymbol{H}_{\mathrm{m}}\boldsymbol{y}_{\mathrm{m}}\left(k\right)+\boldsymbol{A}_{\mathrm{Fy}}\varDelta \boldsymbol{x}_{\mathrm{m}}\left(k\right)+\boldsymbol{B}_{\mathrm{Fy}}\varDelta \boldsymbol{U}_{\mathrm{m}}\left(k+N\right)+\boldsymbol{B}_{\mathrm{Fy}}V\left(k+N\right) \\ \boldsymbol{Y}_{\mathrm{t}}\left(k+N\right)=\boldsymbol{H}_{\mathrm{t}}\boldsymbol{y}_{\mathrm{t}}\left(k\right)+\boldsymbol{A}_{\mathrm{Ft}}\varDelta \boldsymbol{x}_{\mathrm{m}}\left(k\right)+\boldsymbol{B}_{\mathrm{Ft}}\varDelta \boldsymbol{U}_{\mathrm{m}}\left(k+N\right)+\boldsymbol{B}_{\mathrm{Fy}}V\left(k+N\right) \end{array} \right. \tag{21}$$

where $\boldsymbol{V}\left(k+N\right)=\left[\begin{array}{cccc} \boldsymbol{v}^{\mathrm{T}}\left(k\right) & \boldsymbol{v}^{\mathrm{T}}(k+1) & \cdots & \boldsymbol{v}^{\mathrm{T}}(k+N) \end{array}\right]^{T}$

Let the desired positions of each axis and the desired tool posture be denoted as $\boldsymbol{Y}_{\mathrm{mr}}\left(k+N\right),\boldsymbol{Y}_{\mathrm{mt}}\left(k+N\right)$, respectively. The cost function is taken as

$$\begin{aligned} J=&\underbrace{\left[Y_{\mathrm{mr}}\left(k+M\right)-Y_{\mathrm{m}}\left(k+M\right)\right]^{\mathrm{T}}Q_{\mathrm{m}}\left[Y_{\mathrm{mr}}\left(k+M\right)-Y_{\mathrm{m}}\left(k+M\right)\right]}_{\text{axis tracking error}} \\ &+\underbrace{\left[Y_{\mathrm{tr}}\left(k+M\right)-Y_{\mathrm{t}}\left(k+M\right)\right]^{\mathrm{T}}Q_{\mathrm{t}}\left[Y_{\mathrm{tr}}\left(k+M\right)-Y_{\mathrm{t}}\left(k+M\right)\right]}_{\text{tool tracking error}} \\ &+\underbrace{\Delta V^{\mathrm{T}}\left(k\right)Q_{\mathrm{u}}\Delta V\left(k\right)}_{\text{pre}-\text{compensation increment}} \end{aligned} \tag{22}$$

where $\{\boldsymbol{Q}_{\mathrm{m}}\}_{5N\times 5N},\{\boldsymbol{Q}_{\mathrm{t}}\}_{6N\times 6N},\{\boldsymbol{Q}_{\mathrm{u}}\}_{5N\times 5N}$ are weight matrices. The diagonal elements of these matrices represent the weights assigned to each predicted quantity, all of which are real numbers greater than 0, while the off-diagonal elements are set to 0.

Equation (22) indicates that the cost function incorporates the tracking errors of each axis, the tool tracking errors, and the pre-compensation increment. The next step involves determining its minimum value.

Let

$$\boldsymbol{E}_m\left(k\right)=\boldsymbol{Y}_{mr}\left(k+M\right)-\boldsymbol{H}_m\boldsymbol{y}_m\left(k\right)-\boldsymbol{A}_{\mathrm{Fy}}\varDelta\,\boldsymbol{x}_m\left(k\right)-\boldsymbol{B}_{\mathrm{Fy}}\varDelta\,\boldsymbol{U}_{\mathrm{m}}\left(k+N\right)$$
$$\boldsymbol{E}_t\left(k\right)=\boldsymbol{Y}_{tr}\left(k+M\right)-\boldsymbol{H}_t\boldsymbol{y}_t\left(k\right)-\boldsymbol{A}_{\mathrm{Ft}}\varDelta\,\boldsymbol{x}_m\left(k\right)-\boldsymbol{B}_{\mathrm{Ft}}\varDelta\,\boldsymbol{U}_{\mathrm{m}}(k+N)$$

Taking the derivative of Eq. (22), we have

$$\frac{\partial\,J}{\partial\,\boldsymbol{V}(k)}=-2\boldsymbol{B}_y^{\mathrm{T}}\boldsymbol{Q}_m\left[\boldsymbol{E}_m-\boldsymbol{B}_y\varDelta\,\boldsymbol{U}\left(k\right)\right]-2\boldsymbol{B}_{yt}^{\mathrm{T}}\boldsymbol{Q}_t\left[\boldsymbol{E}_t\left(k\right)-\boldsymbol{B}_{yt}\varDelta\,\boldsymbol{U}\left(k\right)\right]+2\boldsymbol{Q}_uV\left(k\right) \tag{23}$$

Let $\partial\,J/\partial\,\boldsymbol{V}=0$, we have

$$V\left(k\right)=\left(\boldsymbol{B}_y^{\mathrm{T}}\boldsymbol{Q}_m\boldsymbol{B}_y+\boldsymbol{B}_{yt}^{\mathrm{T}}\boldsymbol{Q}_t\boldsymbol{B}_{yt}+\boldsymbol{Q}_u\right)^{-1}\left[\boldsymbol{B}_y^{\mathrm{T}}\boldsymbol{Q}_m\boldsymbol{E}_m+\boldsymbol{B}_{yt}^{\mathrm{T}}\boldsymbol{Q}_t\boldsymbol{E}_t\left(k\right)\right] \tag{24}$$

In each cycle, only the first set of commands in $\boldsymbol{V}\left(k\right)$ is taken as the output for the current cycle. Therefore, the final reference command for this cycle is

$$\boldsymbol{u}_{\mathrm{r}}\left(k\right)=\boldsymbol{u}_{\mathrm{m}}\left(k-1\right)+\varDelta\,\boldsymbol{u}_{\mathrm{m}}\left(k\right)+v\left(k\right) \tag{25}$$

where

$$\boldsymbol{v}\left(k\right)=\left[\begin{array}{cc}\boldsymbol{I}_{5\times\,5} & 0_{5\times\,5(N-1)}\end{array}\right]_{5\times\,5N}\boldsymbol{V}\left(k\right)$$

Contour error online feedback control

After obtaining the pre-compensation command, this paper further designs an online contour error feedback controller to achieve more precise control of contour error. In the real-time controller, the foot point can be located using the contour error calculation method described earlier. Once the tool position and orientation at the foot point, $\boldsymbol{P}_{\mathrm{f}}$ and $\boldsymbol{O}_{\mathrm{f}}$, are obtained, the corresponding positions of each axis can be determined through inverse kinematics using Eq. (4). The difference between these positions and the current axis coordinates yields the components of the contour error projected onto each servo axis, denoted as $\boldsymbol{\epsilon}_{\mathrm{m}}=\left[\begin{array}{ccccc}\epsilon_x & \epsilon_y & \epsilon_z & \epsilon_a & \epsilon_c\end{array}\right]^{\mathrm{T}}$.

This component is utilized as the compensation amount $\varDelta\,u_{\mathrm{e}}$. To prevent system oscillation, a gain coefficient K_{ce} is introduced, and a limiter is applied to ensure system stability. The final reference control increment is:

$$\varDelta\,u_{\mathrm{r}q}\left(k\right)=\left\{\begin{array}{cc}-\varDelta\,u_{\mathrm{max}q}, & K_{\mathrm{ce}}\varDelta\,u_{\mathrm{e}q}\left(k\right)<-\varDelta\,u_{\mathrm{max}q}\\ K_{\mathrm{ce}}\varDelta\,u_{\mathrm{e}q}\left(k\right), & -\varDelta\,u_{\mathrm{max}q}\leq\,K_{\mathrm{ce}}\varDelta\,u_{\mathrm{e}q}\left(k\right)\leq\,\varDelta\,u_{\mathrm{max}q}\\ \varDelta\,u_{\mathrm{max}q}, & K_{\mathrm{ce}}\varDelta\,u_{\mathrm{e}q}\left(k\right)>-\varDelta\,u_{\mathrm{max}q}\end{array}\right.,q=X,Y,Z,A,C \tag{26}$$

The principle of online feedback control is shown in Fig. 9:

Experiment

To validate the method proposed in this paper, an experimental system was constructed, as illustrated in Fig. 10. The system includes a five-axis motion platform, a real-time controller, and a digital twin. The control system of this platform is developed by TwinCAT 3 and runs on a Windows 7 industrial computer with an Intel® Core™ i7-6700 CPU @ 3.40 GHz. And the control cycle sets to 2ms.The digital twin platform runs on a workstation with Windows 11, powered by an Intel® Core™ i9-12900 CPU @ 2.40 GHz.

The model parameters of the five-axis motion control platform are shown in Table 1:

This paper adopts an S-shaped curve as the input curve as depicted in Fig. 11. And the curve information is shown in Appendix A, Table A1.The interpolator and the velocity, acceleration, and jerk constraints for each axis are configured according to the settings described in reference[43]. The feedrate is set to 50 mm/s. The interpolation results are illustrated in Figs. 12 and 13.

As indicated in reference[44], the MPCC prediction control cycle should exceed the system time constant to achieve optimal control performance. Therefore, in the offline MPCC stage, the prediction step length is set to $N=10$, $\boldsymbol{Q}_{\mathrm{m}}=5\boldsymbol{I}_{50\times\,50},\boldsymbol{Q}_{\mathrm{t}}=5\boldsymbol{I}_{60\times\,60},\boldsymbol{Q}_{\mathrm{u}}=\boldsymbol{I}_{60\times\,60}$. For online control $\varDelta\,u_{\mathrm{max}q}$ is set to 0.02 mm (or rad). The linear encoder resolution of X, Y and Z are both 0.1 μm while the rotator encoder resolution of A and C are 2^{20} pulses/rev and 2^{17} pulses/rev. The sampling period is 0.002 s.

To verify the effectiveness of the control methods and the impact of the online feedback control gain K_{ce} on the control results, the following groups of experiments are compared: (1) no compensation is performed (case 1); (2) only MPCC (case 2); (3) MPCC and online feedback control with $K_{\mathrm{ce}}=0.1$(case 3) ; (4) MPCC and online feedback control with $K_{\mathrm{ce}}=0.2$(case 4); (5) MPCC and online feedback control with $K_{\mathrm{ce}}=0.25$(case 5); (6) MPCC and online feedback control with $K_{\mathrm{ce}}=0.3$(case 6). The final tool positions and orientations of those cases are shown in Fig. 14.

The tool tracking error is presented in Table 2; Fig. 15. Here, the TTP tracking error and TORI tracking error are defined according to Eq. (8), with the foot point replaced by the command of the current cycle.

It can be observed that after pre-compensation, the tool tracking error of case 2 is significantly reduced than case 1, which achieves the goal of pre-compensation. However, since online feedback control does not account for tool tracking errors, no further improvement in tool tracking error is observed in case 3 – case 6. In terms of average TORI tracking error, case 3 – case 6 have a slight increment compared to case 2. This situation is caused by the different objectives of contour error control and tracking error control. If the dynamic performance of a certain axis is poor, tracking error control will not adjust the response of other axes. However, contour error

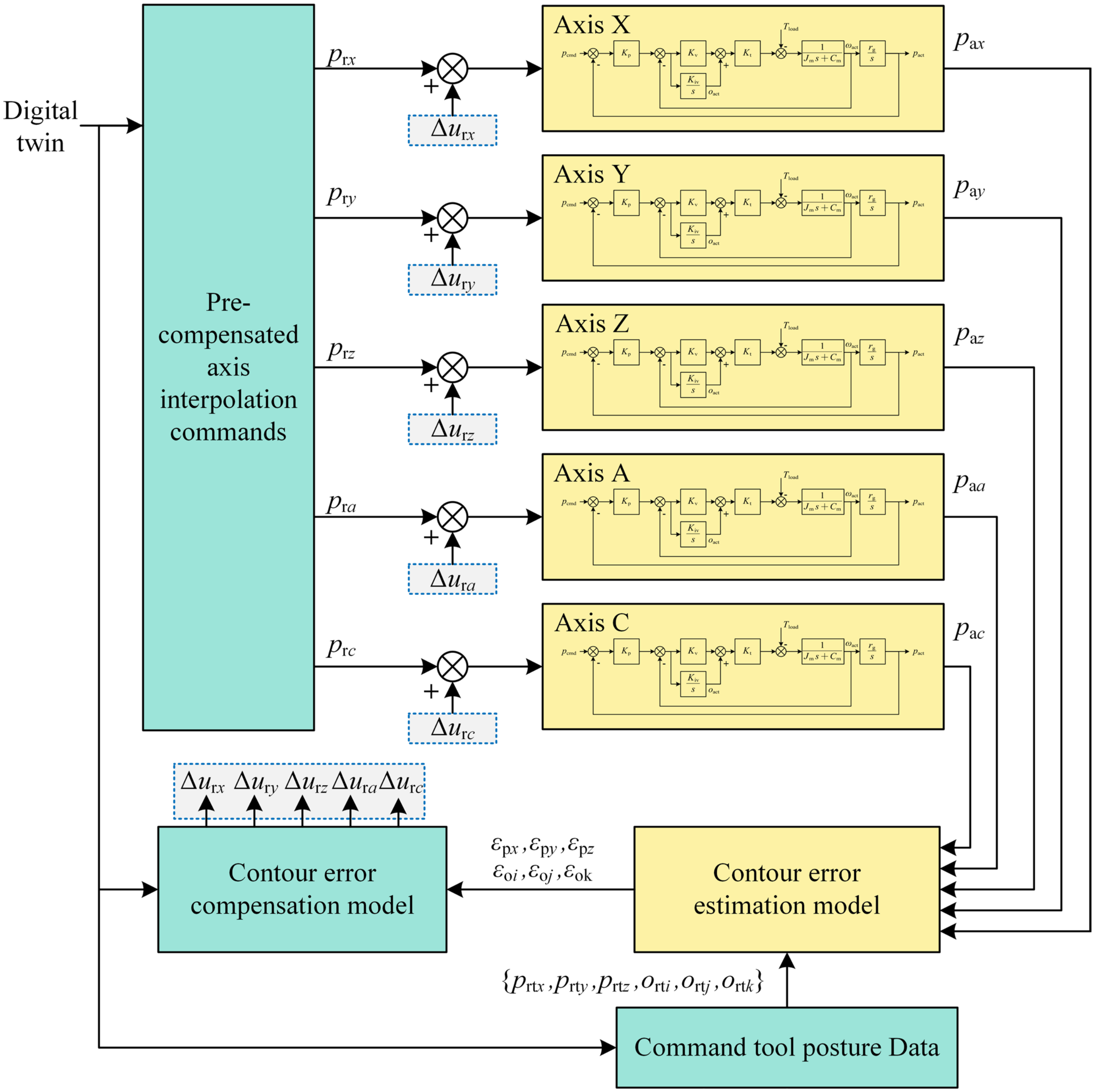

Digital twin
Pre-compensated axis interpolation commands
p_{rx}
p_{ry}
p_{rz}
p_{ra}
p_{rc}
+
Δu_{rx}
Δu_{ry}
Δu_{rz}
Δu_{ra}
Δu_{rc}
Axis X
Axis Y
Axis Z
Axis A
Axis C
p_{cmd}
K_p
K_v
K_t
T_{load}
$\frac{1}{J_m s + C_m}$
ω_{act}
$\frac{r_g}{s}$
p_{act}
$\frac{K_{iv}}{s}$
o_{act}
p_{ax}
p_{ay}
p_{az}
p_{aa}
p_{ac}
Δu_{rx} Δu_{ry} Δu_{rz} Δu_{ra} Δu_{rc}
Contour error compensation model
$\varepsilon_{px}, \varepsilon_{py}, \varepsilon_{pz}$
$\varepsilon_{oi}, \varepsilon_{oj}, \varepsilon_{ok}$
Contour error estimation model
$\{p_{rtx}, p_{rty}, p_{rtz}, o_{rti}, o_{rtj}, o_{rtk}\}$
Command tool posture Data

Fig. 9. Schematic diagram of the contour error online feedback control.

control will adjust the response of other axes to minimize the contour error, such as reducing the speed of other axes, which may result in a larger tracking error. Considering that tool contour error is a more effective indicator of machining quality, the slight increase in the average tool orientation tracking error is considered acceptable.

The contour error is illustrated in Fig. 16; Table 3. Compared to case 1, pre-compensation significantly reduces TTP contour error and TORI contour error. Specifically, the maximum TTP error is reduced by 72.71%, and its average value is reduced by 80.00%. Similarly, the maximum TORI error is reduced by 74.95%, and its average value is reduced by 83.43%.

Compared to the results of case 2, online control can further reduce the contour error. As K_{ce} increases, the contour error control effect improves. However, when K_{ce} is set to 0.3, the system oscillates. Therefore, in this experiment, K_{ce} = 0.25 (case 5) is more reasonable. The maximum TTP error of case 5 is reduced by 18.58% compared to case 2, and its average value is reduced by 19.48%. The maximum TORI error is reduced by 16.37%, and its average value is reduced by 19.47%. Using this combined method, compared to the uncompensated scenario, the maximum TTP error is reduced by 77.78%, its average value is reduced by 83.90%, the maximum TORI error is reduced by 79.05%, and its average value is reduced by 86.66%. These results demonstrate that the method proposed in this paper effectively reduces contour error. Meanwhile, the iterations required to calculate the contour error are reduced after MPCC, as shown in Table 4. It indicates that pre-compensation control

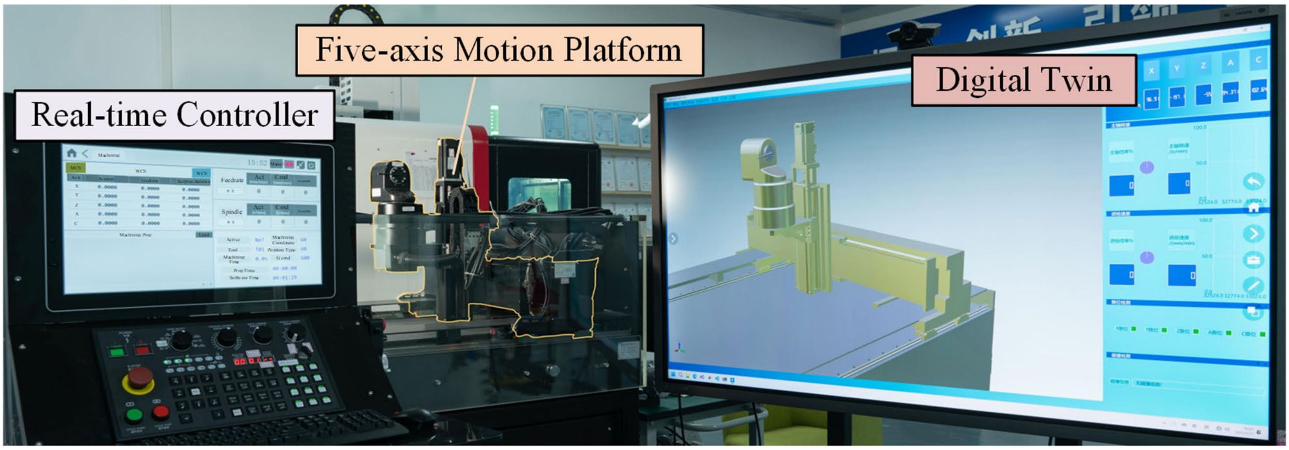


Fig. 10. Experimental equipment.

	X	Y	Z	A	C
$K_{\mathrm{p}}\ (1/\mathrm{s}, \mathrm{rad}/\ (\mathrm{s}\cdot\ \mathrm{m}))$	30.9370	131.985	13,801	47.8374	36.9471
$K_{\mathrm{v}}\ (\mathrm{As/m}, \mathrm{As/rad})$	39.8174	59.8906	0.4083	0.0180	3.3242
$K_{\mathrm{iv}}\ (\mathrm{A/m}, \mathrm{A/rad})$	6910.4	2506.1	1.2443	0.2249	84.5827
$K_{\mathrm{t}}\ (\mathrm{N/A}, \mathrm{N}\cdot\ \mathrm{m/A})$	48.6	48.6	0.508	0.357	0.386
$r_{\mathrm{g}}\ (\mathrm{m}, 1)$	1	1	0.01	1	1
$J_{\mathrm{m}}\ \left(\mathrm{kg}, \mathrm{kg}\cdot\ \mathrm{m}^2\right)$	75.382	29.754	7.8262×10^{-4}	3.1742×10^{-4}	0.022757
$C_{\mathrm{m}}\ (\mathrm{N}\cdot\ \mathrm{s/m})$	70.6892	11.2025	0.0053	3.0655×10^{-4}	0.2574
$f_{\mathrm{c}}\ (\mathrm{N}, \mathrm{N}\cdot\ \mathrm{m})$	59.6729	22.6970	0.2372	1.2790×10^{-4}	0.0903

Table 1. Parameters of the five-axis motion platform.

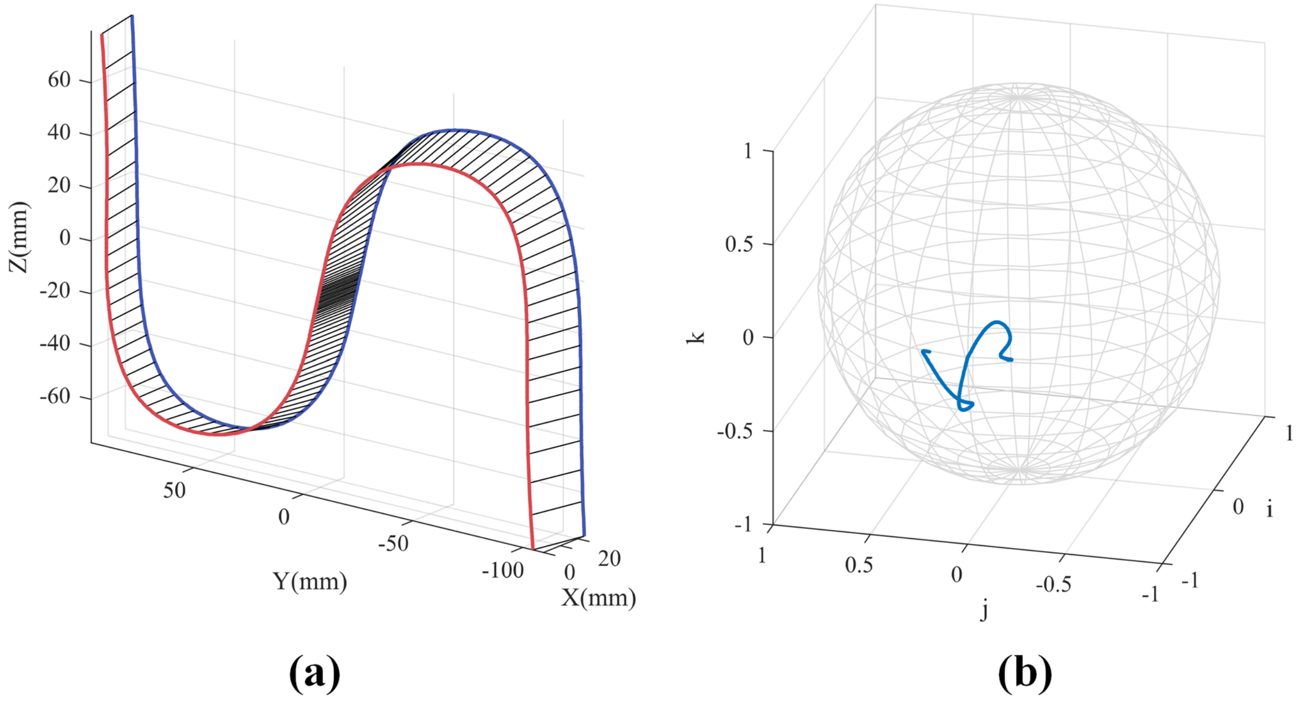


Fig. 11. S-shaped curve: (**a**) Tool tip position trajectory; (**b**) Tool orientation trajectory.

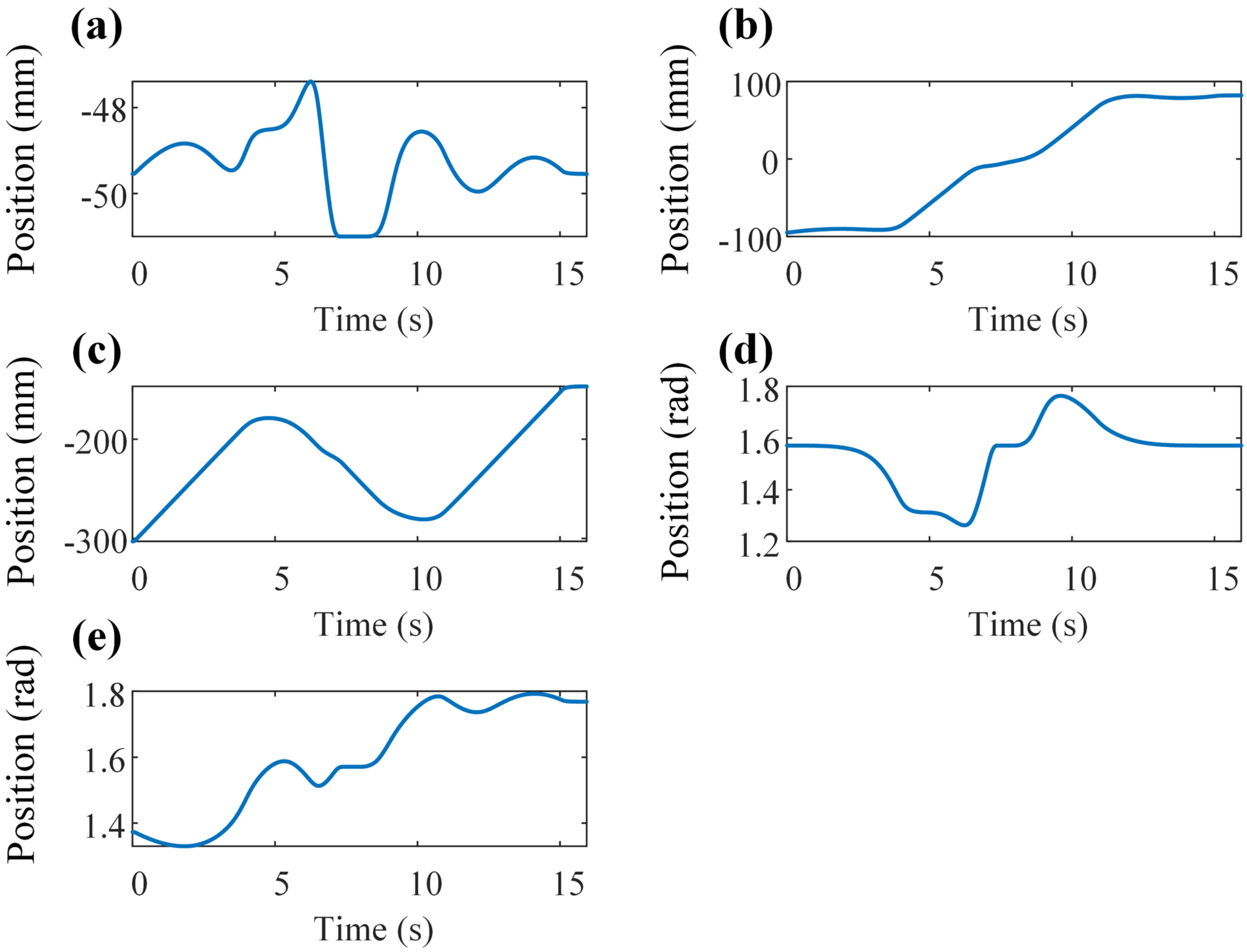


Fig. 12. Digital twin interpolation positions: (**a**) X-axis; (**b**) Y-axis; (**c**) Z-axis; (**d**) A-axis; (**e**) C-axis.

can provide a better basis for online control, effectively reducing the computational load of online control and enhancing the applicability of the control system.

Conclusion

Digital twin has garnered significant attention from both academia and industry. This paper proposes a combined contour error control method for five-axis machine tools based on digital twin. The digital twin established in this paper includes NURBS interpolator, machine kinematic model and servo dynamic model. When get tool path as input, the digital twin process interpolation and model predictive pre-compensation control, generating the pre-compensated command to reduce modeled error. The pre-compensated command with interpolated command and other required commands are sent to the real-time controller. An online feedback control is designed to reduce the unmodeled error. Using the S-shaped curve as an example, the method proposed in this paper reduces the maximum tool tip position contour error by 77.78% and its average value by 83.90%. Similarly, the maximum tool orientation contour error is reduced by 79.05%, and its average value is reduced by 86.66%. Additionally, real-time data (axis position, axis velocity, axis current, etc.) is also transmitted to the digital twin for analyzing machining performance and updating simulation model parameters. For situations where precision requirements are not met, the digital twin can adjust the feedrate during subsequent machining to achieve an overall closed-loop optimization. In this study, a smaller proportional gain was employed to prevent system oscillation, which somewhat limited the effectiveness of online control. Future research could explore more robust online control methods. Furthermore, future work could integrate tool path planning and speed planning into the digital twin closed-loop optimization framework to further enhance performance. Meanwhile, considering cutting force during the MPCC stage can enhance contour error control during actual machining and improve machining accuracy.

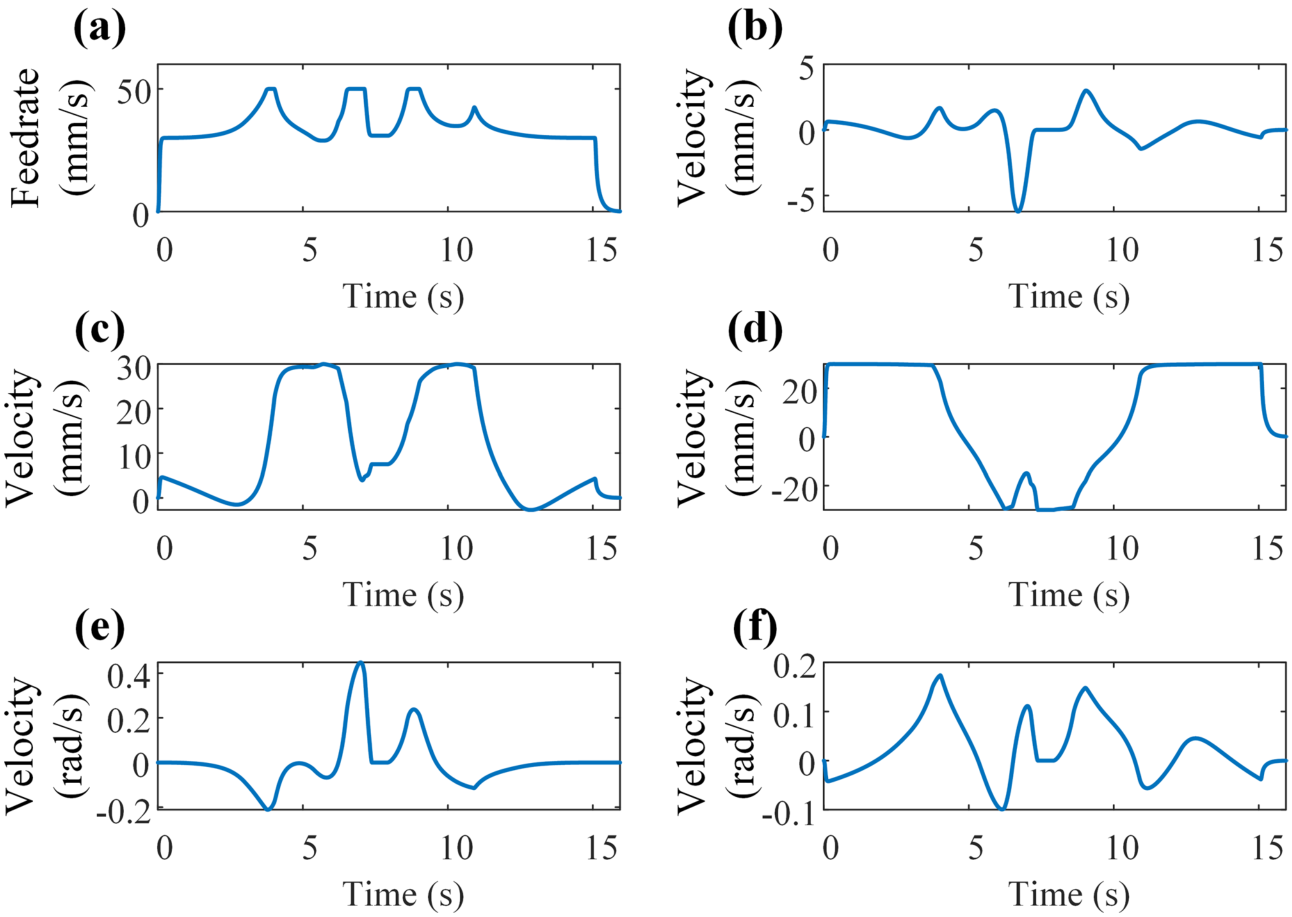


Fig. 13. Digital twin interpolation velocities: (**a**) tool tip position feedrate; (**b**) X-axis velocity; (**c**) Y-axis velocity; (**d**) Z-axis velocity; (**e**) A-axis velocity; (**f**) C-axis velocity.

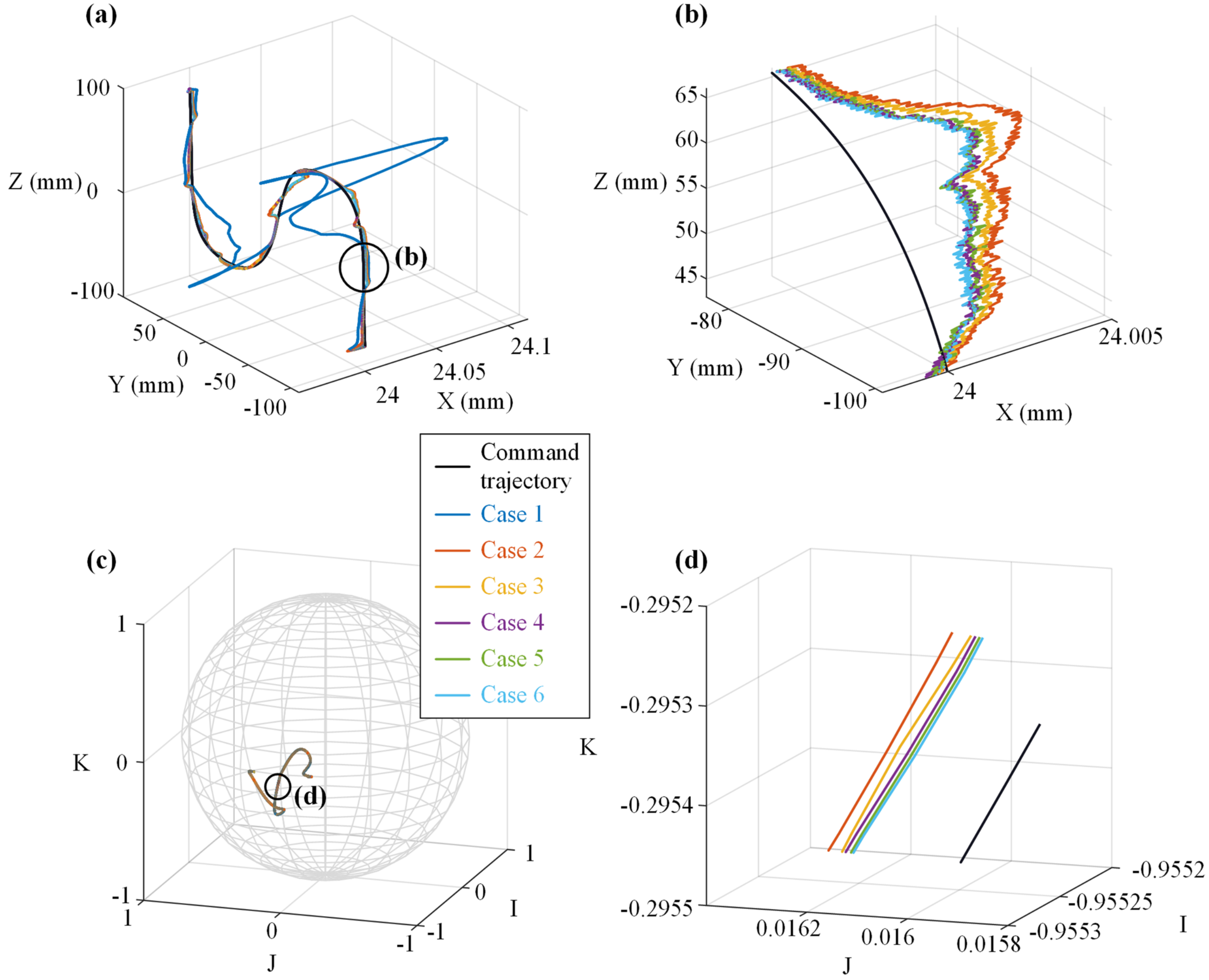


Fig. 14. Experiment results of tool tip position and tool orientation: (**a**) tool tip position; (**b**) partial enlargement of tool tip position; (**c**) tool orientation; (**d**) partial enlargement of tool orientation.

	Tool tip position tracking error(mm)		Tool orientation tracking error (mrad)	
	Max	Mean	Max	Mean
Case 1	0.8258	0.4642	8.4	1.8703
Case 2	0.2337	0.1249	2.1	0.296
Case 3	0.2334	0.1247	2.109	0.2988
Case 4	0.2330	0.1248	2.089	0.2990
Case 5	0.2331	0.1248	2.082	0.2990
Case 6	0.2234	0.1248	2.073	0.2994

Table 2. Tool tracking error under different cases.

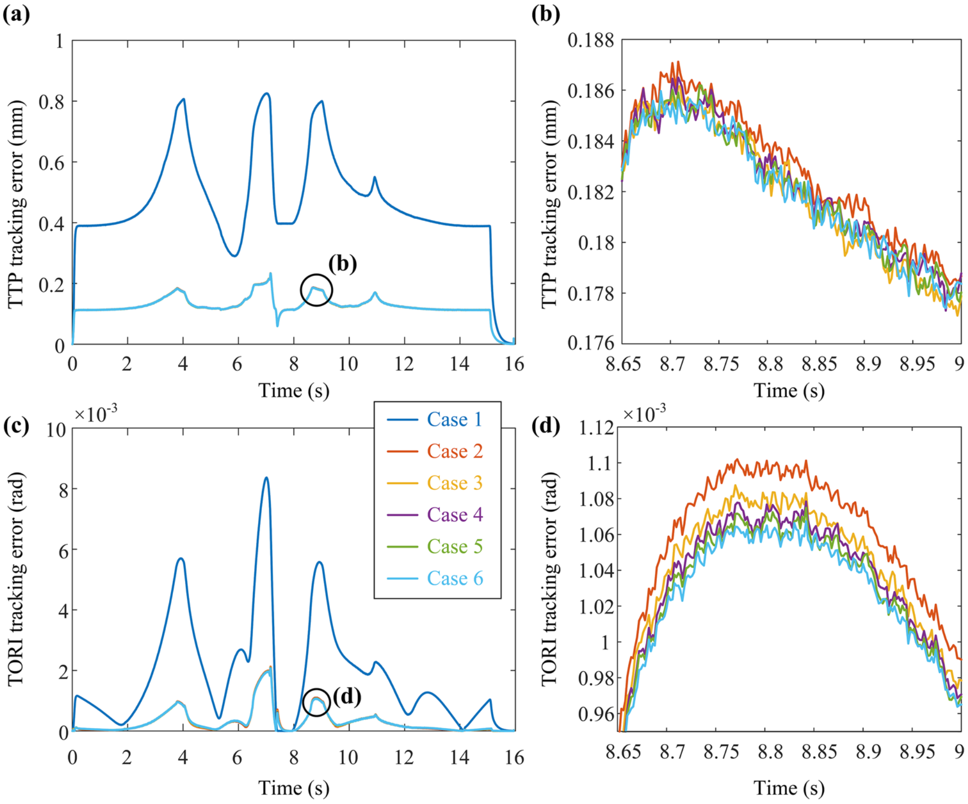


Fig. 15. Tool tracking error under different cases: (**a**) Tool tip position tracking error; (**b**) partial enlargement of tool tip position tracking error; (**c**) Tool orientation tracking error; (**d**) partial enlargement of tool orientation tracking error.

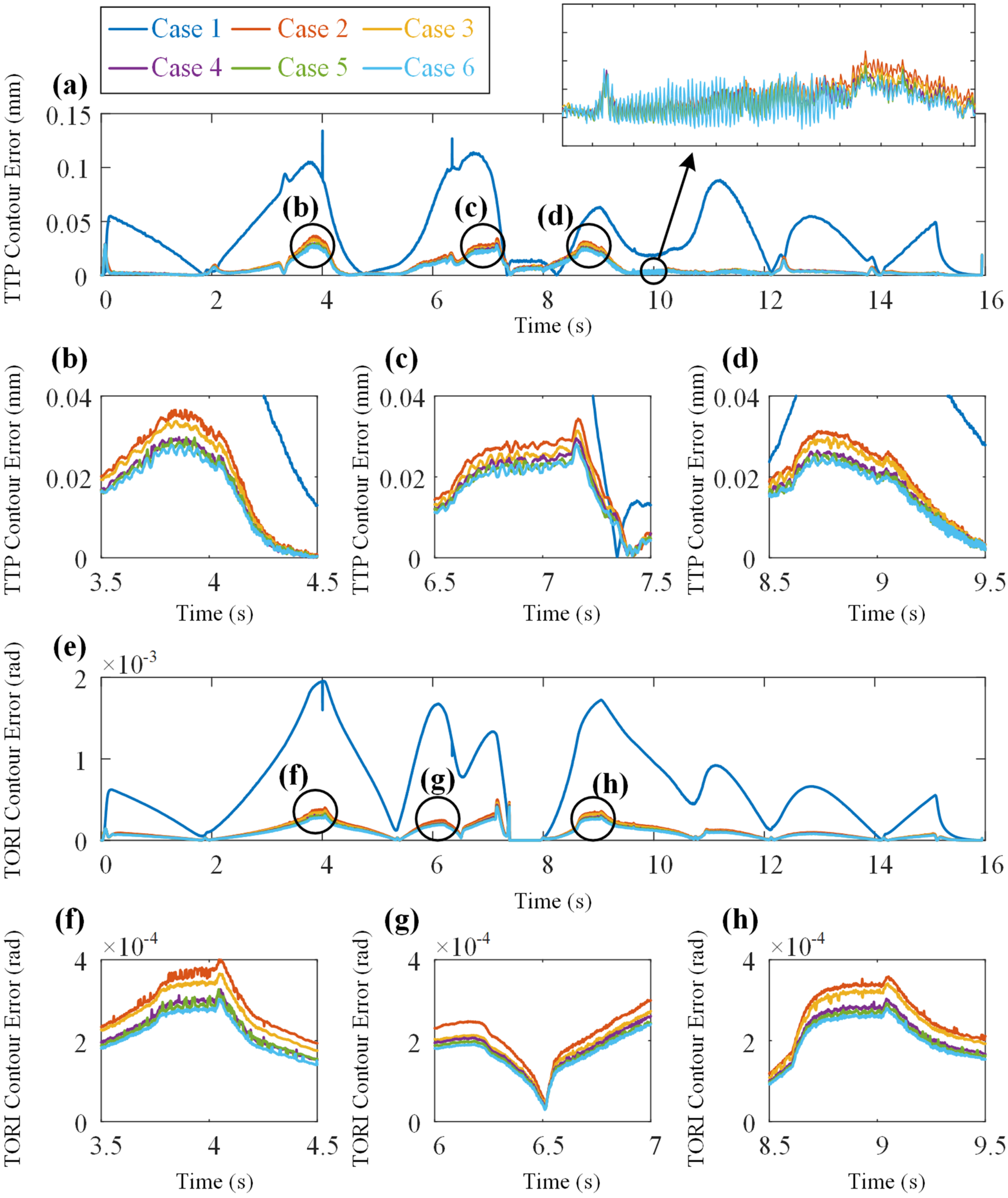


Fig. 16. Tool contour error under different cases: (**a**) tool tip position contour error; (**b**–**d**) partial enlargement of tool tip position contour error; (**e**) tool orientation contour error; (**f**–**h**) partial enlargement of tool orientation contour error.

	Tool tip position contour error(μm)		Tool orientation contour error(mrad)	
	Max	Mean	Max	Mean
Case 1	134.1	38.52	1.956	0.6824
Case 2	36.57	7.717	0.5009	0.1134
Case 3	33.78	7.078	0.4620	0.1039
Case 4	29.78	6.483	0.4364	0.09449
Case 5	29.75	6.211	0.4194	0.09104
Case 6	28.81	5.991	0.4108	0.08734

Table 3. Tool contour error under different cases.

Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
6.2477	1.075	1.0739	1.0764	1.0767	1.0744

Table 4. Average number of iterations for foot point search under different cases.

Data availability

The datasets used or analyzed during the current study are available from the corresponding author on reasonable request.

Received: 31 January 2025; Accepted: 12 May 2025
Published online: 22 May 2025

References

1. Yang, J. & Altintas, Y. A generalized on-line Estimation and control of five-axis contouring errors of CNC machine tools. *Int. J. Mach. Tools Manuf.* **88**, 9–23 (2015).
2. Yang, M., Yang, J. & Ding, H. A high accuracy on-line Estimation algorithm of five-axis contouring errors based on three-point Arc approximation. *Int. J. Mach. Tools Manuf.* **130–131**, 73–84 (2018).
3. Chen, H. R., Cheng, M. Y., Wu, C. H. & Su, K. H. Real time parameter based contour error Estimation algorithms for free form contour following. *Int. J. Mach. Tools Manuf.* **102**, 1–8 (2016).
4. Pi, S., Liu, Q. & Liu, Q. A novel dynamic contour error Estimation and control in high-speed CNC. *Int. J. Adv. Manuf. Technol.* **96**, 547–560 (2018).
5. Liu, Y., Wan, M., Xiao, Q. B. & Qin, X. B. Combined predictive and feedback contour error control with dynamic contour error Estimation for industrial Five-Axis machine tools. *IEEE Trans. Ind. Electron.* **69**, 6668–6677 (2022).
6. Yang, X., Seethaler, R., Zhan, C., Lu, D. & Zhao, W. A novel contouring error Estimation method for contouring control. *IEEEASME Trans. Mechatron.* **24**, 1902–1907 (2019).
7. Wang, Z., Hu, C. & Zhu, Y. Accelerated iteration algorithm based contouring error Estimation for multiaxis motion control. *IEEEASME Trans. Mechatron.* **27**, 452–462 (2022).
8. Li, X., Zhao, H., Zhao, X. & Ding, H. Dual sliding mode contouring control with high accuracy contour error Estimation for five-axis CNC machine tools. *Int. J. Mach. Tools Manuf.* 74–82. https://doi.org/10.1016/j.ijmachtools.2016.05.007 (2016).
9. Ghaffari, A. & Ulsoy, A. G. Dynamic contour error Estimation and feedback modification for High-Precision contouring. *IEEEASME Trans. Mechatron.* **21**, 1732–1741 (2016).
10. Yang, J., Zhang, H. T. & Ding, H. Contouring error control of the tool center point function for five-axis machine tools based on model predictive control. *Int. J. Adv. Manuf. Technol.* **88**, 2909–2919 (2017).
11. Lu, Y. et al. Research on a novel integrated control strategy for contour error compensation of biaxial CNC machining. *Int. J. Adv. Manuf. Technol.* **130**, 385–402 (2024).
12. Zhao, H., Zhu, L. & Ding, H. Cross-coupled controller design for triaxial motion systems based on second-order contour error Estimation. *Sci. China Technol. Sci.* **58**, 1209–1217 (2015).
13. Xu, Z. et al. Intelligent contour error compensation of ultraprecision machining using hybrid Mechanism-Data-Driven model assisted with IoT framework. *IEEE Trans. Ind. Inf.* **20**, 11815–11824 (2024).
14. Wang, Z., Hu, C., Zhu, Y. & Zhu, L. Prediction-Model-Based contouring error iterative precompensation scheme for precision multiaxis motion systems. *IEEEASME Trans. Mechatron.* **26**, 2274–2284 (2021).
15. Yang, X., Seethaler, R., Zhan, C., Lu, D. & Zhao, W. A model predictive contouring error precompensation method. *IEEE Trans. Ind. Electron.* **67**, 4036–4045 (2020).
16. Wang, Z., Hu, C. & Zhu, Y. Dynamical model based contouring error Position-Loop feedforward control for multiaxis motion systems. *IEEE Trans. Ind. Inf.* **15**, 4686–4695 (2019).
17. Pi, S., Liu, Q., Sun, P. & Tong, X. Five-axis contour error control considering milling force effects for CNC machine tools. *Int. J. Adv. Manuf. Technol.* **98**, 1655–1669 (2018).
18. Lyu, D., Ye, X., Zhang, H. & Liu, H. Offline fast modification of position loop gains for five-axis machining. *Int. J. Adv. Manuf. Technol.* **127**, 3951–3963 (2023).
19. Duong, T. Q., Rodriguez-Ayerbe, P., Lavernhe, S., Tournier, C. & Dumur, D. Contour error pre-compensation for five-axis high speed machining: offline gain adjustment approach. *Int. J. Adv. Manuf. Technol.* **100**, 3113–3125 (2019).
20. Kang, Z. J., Zhao, K., Li, S. & Liu, Z. Design of a real-time three-axis controller for contour error reduction based on model predictive control. *Proc. Inst. Mech. Eng. Part. C J. Mech. Eng. Sci.* **236**, 9241–9254 (2022).
21. Margolis, B. W. L. & Farouki, R. T. Inverse dynamics toolpath compensation for CNC machines based on model predictive control. *Int. J. Adv. Manuf. Technol.* **109**, 2155–2172 (2020).

22. Wang, Y. W., Zhang, W. A., Dong, H. & Yu, L. A GESO based MPC approach to contour error control of networked motion control system. *Int. J. Syst. Sci.* **50**, 2216–2225 (2019).
23. Zhang, X., Shi, T., Wang, Z., Geng, Q. & Xia, C. Generalized predictive contour control of the biaxial motion system. *IEEE Trans. Ind. Electron.* **65**, 8488–8497 (2018).
24. Zhang, L., Liu, J. & Zhuang, C. Digital twin modeling enabled machine tool intelligence: A review. *Chin. J. Mech. Eng.* **37**, 47 (2024).
25. Wang, J., Niu, X., Gao, R. X., Huang, Z. & Xue, R. Digital twin-driven virtual commissioning of machine tool. *Robot Comput. -Integr Manuf.* **81**, 102499 (2023).
26. Sun, X., Zhang, F., Niu, X. & Wang, J. A digital twin commissioning method for machine tools based on scenario simulation. *J. Manuf. Syst.* **77**, 697–707 (2024).
27. Li, R., Zhao, L., Zhou, B. & Zhao, W. Research on machining error control method driven by Digital-twin model of dynamic characteristics of machining system. *Integr. Ferroelectr.* **237**, 321–335 (2023).
28. Ward, R. et al. Machining digital twin using real-time model-based simulations and Lookahead function for closed loop machining control. *Int. J. Adv. Manuf. Technol.* **117**, 3615–3629 (2021).
29. Tong, X., Liu, Q., Zhou, Y. & Sun, P. A digital twin-driven cutting force adaptive control approach for milling process. *J. Intell. Manuf.* https://doi.org/10.1007/s10845-023-02193-2 (2023).
30. Kim, H. & Okwudire, C. E. Intelligent feedrate optimization using a physics-based and data-driven digital twin. *CIRP Ann.* **72**, 325–328 (2023).
31. Tong, X., Liu, Q., Pi, S. & Xiao, Y. Real-time machining data application and service based on IMT digital twin. *J. Intell. Manuf.* **31**, 1113–1132 (2020).
32. Lu, Y. et al. Digital Twin-driven smart manufacturing: connotation, reference model, applications and research issues. *Robot Comput. -Integr Manuf.* **61**, 101837 (2020).
33. Zi, X., Gao, S. & Xie, Y. An online monitoring method of milling cutter wear condition driven by digital twin. *Sci. Rep.* **14**, 4956 (2024).
34. Guo, M., Fang, X., Hu, Z. & Li, Q. Design and research of digital twin machine tool simulation and monitoring system. *Int. J. Adv. Manuf. Technol.* **124**, 4253–4268 (2023).
35. Sa, G. et al. A digital twin synchronous evolution method of CNC machine tools associated with dynamic and static errors. *Int. J. Adv. Manuf. Technol.* **134**, 2753–2763 (2024).
36. Tripura, T., Desai, A. S., Adhikari, S. & Chakraborty, S. Probabilistic machine learning based predictive and interpretable digital twin for dynamical systems. *Comput. Struct.* **281**, 107008 (2023).
37. Zhang, C. et al. A multi-access edge computing enabled framework for the construction of a knowledge-sharing intelligent machine tool swarm in industry 4.0. *J. Manuf. Syst.* **66**, 56–70 (2023).
38. Liu, Y., Chen, M. & Sun, Y. Global toolpath modulation–based contour error pre-compensation for multi-axis CNC machining. *Int. J. Adv. Manuf. Technol.* **125**, 3171–3189 (2023).
39. Huang, X. et al. Real-time estimate and control contour errors for five-axis local smoothed toolpaths based on airthoid splines. *Int. J. Adv. Manuf. Technol.* **127**, 1483–1504 (2023).
40. Xiao, Q. B., Wan, M., Zhang, W. H. & Yang, Y. Tool orientation optimization for the five-axis CNC machining to constrain the contour errors without interference. *J. Manuf. Process.* **76**, 46–56 (2022).
41. Chen, M. *Research on Feedrate Scheduling and Contour Error Pre-compensation for Five-axis Machining Based on Constrained Linear Optimization* (Dalian University of Technology, 2021).
42. Sun, P., Liu, Q., Wang, J., Yin, Z. & Wang, L. Rotation-Angle solution and singularity handling of Five-Axis machine tools for dual NURBS interpolation. *Machines* **11**, 281 (2023).
43. Wang, L. et al. Dynamic look-ahead feedrate scheduling method based on sliding mode velocity control. *Sci. Rep.* **14**, 15424 (2024).
44. Tang, L. & Landers, R. G. Predictive contour control with adaptive feed rate. *IEEEASME Trans. Mechatron.* **17**, 669–679 (2012).

Author contributions

L.W.: conceptualization, methodology, formal analysis, software, validation, original draft, review and editing; R.Y.: validation, software, review and editing; S.L.: methodology, review and editing; Z.Y.: validation, software, review and editing; S.X.: software, review and editing; Y.L.: conceptualization, supervision; Q.L.: conceptualization, supervision;

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-025-02047-2.

Correspondence and requests for materials should be addressed to Y.L. or Q.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.