Robotica (2025), **43**, pp. 1533–1547
doi:10.1017/S0263574725000359

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Determination method of limit interpolation points in industrial robot control system

Zhenyang Lv[1], Yi Wang[2], Guangpeng Zhang[1] and Xionghui Wang[1]

[1]School of Mechanical and Precision Instrument Engineering, Xi'an University of Technology, Xi'an, Shaanxi, People's Republic of China
[2]School of Art and Design, Xi'an University of Technology, Xi'an, Shaanxi, People's Republic of China
Corresponding author: Guangpeng Zhang; Email: gpzhang@xaut.edu.cn

Received: 1 October 2024; **Revised:** 21 February 2025; **Accepted:** 24 February 2025; **First published online:** 4 April 2025

Keywords: industrial robot; beckhoff controller; arc interpolation; limit interpolation points

Abstract

With advancements in industrial robot technology and the ongoing enhancements in control system performance, the demand for precise robot motion is increasing. Generally, an increased number of interpolation points enhances the precision of robot movement, but excessive points can lead to jittering and out-of-step issues. This paper investigates the relationship between the number of motion interpolation points and the response times of the control system and the robot's terminal velocity, based on the theoretical calculation and experimental analysis of the limit interpolation points for the control system of a self-developed 6-DOF (Six Degree of Freedom) robot. The method for calculating limit interpolation points is refined using the least squares method, and equations are derived for different control system response time and robot's terminal velocity reaction times. The validity of the prediction curves is verified through experimental analysis.

1. Introduction

Currently, the application of industrial robots is expanding rapidly, encompassing fields such as food packaging, medical equipment, spraying, handling, and processing [1–5]. The control system is central to industrial robots, and its quality not only influences the robot's flexibility and execution capability but also its operational accuracy [6]. Interpolation points are the primary means by which the control system manages industrial robots, and their quantity directly impacts the robots' accuracy and stability [7–9]. It is well understood that increasing the number of interpolation points can enhance motion accuracy and stability, and prevent abrupt movements, but an excessive number can lead to jittering and loss of synchronization during motion [10–12]. In this paper, we investigate the relationship between the response time of the control system and the robot's terminal velocity to determine the limit interpolation points under various conditions.

In recent years, as industrialization has advanced, the demands on industrial robots have become more stringent, with applications increasingly requiring high precision and efficiency [13–16]. The motion accuracy of a robot is influenced not only by its structure but also significantly by the control system [17–25]. The greater the number of interpolation points the control system can manage, the smoother the path planning becomes, enhancing the stability of robot motion and allowing for better control of speed and acceleration to prevent abrupt movements. Fewer interpolation points can reduce computational and control complexity, improve the efficiency and real-time performance of path planning, and decrease system load. However, the robot's terminal velocity and the control system's response time are coupled with the interpolation points, necessitating the selection of an appropriate number of interpolation points to balance accuracy, efficiency, and real-time performance based on specific conditions. In practical application, experiments and theoretical calculations are typically required to determine the

© The Author(s), 2025. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

limit interpolation points and provide a basis for selecting the number of interpolation points in path planning.

Current research has identified that robots exhibit significant jittering and out-of-step when their terminal velocity exceeds a certain threshold [26, 27], yet no comprehensive studies have been undertaken. It is widely recognized that increasing the number of interpolation points enhances the precision of the robot's motion trajectory; however, excessive interpolation points can lead to CPU (Central Processing Unit) utilization exceeding acceptable levels. If the control system's response time is prolonged, the robot may experience delays in the executing interpolation motion, potentially causing discrepancies between planned and actual trajectories, thereby affecting the robot's accuracy and stability.

In summary, this paper presents the relationship between the control system's response time, the robot's terminal velocity, and the limit interpolation points through theoretical calculations and experimental validation. This provides a foundation for the rational selection of interpolation points, ensuring stable and smooth robot motion while avoiding abrupt movements.

The main research contents of this paper include the following: Section 2 introduces division of the control system's response time and the principle of time-sharing multitasking; Section 3 presents a method for calculating the theoretical limit interpolation points. In Section 4, the experimental equipment and procedures are introduced, and multiple experimental results are obtained by varying the response time and terminal velocity, followed by analysis of the results. In Section 5, the method for calculating theoretical limit interpolation points is refined, and the optimized calculation method is validated; Section 6 is the summary of the full text.

2. Response mechanism analysis of control system

The number of limit interpolation points in a controller typically depends on its computing power, sampling frequency, and real-time requirements. In a real-time control system, generating and executing interpolation points consumes the controller's computational resources. Therefore, to determine the controller's capacity for handling limit interpolation points, a comprehensive evaluation based on these factors is necessary. In practical applications, the controller's maximum interpolation point processing capacity is typically determined through experiments. Analyzing the number of limit interpolation points in a robot control system can address the issue of excessive CPU utilization during the robot control process.

In the robot control system, response time directly influences the robot's accuracy and stability during trajectory planning and interpolation motion. The shorter the response time, the more quickly the robot can initiate movement following a control instruction, thereby reaching each interpolation point with greater accuracy and achieving a smoother motion trajectory. If the response time of the robot control system is prolonged, there may be delays in the execution of interpolation motions, and the planned trajectory may deviate from the actual motion, thereby affecting the robot's accuracy and stability.

This paper uses the Beckhoff controller in the self-developed robot as an example to demonstrate that the control system's response time comprises three components: input time, calculation time, and output time. The input time and output time are determined by I/O operations, while the calculation time is ensured by the real-time core of TwinCAT. Based on the control system's architecture, during the execution of the TwinCAT control task, the computer allocates a portion of the CPU's computing power through real-time checks of various functional blocks at the operating system's base level [28], as shown in Fig. 1.

Once the response time for the corresponding task in the robot control system is set, the subsequent allocation of CPU time is shown in Fig. 2. The base time is the minimum time slice that Windows allocates for CPU time resources to TwinCAT. The CPU limit represents the maximum percentage of CPU resources that TwinCAT can utilize within this time slice. In PLC (Programmable Logic Controller) tasks, the real-time kernel initially allocates the CPU based on the base time, then divides it into four processes within a PLC cycle, as shown in Fig. 2, for each 1 ms real-time task, and ultimately responds to human-computer interaction requests. If the CPU limit is reached during the application and the task

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

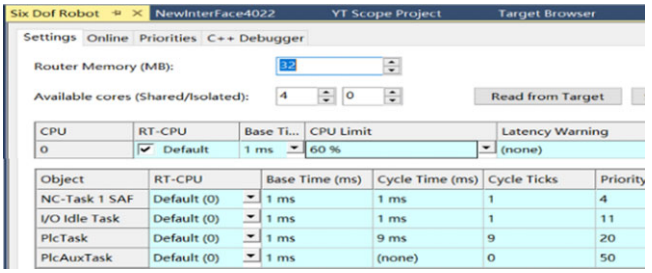


Figure 1. *Task division of CPU in the control system.*

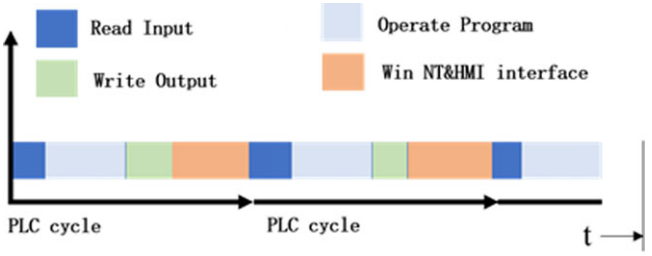


Figure 2. *CPU time allocation during system running.*

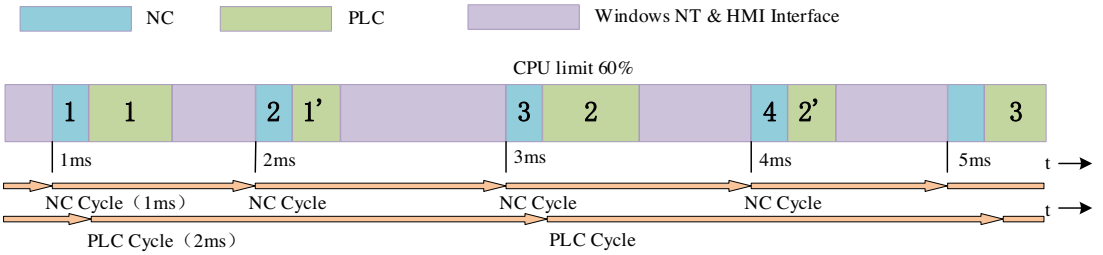


Figure 3. *Execution sequence of time-sharing multitasking.*

is not completed, the TwinCAT thread will suspend and proceed to the next stage of execution [29, 30]. Generally, the CPU limit and base time can be adjusted, and setting the response time appropriately facilitates a broader range of system applications.

The CPU of a computer equipped with TwinCAT must allocate a portion of its computational power to perform the functions of PLC module and the NC (Numerical Control) module in the motion controller while the system is operational. Consequently, there are two primary tasks in the system: NC-TASK1 SAF and Plc Task, with periods of 1 and 10 ms, respectively. Based on this configuration, when TwinCAT is operational, the execution timing of the two tasks in the system is shown in Fig. 3.

As shown in Fig. 3, in this system, NC and PLC tasks run on dedicated real-time cores, the NC task has a higher priority than the PLC task due to its higher priority setting. Consequently, at the start of the 1 ms cycle, the NC task is executed first. The PLC task, which has a lower priority, is executed only after the NC task completes and the CPU Limit is not exceeded. If the first pass of the PLC task is not completed within the CPU time limit, the CPU will interrupt the current PLC task and proceed to the next task. For instance, during task execution, if the system is still executing the first pass of the PLC task after 3 ms, a timeout marker, "timeout completion counter," will automatically trigger, allowing the PLC program to immediately detect the task status. When the system experiences consecutive timeouts, the PLC will detect that the CPU utilization has exceeded the standard threshold.

There are two approaches to address the issue of excessive CPU utilization: optimizing the code and optimizing task response time. Of these two approaches, optimizing task response time can lead to more efficient control system responses and faster coordination of CPU resources. Therefore, to ensure that

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

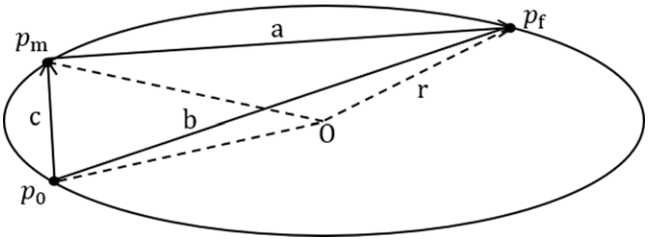


Figure 4. *Schematic diagram of arc trajectory interpolation.*

the robot control system achieves accurate and stable motion control, the control system's response time is optimized, and the relationship between response time and motion interpolation points is established. This allows the robot to respond quickly to control commands and achieve effective motion interpolation, thereby enhancing production and operational efficiency.

3. Determination method of limit interpolation points

To investigate the relationship between the control system's response time, the robot's terminal velocity, and the number of limit interpolation points, it is essential to control the robot to perform point-to-point interpolation motion, and subsequently assess the control system's performance based on the robot's motion.

The position of the manipulator's end-effector is interpolated using the circular interpolation principle. Define $p_0(x_0, y_0, z_0)$ as the starting point of the arc trajectory, $p_m(x_m, y_m, z_m)$ as the middle point, and $p_f(x_f, y_f, z_f)$ as the target point. These three points form the internal triangle of the arc trajectory, as shown in Fig. 4.

In Fig. 4 (a), (b), and (c) represent the sides of the internal triangle, and the distance l between any two points can be expressed in coordinates as: $l = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Let C_0 be the circumference of the internal triangle, then the radius of the arc of the path traveled can be expressed as:

$$r = \frac{abc}{4S_\Delta} = \frac{abc}{4\sqrt{\frac{C_0}{2}\left(\frac{C_0}{2} - a\right)\left(\frac{C_0}{2} - b\right)\left(\frac{C_0}{2} - c\right)}} \tag{1}$$

where r is the radius and S_Δ is the circumcircle area.

Utilize the points on the circumference to determine the trajectory function of the circle and set the center coordinates of the circular trajectory to be $O(x, y, z)$. The trajectory function of the circle is then given by:

$$\begin{cases} (\mathrm{x} - \mathrm{x_0})^2 + (\mathrm{y} - \mathrm{y_0})^2 + (\mathrm{z} - \mathrm{z_0})^2 = (\mathrm{x} - \mathrm{x_m})^2 + (\mathrm{y} - \mathrm{y_m})^2 + (\mathrm{z} - \mathrm{z_m})^2 \\ (\mathrm{x} - \mathrm{x_m})^2 + (\mathrm{y} - \mathrm{y_m})^2 + (\mathrm{z} - \mathrm{z_m})^2 = (\mathrm{x} - \mathrm{x_f})^2 + (\mathrm{y} - \mathrm{y_f})^2 + (\mathrm{z} - \mathrm{z_f})^2 \end{cases} \tag{2}$$

Then,

$$\begin{cases} 2\,(x_m - x_0)\,x + 2\,(y_m - y_0)\,y + 2\,(z_m - z_0)\,z = -x_0^2 - y_0^2 - z_0^2 + x_m^2 + y_m^2 + z_m^2 \\ 2\,(x_f - x_0)\,x + 2\,(y_f - y_0)\,y + 2\,(z_f - z_0)\,z = -x_0^2 - y_0^2 - z_0^2 + x_f^2 + y_f^2 + z_f^2 \end{cases} \tag{3}$$

Additionally, let the plane equation of the trajectory circle be:

$$\mathrm{Ax} + \mathrm{By} + \mathrm{Cz} = 1 \tag{4}$$

Equations (3) and (4) are transformed into matrix form:

$$\begin{bmatrix} \mathrm{x_0} & \mathrm{y_0} & \mathrm{z_0} \\ \mathrm{x_m} & \mathrm{y_m} & \mathrm{z_m} \\ \mathrm{x_f} & \mathrm{y_f} & \mathrm{z_f} \end{bmatrix} \begin{bmatrix} \mathrm{A} \\ \mathrm{B} \\ \mathrm{C} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{5}$$

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

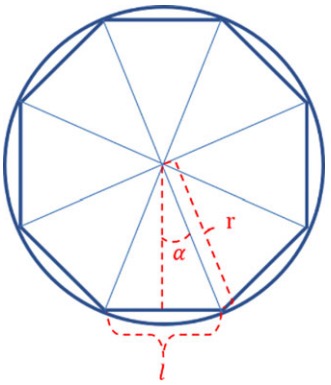


Figure 5. *Division of circular track points.*

By combining Eqs. (3) and (5), the center coordinates can be determined as:

$$\begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \end{bmatrix} = \begin{bmatrix} 2\,(\mathrm{x_m}-\mathrm{x_0}) & 2\,(\mathrm{y_m}-\mathrm{y_0}) & 2\,(\mathrm{z_m}-\mathrm{z_0}) \\ 2\,(\mathrm{x_f}-\mathrm{x_0}) & 2\,(\mathrm{y_f}-\mathrm{y_0}) & 2\,(\mathrm{z_f}-\mathrm{z_0}) \\ \mathrm{A} & \mathrm{B} & \mathrm{C} \end{bmatrix}^{-1} \begin{bmatrix} -\mathrm{x_0^2}-\mathrm{y_0^2}-\mathrm{z_0^2}+\mathrm{x_m^2}+\mathrm{y_m^2}+\mathrm{z_m^2} \\ -\mathrm{x_0^2}-\mathrm{y_0^2}-\mathrm{z_0^2}+\mathrm{x_f^2}+\mathrm{y_f^2}+\mathrm{z_f^2} \\ 1 \end{bmatrix} \tag{6}$$

By determining the coordinates of the circle's center, the radius can be calculated, and the radius parameter can be incorporated into the G-code. The information for a single arc interpolation point is generated, and this method can be used to generate the information for each point on the arc trajectory.

Assuming that there are n trajectory points on the arc, dividing the arc angle θ into n equal parts. If i is less than n, the arc angle formed by the i th tracepoint and the starting point p_{02} can be expressed as:

$$\theta_\mathrm{i} = \frac{\mathrm{i}}{\mathrm{n}}\theta \tag{7}$$

Then, introduce a Z axis to the plane containing the arc trajectory. The angle θ_i through which the robot's end effector rotates around the Z axis, from the starting point $p_{02}(x_0, y_0, z_0)$ to the i th trajectory point.

At the same time, the unit vector of the arc trajectory's rotation axis is:

$$\mathrm{w} = \frac{\mathrm{p_{02}}}{\|\mathrm{p_{02}}\|} \times \frac{\mathrm{p_{f2}}}{\|\mathrm{p_{f2}}\|} = \begin{bmatrix} \mathrm{w_x} & \mathrm{w_y} & \mathrm{w_z} \end{bmatrix} \tag{8}$$

where p_{f2} represents the coordinates of the target point of the arc trajectory, and p_{02} represents the coordinates of the starting point.

The rotation matrix is given by:

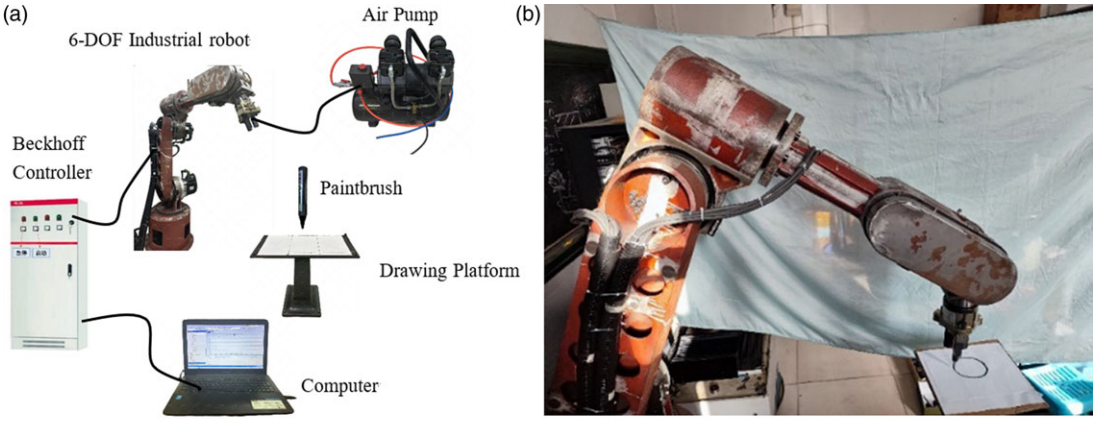
$$\mathrm{R_i} = \begin{bmatrix} \mathrm{c\theta_i} + \mathrm{w_x^2}\,(1-\mathrm{c\theta_i}) & \mathrm{w_{xy}}\,(1-\mathrm{c\theta_i}) - \mathrm{w_z s\theta_i} & \mathrm{w_{xz}}\,(1-\mathrm{c\theta_i}) + \mathrm{w_y s\theta_i} \\ \mathrm{w_{xy}}\,(1-\mathrm{c\theta_i}) + \mathrm{w_z s\theta_i} & \mathrm{c\theta_i} + \mathrm{w_y^2}\,(1-\mathrm{c\theta_i}) & \mathrm{w_{yz}}\,(1-\mathrm{c\theta_i}) - \mathrm{w_x s\theta_i} \\ \mathrm{w_{xz}}\,(1-\mathrm{c\theta_i}) - \mathrm{w_y s\theta_i} & \mathrm{w_{yz}}\,(1-\mathrm{c\theta_i}) + \mathrm{w_x s\theta_i} & \mathrm{c\theta_i} + \mathrm{w_z^2}\,(1-\mathrm{c\theta_i}) \end{bmatrix} \tag{9}$$

where $w_{xy} = w_x \times w_y$, $w_{xz} = w_x \times w_z$, $w_{yz} = w_y \times w_z$, $c\theta_i = \cos\theta_i$, $s\theta_i = \sin\theta_i$.

Consequently, by combining the target position coordinates and rotation matrix, the pose matrix of the robot 's end effector can be derived. This pose matrix, corresponding to each point on the arc trajectory, can be utilized to determine the robot pose, thereby enabling the robot to perform arc interpolation motion.

As shown in Fig. 5, the greater the number of interpolation points along the robot's motion trajectory on the circumference, the smaller the approximation error of the line becomes. However, the number of steps in the robot's motion increases, while the step length decreases. Therefore, it is assumed that there is a limit to the number of interpolation points given a certain robot's terminal velocity and the control system's response time. Thus, there are m points on the entire circular trajectory of the robot,

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press



Experimental platform construction *Experimental process diagram*

Figure 6. *Limit interpolation point exploration experiment.*

with a step between each pair of points. When the number of points on the circle is sufficiently large, the length of each step approaches the length of the circle. The robot's terminal velocity is v, the control system's response time is t, and the number of limit interpolation points on the circumference is m. The circumference is then divided into m segments, with the chord length of each segment being l, then:

$$\alpha = \frac{360^\circ}{2\mathrm{m}} \tag{10}$$

where α is the angle between the angular bisector of the circular radius chord and the circle's center angle.

Let the robot's step size be s, then $s=vt$. If the step length s is less than the chord length l, the robot movement will not be out-of-step, so $l > s$, then:

$$2\mathrm{r}\sin\left(\frac{360^\circ}{2\mathrm{m}}\right) \geq \mathrm{vt} \tag{11}$$

Then,

$$\mathrm{m} \leq \frac{180^\circ}{\arcsin\left(\frac{\mathrm{vt}}{2\mathrm{r}}\right)} \tag{12}$$

4. Limit interpolation points determination experiment

To validate the accuracy of the theoretical calculation of limit interpolation points, the robot's motion under varying response times and terminal velocities was recorded through experiments, and the results were analyzed.

4.1 Experimental process

To investigate the relationship between the control system's response time, the robot's terminal velocity, and the limit interpolation point, an experiment was designed to have the robot's gripper draw a circular trajectory on the drawing plane. The motion data were recorded and analyzed using ScopeView, a graphical output analysis tool in Beckhoff's PC-based control software TwinCAT3. The experimental equipment primarily consists of a 6-DOF Industrial Robot, Beckhoff Controller, Air Pump, Drawing Platform, etc., as shown in Fig. 6(a). During the experiment, the controller directs the robot to draw the circular trajectory on the platform, as shown in Fig. 6(b).

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

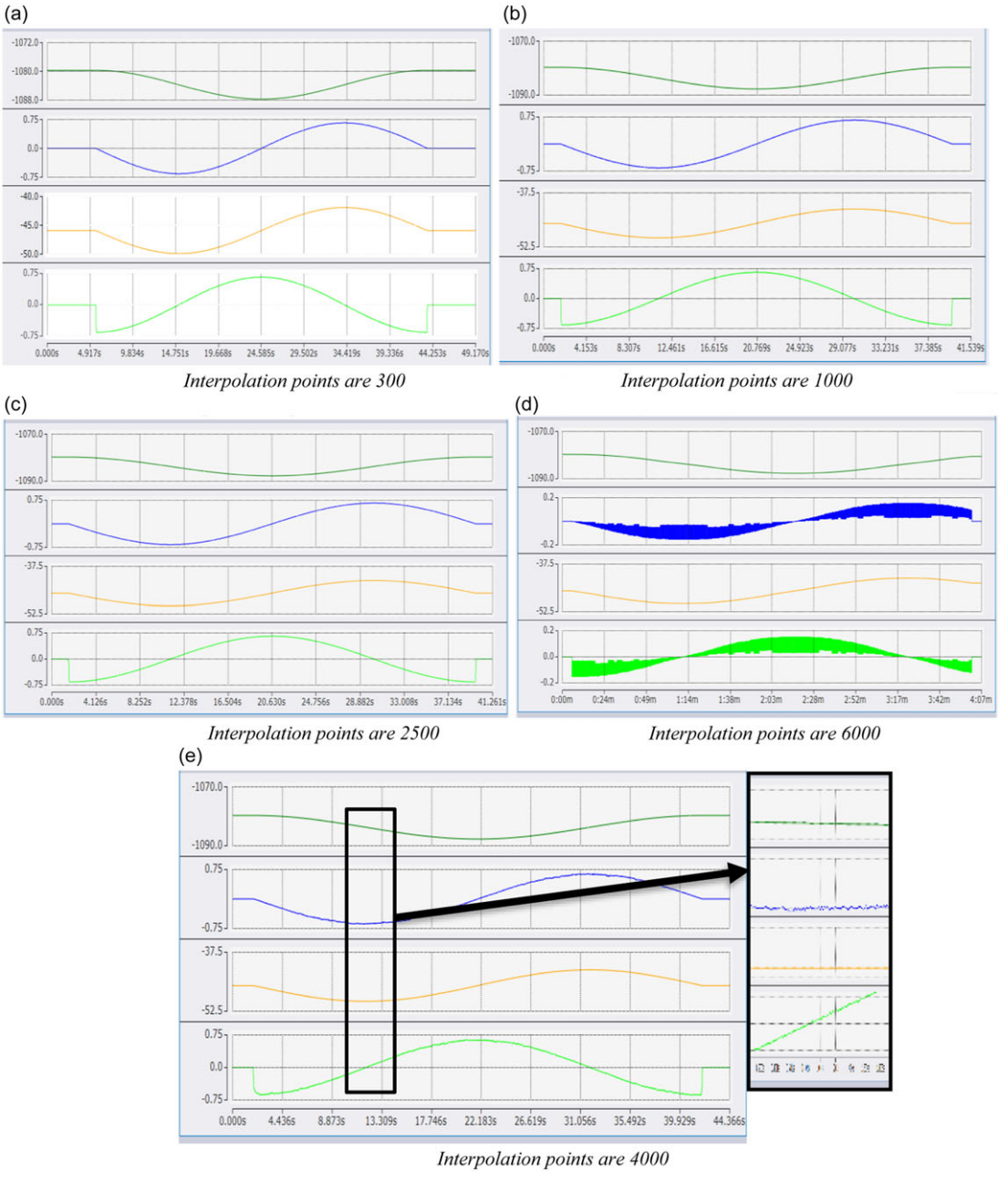


Figure 7. *Experimental data recording when the response time is 10 ms and the robot's terminal velocity is 40 mm/min.*

The steps of the experiment are as follows:

(1) To obtain the robot's motion trajectory, a cylinder with a diameter of 8 mm and a height of 25 mm was modeled in 3D software, and a G-code program with varying interpolation points was generated using post-processing software.

(2) Position the drawing platform at the designated measurement location based on the robot's origin and adjust it horizontally.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

(3) Create a new project, Scope YT project, using ScopeView in TwinCAT3, and add variable parameters for the speed and position of the X and Y axes in the Cartesian coordinate system.

(4) Set the robot to its zero position and adjust the control system's response time to 5 ms.

(5) During the experiment, the robot's terminal velocity was varied from 40 to 500 mm/min, and all programs with different interpolation points were executed and recorded.

(6) Switch the system to configuration mode, adjust the response time to 10 ms, and repeat the above steps to complete the experiment.

During the experiment, the changes in velocity and position of the robot's end effector along the X and Y axes over time were recorded. For example, one set of data was collected under the conditions of a 10 ms control system response time and a robot's terminal velocity of 40mm/min. The number of motion interpolation points was 300, 1000, 2500, 4000, and 6000, respectively, and the recorded data are shown in Fig. 7. In Fig. 7, the horizontal axis represents time, and the vertical axis represents displacement in the X, Y, and Z directions. (The axis labels for Figs. 12 and 13 are consistent with those in Fig. 7.)

Figure 7 shows that the position and velocity curves for the X and Y axes in (a), (b), and (c) are relatively smooth, indicating that the robot did not exhibit jitter during the experiment. The blue and green lines in Fig. 7(d) represent the speed curves of the X and Y axes, respectively, and it can be observed that these two lines experienced severe shaking and partial missing. In Fig. 7(e), the overall motion curve appears relatively smooth, but upon magnification, it is evident that the velocity and position curves have fluctuated, and the motion time has increased. This indicates that the robot noticeably dithered as the number of interpolation points increased, resulting in unstable speed.

As indicated by Eq. (12), when the model diameter is 8 mm, the control system's response time is 10 ms, and the robot's terminal velocity is 40 mm/min, the number of limit interpolation points is 3769. Consequently, the motion in Fig. 7(a), (b), and (c) remains stable as it does not exceed the limited number of motion interpolation points. Figure 7(d) significantly exceeds the limit of interpolation points, resulting in the control system's response time exceeding the limit, causing the robot's movement to be out-of-step and exhibit drastic fluctuations. Figure 7(e) slightly exceeds the limit of interpolation points, leading to jitter.

4.2 Analysis of experimental data

After all experiments are completed and experimental data are recorded, the data are divided into two categories based on the control system's response times. One category corresponds to a control system's response time of 5 ms, while the other corresponds to 10 ms, with the robot's terminal velocity incrementally varying from 40 to 500 mm/min. The data from both categories were compiled to create Fig. 8. In Fig. 8, the vertical axis represents the reaction time of the interpolation velocity, which is the time taken for the robot's end to move along an 8 mm diameter circular path at different speeds over one cycle, and the horizontal axis represents the number of interpolation points the robot moves during actual motion. When the control system's response time is 5 ms, the different robot terminal velocities are represented by 8 lines ranging from blue to purple; when the control system's response time is 10 ms, the different robot terminal velocities are represented by various shades of green; the two red lines represent the predicted limit interpolation points for a specific robot's terminal velocity under different control system's response time.

As observed from the curves of different speeds in Fig. 8:

(1) With the same response time and at a certain velocity, the reaction time remains relatively constant initially as the number of interpolation points increases, but once the limit is reached, the reaction time increases linearly with further increases in the number of interpolation points.

(2) With the same response time, the number of limit interpolation points decreases as the robot's terminal velocity increases.

(3) Compared to a control system's response time of 5 ms, the number of interpolation points for a response time of 10 ms, at the same robot's terminal velocity, is reduced.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

Table I. *Limit interpolation points and reaction time under 5 and 10 ms response time.*

	5 ms			10 ms		
Velocity mm/min	**Actual limit interpolation points/(PCS)**	**Theoretical limit interpolation points/(PCS)**	**reaction time/s**	**Actual limit interpolation points/(PCS)**	**Theoretical limit interpolation points/(PCS)**	**reaction time/s**
40	7539	7359	37.728	3768	3769	38.208
50	6031	6031	30.176	3015	3015	30.448
70	4308	4308	21.584	2153	2154	21.654
100	3015	3015	15.125	1500	1508	15.282
200	1000	1508	7.581	753	753	8.01
300	550	1005	5.276	380	502	5.232
400	300	753	3.957	250	376	4.026
500	100	603	3.138	150	301	3.222

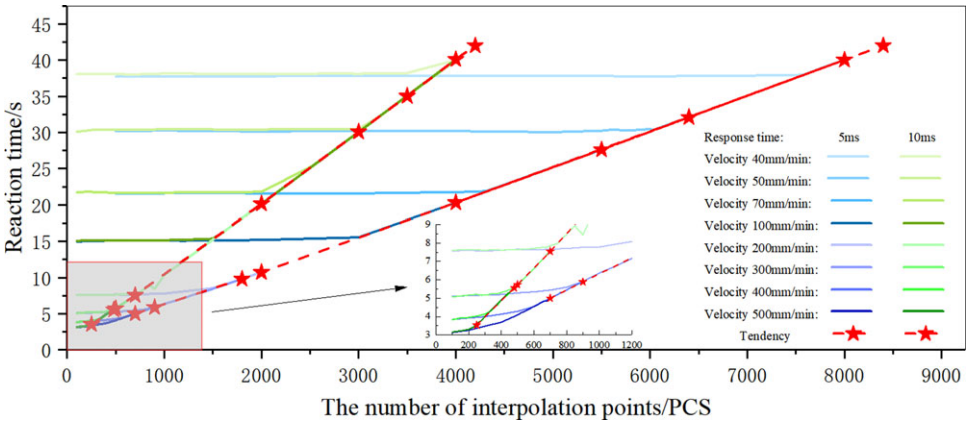


Figure 8. *The relationship between the number of interpolation points and the reaction time for different response times.*

(4) The number of limit interpolation points at different speeds is approximately fitted to a straight line, so that the number of limit interpolation points presents a linear distribution.

(5) As the control system's response time increases, the linear slope of the limit interpolation point distribution becomes steeper.

4.3 Error analysis of theoretical and practical limit interpolation points

Based on the data organized in Section 4.2, the reaction time and limit interpolation points from the experimental data were extracted, and the theoretical reaction time and limit interpolation points were calculated using Eq. (12). After performing the calculations and organizing the data, the results are shown in Table I.

The data in Table I were fitted into a graph, with the robot's terminal velocity as the independent variable and the experimental and theoretical values of the limit interpolation points under different control system's response times as the dependent variable, resulting in the line chart shown in Fig. 9. The error between the experimental and theoretical values of the limit interpolation point under different control system's response time is shown in Fig. 10.

According to the fitted curve in Fig. 9 and the error curve in Fig. 10, it is evident that when the control system's response time is 5 ms and the robot's terminal velocity is within 100 mm/min, the number of limit interpolation points can be more accurately calculated using Eq. (12). When the speed exceeds 100 mm/min, the equation does not accurately calculate the number of limit interpolation points. When

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

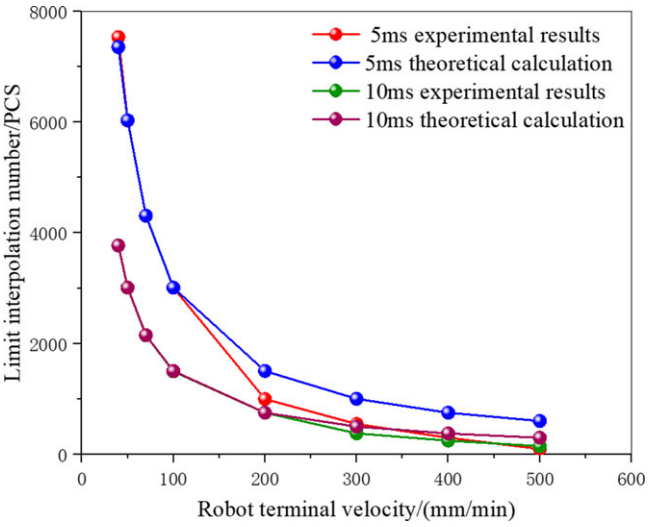


Figure 9. *Relationship between terminal velocity and limit interpolation points.*

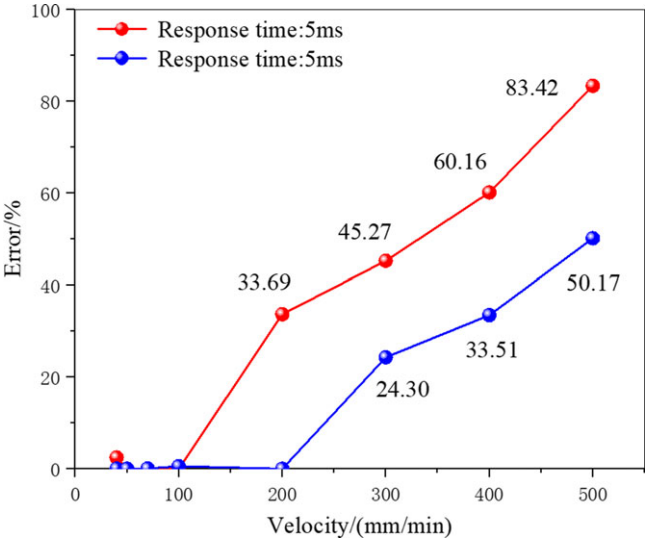


Figure 10. *Error diagram between experimental results and theoretical calculation under different response time.*

the control system's response time is 10 ms, the theoretical calculation of the robot's terminal velocity is more accurate within 200 mm/min. When the speed exceeds 200 mm/min, significant errors occur, necessitating the use of alternative prediction methods.

5. Limit interpolation point prediction model based on least square method

To more accurately predict the relationship between the control system's response time, robot's terminal velocity, and the limit interpolation points, a prediction model for limit interpolation points was developed using the least squares method, and the model's accuracy was validated.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

Table II. *Limit interpolation points and reaction time under 5 ms response time.*

Limit interpolation points/(PCS)	7539	6031	4308	3015	1000	550	300	100
Reaction time/s	37.728	30.176	21.584	15.125	7.581	5.276	3.957	3.138

Table III. *Limit interpolation points and reaction time under 10 ms response time.*

Limit interpolation points/(PCS)	3768	3015	2153	1500	753	380	250	150
Reaction time/s	38.208	30.448	21.584	15.282	8.01	5.232	4.026	3.222

5.1 Establishment of prediction model

As the same response time but different reaction time, the number of limit interpolation points is linearly distributed along a straight line. After organizing the data along the trend line, the results are presented in Tables II and 3.

According to the principles of the least square method, let the trend curve expression be $t = m + b$, then the best estimate of a and b can be expressed as:

$$\hat{a} = \frac{N\left(\sum m_i t_i\right) - \left(\sum m_i\right)\left(\sum t_i\right)}{N\left(\sum m_i^2\right) - \left(\sum m_i\right)^2} \tag{13}$$

$$\hat{b} = \frac{\left(\sum m_i^2\right)\left(\sum t_i\right) - \left(\sum m_i\right)\left(\sum m_i t_i\right)}{N\left(\sum m_i^2\right) - \left(\sum m_i\right)^2} \tag{14}$$

where t represents the reaction time, m represents the number of limit interpolation points, and i ranges from 1 to 8.

The trend curve for a response time of 5 ms is t_1, and the trend curve for a response time of 10 ms is t_2, then,

$$t_1 = a_1 m_1 + b_1 \Rightarrow \begin{cases} a_1 = 0.0097 \\ b_1 = 1.3040 \end{cases} \Rightarrow m_1 = \frac{t_1 - 1.3040}{0.0097} \tag{15}$$

$$t_2 = a_2 m_2 + b_2 \Rightarrow \begin{cases} a_2 = 0.0046 \\ b_2 = 2.4886 \end{cases} \Rightarrow m_2 = \frac{t_2 - 2.4886}{0.0046} \tag{16}$$

Linear fitting was performed on the trend curves for response times of 5 and 10 ms, respectively, as shown in Fig. 11.

In order to facilitate engineering applications, particularly in real-time control of numerical control systems, the two fitting equations can be approximated using the linear equation from Eq. (17), considering the characteristics of the relationship between the control system's response time, the robot's terminal velocity, and the limit interpolation point.

$$m = \frac{t - 0.2548k}{0.0475/k} \tag{17}$$

where k is the response time.

According to the relationship between the optimized control system's response time, the robot's terminal velocity, and the limit interpolation point, the following conclusions can be drawn:

(1) As the control system's response time increases, the slope of the prediction line for the distribution of limit interpolation points becomes steeper, corresponding to a multiple of the response time.

(2) Within the same control system's response time, as the number of motion interpolation points increases, the reaction time also increases.

(3) As the control system's response time increases, the robot's end effector moves along the same trajectory at the same velocity with fewer interpolation points.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

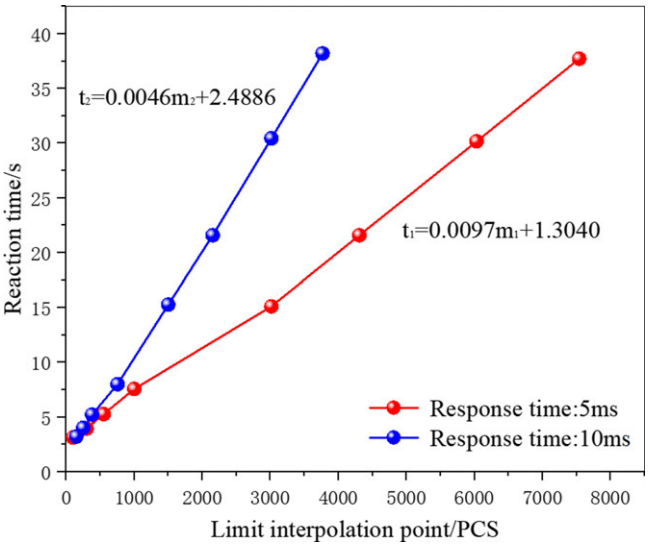


Figure 11. *Fitting curve of reaction time and limit interpolation point.*

(4) Within the same control system's response time, there is a limit interpolation point for each robot's terminal velocity. When this limit interpolation point is exceeded, the reaction time of the robot's terminal velocity becomes variable and slows down.

(5) Within the same control system's response time, the number of limit interpolation points at different velocities forms a straight line, indicating a linear distribution. Therefore, knowing the robot's terminal velocity and the reaction time of the trajectory at this velocity allows us to predict the number of limit interpolation points.

5.2 Prediction model verification

When the control system's response time is 10 ms and the robot's terminal velocity is 500mm/min, the limit interpolation points can be derived from Eq. (17) as follows:

$$m = \frac{t - 0.2548\text{k}}{0.0475/k} = \frac{3.222 - 2.548}{0.00475} = 142$$

By setting the number of motion interpolation points to 100, 300, 500, and 1500, respectively, the speed and position of the manipulator's end effector in the X and Y axes direction can be obtained, as shown in Fig. 12.

From Eq. (17), it is calculated that when the control system's response time is 10 ms and the robot's terminal velocity is 500 mm/min, the number of theoretical limit interpolation points is 142. Consequently, the speed and position curves for X and Y axes in Fig. 12(a) are relatively smooth, indicating that the robot does not exhibit jitter during the experiment. In the locally enlarged view of Fig. 12(d), it is evident that the X and Y axes velocity curves exhibit jitter, as the number of interpolation points has exceeded ten times of the theoretical limit interpolation points, leading to noticeable jitter during motion and unstable velocity. Although the number of interpolation points in Fig. 12(b) and (c) exceeds the theoretical limit interpolation points, no jitter is observed in the velocity curve due to the set motion velocity. However, the reaction time of the robot's terminal velocity in Fig. 12(b) and (c) is 4.073 and 5.552 s, respectively. Compared to the theoretical limit interpolation points, the reaction time is significantly increased, indicating that the control system's response time has exceeded the limit, which will affect the stability of the system's performance.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

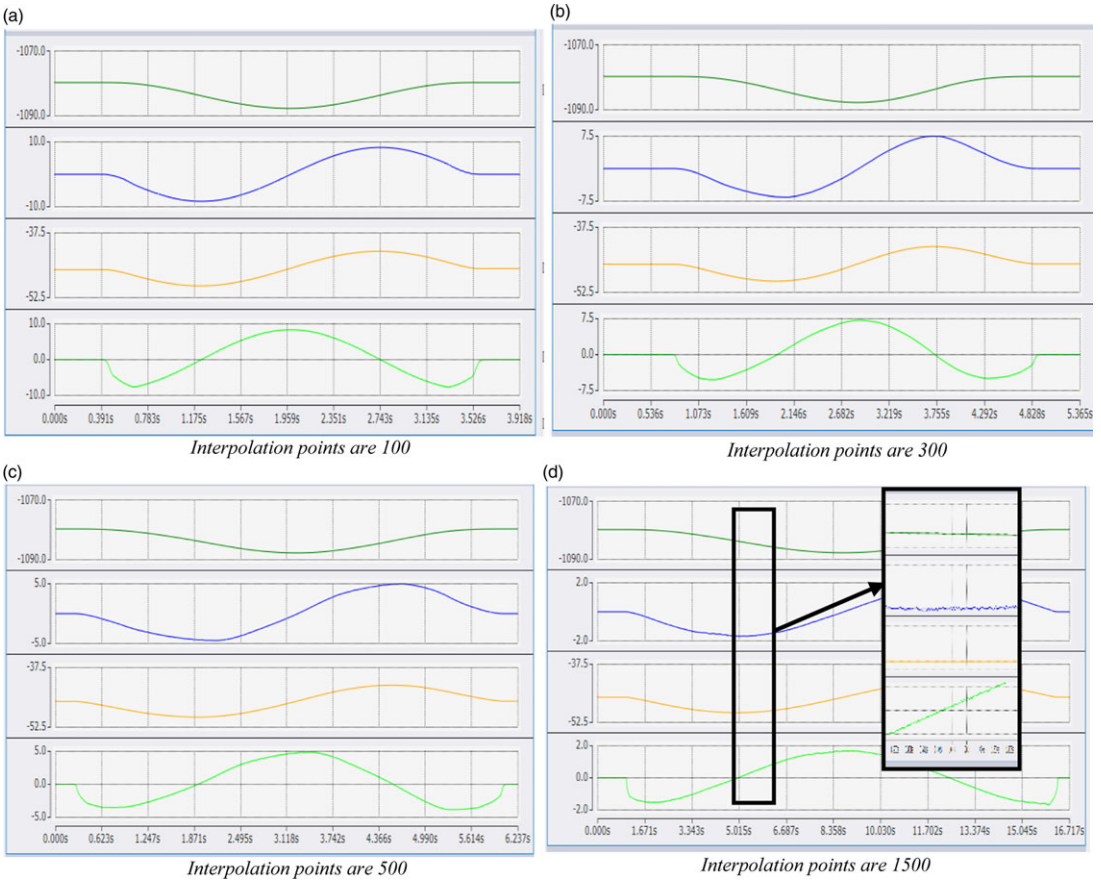


Figure 12. *Experimental data recording when the response time is 10 ms and the robot's terminal velocity is 500 mm/min.*

When examining the X axis velocity curve at 20,000 interpolation points, as shown in Fig. 13, the green area represents the overlapping portion of the velocity curve. At 20,000 interpolation points, the X axis velocity fluctuation curve exhibits violent jitter and lacks coherence, causing the control system's response time to exceed the limit, resulting in the robot's movement being out-of-step and fluctuating violently.

6. Conclusion

In this paper, the number of limit interpolation points under different control system's response time and robot's terminal velocity is analyzed both theoretically and experimentally, and the theoretical calculations are optimized based on experimental data. The results demonstrate that the fitting prediction curve for limit interpolation points accurately calculates the number of limit interpolation points. The main conclusions are as follows:

(1) Based on the principle of arc interpolation, the relationship between the number of interpolation points and arc length is analyzed, and the calculation equation for limit interpolation points is derived, providing a basis for the theoretical calculation of limit interpolation points.

(2) Through experimental analysis of different control system's response time and robot's terminal velocity, it is verified that when the control system's response time is 5 ms and the robot's terminal velocity is within 100 mm/min, the theoretical calculation can more accurately predict the number of limit interpolation points.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

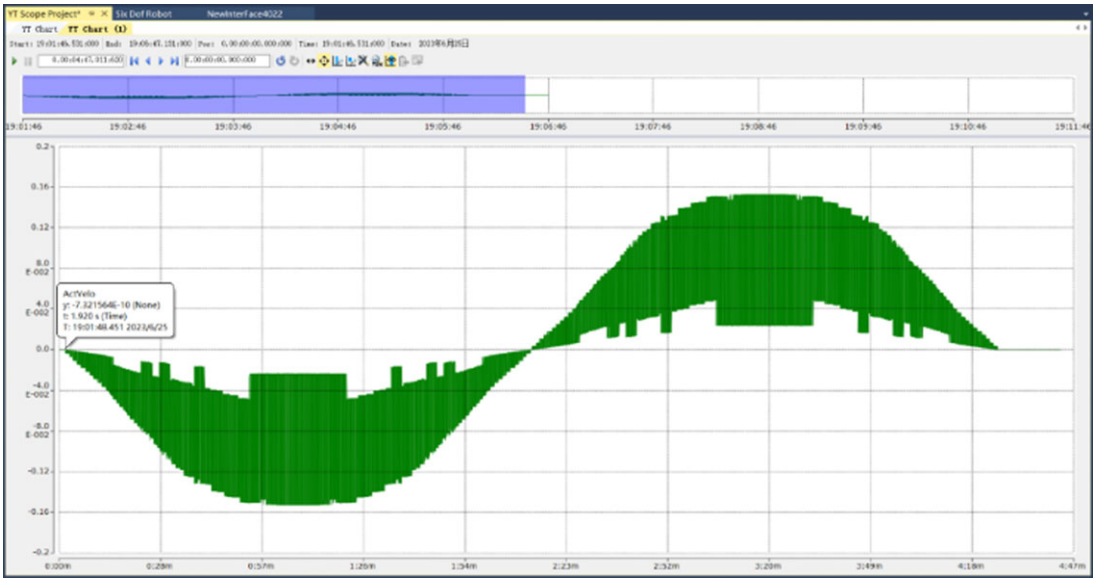


Figure 13. *X axis velocity curve when interpolation point is 20,000.*

(3) Using the least squares optimization theoretical calculation method, the prediction equation for limit interpolation points under different control system's response time and robot's terminal velocity was obtained: $m = \frac{t-0.2548k}{0.0475/k}$, and the accuracy of the prediction model was verified through experiments.

(4) Other controllers can determine their limit interpolation points using the above experimental method and deduce the prediction equation for these limit interpolation points.

This study provides a theoretical basis for selecting an appropriate number of interpolation points under different control system's response time and robot's terminal velocity, which is essential for ensuring smooth and stable robot motion during path planning. In the future, we will further study and analyze the influence of the performance of AC servo motor and the ratio of load inertia to motor inertia on the dynamic response and stability of the system.

Author contributions. Yi Wang and Guangpeng Zhang conceived and designed the study. Zhenyang Lv and Xionghui Wang conducted data gathering. Xionghui Wang performed statistical analyses. Zhenyang Lv and Guangpeng Zhang wrote the article.

Financial support. The authors would like to thank the support of Shaanxi Province key research and development plan project (NO. 2017ZDXM-GY-133).

Conflicts of interest. The authors declare no conflicts of interest exist.

Ethical approval. Not applicable.

References

[1] A. Maghami and M. Khoshdarregi, "Vision-based target localization and online error correction for high-precision robotic drilling," *Robotica* (2024). doi: 10.1017/S0263574724001255.

[2] E. S.Moghaddam, M. G. Saryazdi and A. Taghvaeipour, "Trajectory optimization of a spot-welding robot in the joint and Cartesian spaces," *Int. J. Robot. Autom.* **38**(2), 109–125 (2023). doi:10.2316/J.2023.206-0701.

[3] A. Kapsalyamov, S. Hussain, R. Goecke, N. A. T. Brown and P. K. Jamwal, "Customized stiffness control strategy for a six-bar linkage-based gait rehabilitation robot," *Robotica*, 1–18 (2024). doi:10.1017/S0263574724001425.

[4] G. Shanmugasundar, G. Sapkota, R. Čep and Kalita, K., "Application of MEREC in multi-criteria selection of optimal spray-painting robot," *Processes* **10**(6), 1172 (2022). doi: 10.3390/pr10061172.

[5] Y. Liu, S. Guo, Y. Yin, Z. Jiang and T. Liu, "Design and compliant control of a piggyback transfer robot," *J. Mech. Robot.* **14**(3), 031009 (2022). doi: 10.1115/1.4052700.

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press

[6] X. Jin, K. K. Chen, Y. Zhao, J. Ji and P. Jing, "Simulation of hydraulic transplanting robot control system based on fuzzy PID controller," *Measurement* **164**, 108023 (2020). doi: 10.1016/j.measurement.2020.108023.

[7] Y. Domae, "Recent trends in the research of industrial robots and future outlook," *J. Robot. Mechatron.* **31**(1), 57–62 (2019). doi: 10.20965/jrm.2019.p0057.

[8] C. Tepper, A. Matei, J. Zarges, S. Ulbrich and M. Weigold, "Optimal design for compliance modeling of industrial robots with Bayesian inference of stiffnesses," *Prod. Eng.* **17**(5), 643–651 (2023). doi: 10.1007/s11740-023-01198-3.

[9] A. Clark and I. Ahmad, "Touchless and nonverbal human-robot interfaces: An overview of the state-of-the-art," *Smart Health* **27**, 100365 (2023). doi:10.1016/j.smhl.2022.100365.

[10] P. Sun, Q. Liu, J. Ding and S. Pi, "Open CNC System Design for Multiple Intelligent Functions Based on Twincat and Net Framework," *IEEE International Conference on Mechatronics and Automation (ICMA)*, Takamatsu, Japan (2017) pp. 910–915. doi:10.1109/ICMA.2017.8015937.

[11] C. J. Liang, W. McGee, C. C. Menassa and V. R. Kamat, "Real-time state synchronization between physical construction robots and process-level digital twins," *Construct. Rob.* **6**(1), 57–73 (2022). doi:10.1007/s41693-022-00068-1.

[12] Z. Feng, T. Sun and X. Tian, "Research on SCARA Robot Sorting System Based on CoDeSys," *Chinese Control and Decision Conference (CCDC)*, Nanchang, China (2019) pp. 5611–5615.

[13] A. Hijazi, J.-F. Brethé and D. Lefebvre, "Design of an XY-theta platform held by a planar manipulator with four revolute joints and evaluation of its precision performances," *Robotica* **34**(11), 2532–2545 (2016). doi:10.1017/S0263574715000193.

[14] S. Yang, S. Wang and S. Huang, "ROS-based remote control of industrial robot joystick," *J. Mech. Eng. Sci.* **237**(1), 160–169 (2023). doi: 10.1177/09544062221115549.

[15] M. Eugster, J.-P. Merlet, N. Gerig, P. C. Cattin and G. Rauter, "Miniature parallel robot with submillimeter positioning accuracy for minimally invasive laser osteotomy," *Robotica* **40**(4), 1070–1097 (2022). doi:10.1017/S0263574721000990.

[16] J. Zhao, S. Liu and J. Li, "Research and implementation of autonomous navigation for mobile robots based on SLAM algorithm under ROS," *Ah S Sens.* **22**(11), 4172 (2022). doi:10.3390/s22114172.

[17] J. Zhang, W. Meng and Y. Yin, "Design of a decentralized internet-based control system for industrial robot robotic arms," *Appl. Math. Nonlinear Sci.* **9**(1) (2024). doi: 10.2478/amns.2023.1.00100.

[18] J. Zhou, Z. Li, G. Lv and E. Liu, "Fuzzy logic-based adaptive tracking control of manipulator actuated by dc motor," *Int. J. ROB. Autom.* **36**(3), 196–203 (2021). doi:10.2316/J.2021.206-0526.

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

[20] N. Yan, G. Chen, J. Wu and G. Jin, "Trajectory tracking and compensation control of redundant manipulator based on integratedcontroller," *Int. J. Robot. Autom.* **37**(4), 362–371 (2022). doi: 10.2316/J.2022.206-0762.

[21] F. J. Mañas-Álvarez, M. Guinaldo, R. Dormido and S. Dormido, "Robotic park: Multi-agent platform for teaching control and robotics," *IEEE Access* **11**, 34899–34911 (2023). doi:10.1109/ACCESS.2023.3264508.

[22] H. Tourajizadeh and O. Gholami, "Closed loop nonlinear optimal control of a 3PRS parallel robot," *Int. J. Robot. Autom.* **36**(2), 128–132 (2021). doi:10.2316/J.2021.206-0401.

[23] R. Chotikunnan, K. Roongprasert, P. Chotikunnan, P. Imura, M. Sangworasil and A. Srisiriwat, "Robotic arm design and control using MATLAB/Simulink," *Int. J. Membtance Sci. Technol.* **10**(3), 2448–2459. doi:10.15379/ijmst.v10i3.1974.

[24] J. Whitman, M. Travers and H. Choset, "Learning modular robot control policies," *IEEE Trans. Robot.* **39**(5), 4095–4113 (2023). doi:10.1109/TRO.2023.3284362.

[25] W. L. Xiao, K. Y. Zhang, S. P.Wang, J. H. Xiao, H. W. Xing, R. P. Li and G. Zhao, "STEP-NC enabled edge-cloud collaborative manufacturing system for compliant CNC machining," *J. Manuf. Syst.* **72**, 460–474 (2024). doi:10.1016/j.jmsy.2023.12.005.

[26] H. Zu, M. Zhao, Z. Chen, J. He and J. Zhou, "Measurement and Analysis of End Jitter of Six-Axis Industrial Robots," *International Conference on Intelligent Robotics and Applications (ICRIA)* (Springer Nature Singapore, Singapore, 2023) pp. 101–109. doi:10.1007/978-981-99-6504-5_9.

[27] J. Li, H. Ma, X. Zhang, X. Wu, Z. Chen, X. Li, N. Li and Z. Cheng, "Measurement and Application of Industrial Robot Jitter," *International Conference on Intelligent Robotics and Applications(ICRIA)* (Springer Nature Singapore, Singapore, 2023) pp. 371–380.

[28] R. Li, N. Ding, Y. Zhao and H. Liu, "Real-time trajectory position error compensation technology of industrial robot," *Measurement* **208**, 112418 (2023). doi:10.1016/j.measurement.2022.112418

[29] N. Hughes, Y. Chang, S. Hu, R. Talak, R. Abdulhai, J. Strader and L. Carlone, "Foundations of spatial perception for robotics: Hierarchical representations and real-time systems," *Ind. Robot.* 02783649241229725 (2024). doi:10.1177/0278364924122972.

[30] A. Laurenzi, D. Antonucci, N. G. Tsagarakis and L. Muratore, "The xbot2 real-time middleware for robotics," *Robot. Auton. Syst.* **163**, 104379 (2023). doi:10.1016/j.robot.2023.104379.

Cite this article: Z. Lv, Y. Wang, G. Zhang and X. Wang (2025). "Determination method of limit interpolation points in industrial robot control system", Robotica **43**, 1533–1547. https://doi.org/10.1017/S0263574725000359

https://doi.org/10.1017/S0263574725000359 Published online by Cambridge University Press