Robotica (2025), **43**, pp. 4212–4235
doi:10.1017/S0263574725102749

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Trajectory planning and tracking control for a robotic additive manufacturing system based on binocular vision guidance

Delan Wei[1], Pengcheng Li[1], Jun Xiong[2], Yansheng Cao[3], Xuewen Wei[1], Wei Tian[1] and Wenhe Liao[1]

[1]College of Mechanical & Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China
[2]Hafei Aviation Industry Co., Ltd, Harbin, 150060, China
[3]Beijing Xinfeng Aerospace Equipment Co., Ltd, Beijing, 100854, China
Corresponding author: Pengcheng Li; Email: lpccmee@nuaa.edu.cn

Received: 3 July 2024; **Revised:** 16 September 2025; **Accepted:** 27 September 2025;
First published online: 14 November 2025

Keywords: robot control; trajectory planning; additive manufacturing; visual servoing; tracking accuracy

Abstract

Robotic manufacturing systems offer significant advantages, including increased flexibility and reduced costs. However, challenges in trajectory planning, error compensation, and system integration hinder their broader application in additive manufacturing. To address these issues, this paper proposes a generalized non-parametric trajectory planning method tailored for robotic additive manufacturing. The proposed trajectory planner incorporates chord error and speed continuity constraints and integrates the look-ahead planning with real-time interpolation in a parallel structure to ensure smooth transitions in the robot's trajectory. Additionally, a real-time path tracking control method is introduced, combining RBF neural network-based dynamic feedforward control with visual servoing-based feedback control. This control strategy significantly improves the robot's tracking accuracy, particularly for complex additive manufacturing paths that involve multiple short connected line segments and frequent speed variations. The effectiveness of the proposed methods is validated through experiments on a robotic additive manufacturing platform. The experimental results (line segment, circular arc segment, and continuous path tracking) show that the robot's tracking error remains within ± 0.15 mm and $\pm 0.05^{\circ}$.

1. Introduction

Industrial robots are widely used in welding, spraying, handling, and automated production lines due to their stability, rapid movement, low cost, and ability to operate in environments unsafe for humans [1]. In recent years, advances in additive manufacturing and robotics have spurred their adoption across various industries, fundamentally reshaping product manufacturing. The combination of these technologies enables multi-axis additive manufacturing and the fabrication of complex geometries. A significant limitation of traditional additive manufacturing systems, such as the three-axis gantry system used for layer-by-layer printing, is their constrained working envelope [2, 3]. In contrast, industrial robots perform printing within large workspace areas. Although robotic systems may sacrifice some accuracy compared to three-axis systems, they are ideal for manufacturing large parts and integrating multiple techniques in a single workspace [4].

The RAMLAB laboratory at the Port of Rotterdam, Netherlands, combined arc additive manufacturing and subtractive manufacturing to print a marine propeller with a diameter of 1.3 m and a mass of 400 kg [5]. Song et al. [6] developed a mobile manipulator platform for concrete 3D printing that manufactures products through layer-by-layer deposition. Poudel et al. [7] introduced cooperative 3D printing, a concept wherein multiple mobile 3D printing robots collaborate to produce a single part.

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

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

However, the aforementioned industrial robots in Fused Deposition Modeling (FDM) additive manufacturing are typically employed either for applications with relatively low precision requirements or in hybrid manufacturing systems that combine additive and subtractive processes.

The quality of final products in additive manufacturing is primarily related to the machine accuracy. Several factors influence the accuracy of additive manufacturing, including (1) robot motion accuracy, which is affected by kinematic model errors and nozzle calibration. (2) The non-parametric nature of the additive manufacturing path, which consists of continuous short line segments and tens of thousands of path points. (3) The high nonlinear dynamic model and uncertainties cause difficulties in designing high-performance controllers. (4) Additive manufacturing process parameters, such as processing technology and environmental control. (5) Mechanical performance of robot equipment. This paper mainly addresses the first three factors.

To address these challenges, several research efforts have focused on accuracy control through both offline and online compensation methods. Offline compensation typically involves identifying parameters [8, 9] and modeling error sources [10, 11]. Online compensation methods, utilizing external measuring and sensing equipment for feedback, are more commonly employed. These methods include secondary grating feedback for joint space [12, 13] and visual servoing feedback for Cartesian space [14, 15]. It is worth noting that the secondary grating feedback, as a semi-closed-loop method, requires modifications and installation of a grating scale. In contrast, visual feedback provides direct measurements of the end-effector's position and orientation, which represents a fully closed-loop approach. Zhao et al. [16] designed the robotic riveting system with visual servoing and proposed a control scheme with varying parameter-linear sliding surfaces and second-order sliding surfaces to ensure accurate rivet positioning and robust path-following. Shu et al. [17] introduced a dynamic path-tracking method for industrial robots based on position visual servoing and an adaptive Kalman filter, improving tracking accuracy to ± 0.2 mm. Li et al. [18] proposed an attitude measurement method for robot end-effectors using inertial technology with the fourth-order Runge-Kutta algorithm, achieving an attitude accuracy reach 0.15°. In visual system processing, Ma et al. [19] developed a calibration method for vision system parameters, employing a minimal depth-dependent distortion model to reduce lens distortion and enhance measurement precision. Jiang et al. [20] focused on calibrating the coordinate system within the binocular vision field, creating a method that combines both global and local vision systems for transformation matrix calibration.

Current research on visual servoing mainly focuses on simple straight or arc trajectories. However, additive manufacturing paths, typically generated by Computer-Aided Manufacturing (CAM) software, are composed of numerous short linear segments, making it difficult to define trajectories mathematically. It becomes challenging to obtain accurate target points in robot motion control, and pauses at corners affect the effectiveness of visual servoing. Overfill and underfill of material at the corners negatively impact the geometrical quality of products manufactured by FDM 3D printing. Ref. [21] indicates that the printing path influences the shape accuracy of printed parts directly. To alleviate this issue, the implementation of corner smoothing for the trajectory is deemed essential.

To achieve corner smoothness in continuous linear paths, transition methods known as corner smooth methods are proposed, which typically utilize parametric curves of circular arcs to blend adjacent linear segments [22, 23]. Zhang et al. [24] introduced a real-time look-ahead algorithm to improve precision and minimize damage. The path-smoothing method inserts a quintic Bezier curve between small adjacent line segments to achieve G^2 continuity at junctions. To achieve better shape accuracy in 3D printing, Communal et al. [25] developed a path-planning strategy that considers the shape accuracy of the corners in each layer of additive manufacturing. Liu et al. [26] proposed a composite path planning method that incorporates a sharp corner correction strategy to improve the shape accuracy of parts. Zapata et al. [27] introduced a pixel-based model predicting overfill based on trajectory and traverse speed, controlling wire speed to mitigate corner overflow. Nevertheless, these methods have been primarily studied in machine tools, with fewer studies focusing on the integration into robotic systems. Zhu et al. [28] proposed a smooth trajectory-planning method combining Cartesian and joint space (using a fifth-order B-spline curve in joint space) to enhance smoothness and reduce vibrations. However, this

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

planning may lead to deviations from the original Cartesian trajectory, which is not permitted in additive manufacturing.

The research outlined above has made notable contributions in the areas of robot motion tracking compensation and trajectory planning. However, there remains an urgent need to further enhance accuracy, particularly for intricate paths in robotic additive manufacturing. This paper aims to build on prior work by designing a generalized non-parametric trajectory planning method that integrates corner-smoothing and real-time interpolation in parallel. By applying chord error and speed continuity as constraints, the proposed method reduces corner overflow and enhances shape accuracy in additive manufacturing. Additionally, the paper introduces a novel robot error compensation controller that combines RBF neural network-based dynamic feedforward control with visual servoing-based feedback control. The controller incorporates variable gain parameters to adapt to speed variations in the additive manufacturing process, along with a velocity observer to reduce measurement noise, enhancing the robot's dynamic response and trajectory tracking accuracy. Contributions of this paper are summarized as follows.

(1) A generalized non-parametric trajectory planning method is developed, integrating chord error and speed continuity as constraints. By combining the dynamic look-ahead planning with real-time interpolation in a parallel structure, it ensures smooth trajectory transitions and speed continuity, effectively balancing precision and quality.

(2) An error control method based on visual servoing feedback and RBF neural networks dynamic feedforward is proposed. This method enables the online prediction of dynamic parameters and employs a variable gain controller to address frequent velocity changes in the continuous short-line trajectory of the additive manufacturing process, significantly enhancing the trajectory accuracy of the robot.

(3) An open and compatible robotic additive manufacturing platform is developed to address challenges in system integration. The host computer is responsible for NC code parsing, task scheduling, high-level command generation, and non-real-time state monitoring. The PLC-based real-time controller handles low-level motion control, including dynamics modeling, trajectory interpolation, and real-time execution of control algorithms. All components are integrated into one system, facilitating a collaborative system and providing a new solution for robotic manufacturing system integration.

The remainder of this paper is structured as follows. Section 2 introduces the primary methods. Section 3 discusses the experiments and results. The conclusion is provided in Section 4.

2. Methods for trajectory planning and tracking control

2.1. Establishment of the workspace frame

As illustrated in Figure 1, a robotic additive manufacturing system is proposed, consisting of an industrial robot equipped with a printing end-effector, a rotating platform, and a binocular vision instrument (BVI). The robot is installed upside down on the gantry for additive manufacturing. Within the BVI coordinate system, the camera frame Σc is defined with its origin located at the midpoint between the two cameras and the z-axis aligned with the optical axis of the cameras. The coordinate frames of the robot base (Σb), robot wrist (Σw), and turning platform (ΣW) are also established within the BVI software environment. The tool frame Σt is defined such that the x-axis is aligned with the direction connecting the two nozzles, and the z-axis is aligned with the wrist frame; the y-axis is determined by the right-hand rule. The relationship between the tool frame Σt on the end-effector and the wrist frame Σw can be obtained, as shown in Figure 2. The reflectors are mounted on the robot base, end-effector, and turntable, respectively, and are tracked by the BVI system. As a position-based visual feedback device, the BVI provides real-time pose information of the tracked objects using three or more non-collinear

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

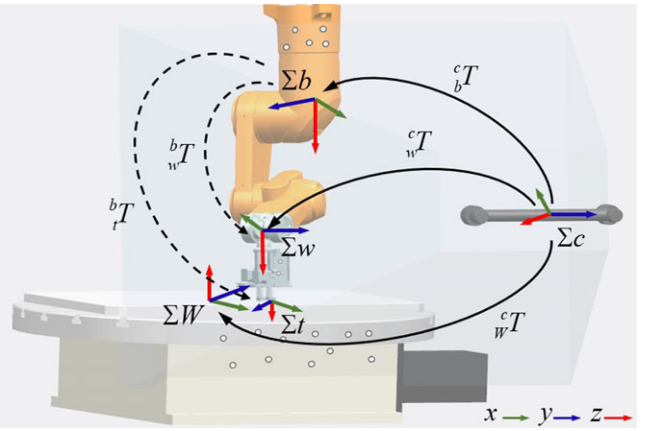


Figure 1. *Robotic additive manufacturing system and space transformation of each frame.*

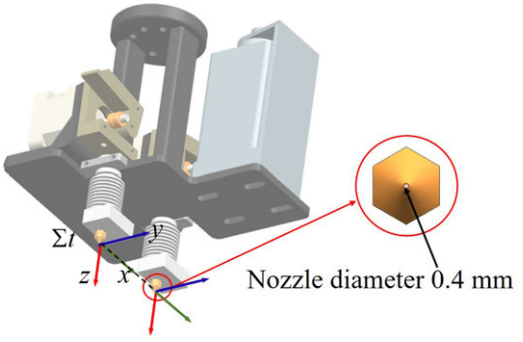


Figure 2. *Schematic diagram of the tool frame.*

feature points. It is important to note that during any movement, all corresponding frames must remain within the field of view of the BVI.

Using VXelements software and the BVI (C-track 780) from Creamform, the positional and rotational information of the tracked model relative to the camera frame Σc can be acquired, recorded, and displayed simultaneously. Therefore, the homogeneous transform matrix ${}^{c}_{b}\boldsymbol{T}$ from Σb to Σc, ${}^{c}_{w}\boldsymbol{T}$ from Σw to Σc, and ${}^{c}_{W}\boldsymbol{T}$ from ΣW to Σc can be obtained at any time. The homogeneous transformation matrix ${}^{b}_{w}\boldsymbol{T}$, ${}^{b}_{W}\boldsymbol{T}$ and ${}^{W}_{w}\boldsymbol{T}$ can be obtained as follows

$$\begin{aligned} {}^{b}_{w}\boldsymbol{T} &= {}^{b}_{c}\boldsymbol{T}\,{}^{c}_{w}\boldsymbol{T} = ({}^{c}_{b}\boldsymbol{T})^{-1}\,{}^{c}_{w}\boldsymbol{T} \\ {}^{b}_{W}\boldsymbol{T} &= {}^{b}_{c}\boldsymbol{T}\,{}^{c}_{W}\boldsymbol{T} = ({}^{c}_{b}\boldsymbol{T})^{-1}\,{}^{c}_{W}\boldsymbol{T} \\ {}^{W}_{w}\boldsymbol{T} &= {}^{W}_{c}\boldsymbol{T}\,{}^{c}_{w}\boldsymbol{T} = ({}^{c}_{W}\boldsymbol{T})^{-1}\,{}^{c}_{w}\boldsymbol{T} \end{aligned} \tag{1}$$

Suppose n feature points on the rigid end-effector are fixed and known from the definition of the tool frame Σw, whose homogeneous coordinates ${}^{w}\boldsymbol{P}_i = \begin{bmatrix} {}^{w}x_i & {}^{w}y_i & {}^{w}z_i & 1 \end{bmatrix}$ are denoted. With the pose $(x_t, y_t, z_t, \alpha_t, \beta_t, \gamma_t)$ of Σw in Σc, the homogeneous transformation matrix ${}^{c}_{w}\boldsymbol{T}$ can be formulated as follows

$${}^{c}_{w}\boldsymbol{T} = \begin{bmatrix} R(\alpha_t, \beta_t, \gamma_t) & (x_t, y_t, z_t)^T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} c\alpha_t c\beta_t & c\alpha_t s\beta_t s\gamma_t - s\alpha_t c\gamma_t & c\alpha_t s\beta_t c\gamma_t + s\alpha_t s\gamma_t & x_t \\ s\alpha_t c\beta_t & s\alpha_t s\beta_t s\gamma_t + c\alpha_t c\gamma_t & s\alpha_t s\beta_t c\gamma_t - c\alpha_t s\gamma_t & y_t \\ -s\beta_t & c\beta_t s\gamma_t & c\beta_t c\gamma_t & z_t \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

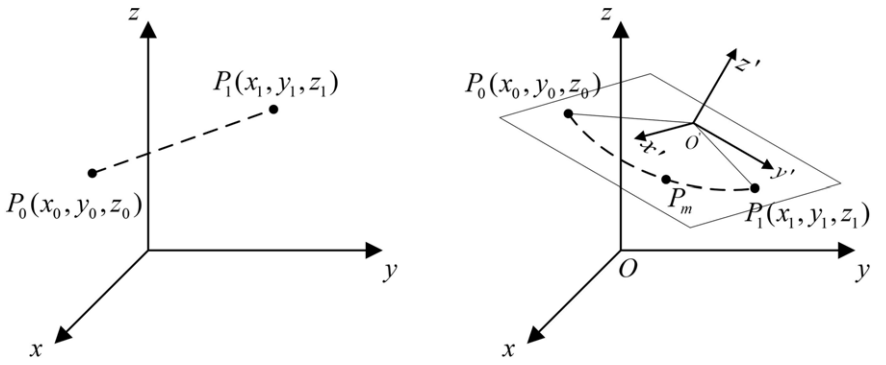


Figure 3. *Diagram of linear and circular interpolation.*

where $R(\alpha_t, \beta_t, \gamma_t)$ is the rotation matrix from Σw in Σc and is taken from the ZYX-form Euler angle, s and c are the abbreviated forms of the functions *sin* and *cos*, respectively. Therefore, for the i th feature point, the transformation equation as

$$^{c}\boldsymbol{P}_i = {}^{c}_{w}\boldsymbol{T}^{w}\boldsymbol{P}_i \tag{3}$$

The pose $^{c}\boldsymbol{P}_i$ of the manipulator can be obtained, with computations carried out in VXelements.

2.2. Trajectory planning algorithm for continuous path

The printing path is determined by the geometric shape of the additive manufacturing part, which consists of contour and infill paths. The slicing software divides the model into layers and generates cross-sectional paths based on the contours. For the printing plane with a spatial or inclined angle, the slicing software generates the infill path by setting the angle between the slicing plane and the printing base plane. A local interpolation frame is established in the slicing plane, and the coordinate transformation relationship between it and the printing base plane is obtained. In the local interpolation frame, the printing path is discretized into a series of planar free-form paths, which consist of straight lines and arcs. Therefore, trajectory interpolation for these straight lines and arcs is a critical aspect of robotic motion. In this paper, the trapezoidal velocity curve interpolation algorithm is employed for linear trajectory planning. This algorithm ensures smooth robotic movement, avoiding sudden speed changes that could cause shaking. For circular arc interpolation, a constant velocity interpolation method is used. Ensuring a smooth transition in velocity between the end of the linear segment and the circular arc. A schematic of linear and circular interpolation is shown in Figure 3.

The NC code contains essential information such as robot pose, speed and acceleration limits, nozzle selection, and filament feed rate. When moving in continuous linear no-arc transition mode (the stop-at-turn motion), the robot is required to decelerate to a complete stop before changing direction and then re-accelerate afterward, which significantly increases the printing time [22]. Meanwhile, due to the filament material overflow caused by the nozzle pressure, the quality of the additive manufacturing products is further compromised when the robot stops at the corner. To enhance the speed continuity and print quality, the trajectory planning method for continuous linear arc transition is proposed. By limiting the maximum normal acceleration of small arcs, the speed at the corners can be restricted, and the accuracy effect is considered. The method results in smaller shape deviations and enables faster printing compared to the continuous linear no-arc transition method, thereby achieving a balance between print quality and manufacturing time.

For continuous short straight paths, a register is used to store the path information from the NC code for look-ahead planning, as illustrated in Figure 4. The register is first initialized, and the point data is sequentially stored from the $n-th$ container to the first container. The top of the register is then accessed to enable continuous linear arc transition planning, with the container data moving forward sequentially.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

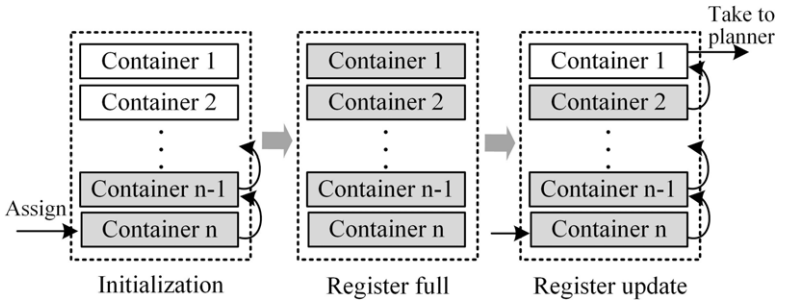


Figure 4. *The design for the register.*

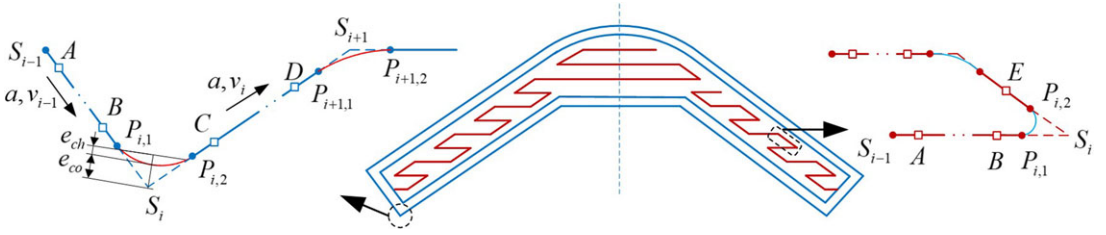


Figure 5. *Continuous linear corners with arc transition.*

After completing a segment of continuous path motion, a request is sent to the host computer system to retrieve a new set of data, thereby facilitating the update to the register.

For trajectory planning around corners, the robot's velocity must transition smoothly at corner points to minimize speed fluctuations and improve motion stability. As discussed in Ref. [26], adjusting the robot's speed to maintain continuity and smoothness through arc corners minimizes overflow and improves part quality. As shown in Figure 5, the additive manufacturing path employs smooth transitions using arcs, where the distance between the arc and the corner point is defined as the contour error e_{co}. The deviation between the actual and the theoretical chord is termed the chord error e_{ch}. Both errors are kept within acceptable limits. At the initial transition radius of the path at the $i-th$ corner point and the $(i+1)-th$ corner point, the radius is defined as follows

$$\begin{aligned} r_i &= \sin(\theta_i/2) \cdot (e_{co} - e_{ch})/(1 - \sin(\theta_i/2)) \\ r_{i+1} &= \sin(\theta_{i+1}/2) \cdot (e_{co} - e_{ch})/(1 - \sin(\theta_{i+1}/2)) \end{aligned} \tag{4}$$

where θ_i represents the angle between the two lines at the $i-th$ corner point. The projection lengths of the two arc segments on $\overrightarrow{S_iS_{i+1}}$ are

$$\begin{aligned} L_{S_iP_{i,2}} &= r_i/\tan(\theta_i/2) \\ L_{S_{i+1}P_{(i+1),1}} &= r_{i+1}/\tan(\theta_{i+1}/2) \end{aligned} \tag{5}$$

If $L_{S_iS_{i+1}} < L_{S_iP_{i,2}} + L_{S_{i+1}P_{(i+1),1}}$ the midpoint constraint method is required for initial adjustment.

$$\begin{cases} L_{S_iP_{i,2}} = L_{S_iS_{i+1}} - L_{S_{i+1}P_{(i+1),1}}, \min\left\{L_{S_iP_{i,2}}, L_{S_{i+1}P_{(i+1),1}}\right\} < 0.5 \times L_{S_iS_{i+1}} \cap L_{S_iP_{i,2}} > L_{S_{i+1}P_{(i+1),1}} \\ L_{S_{i+1}P_{i(i-1),1}} = L_{S_iS_{i+1}} - L_{S_iP_{i,2}}, \min\left\{L_{S_iP_{i,2}}, L_{S_{i-1}P_{i(i+1),1}}\right\} < 0.5 \times L_{S_iS_{i+1}} \cap L_{S_{i+1}P_{(i+1),1}} > L_{S_iP_{i,2}} \\ L_{S_iP_{i,2}} = L_{S_{i+1}P_{i(i+1),1}} = 0.5 \times L_{S_iS_{i+1}}, \min\left\{L_{S_iP_{i,2}}, L_{S_{i+1}P_{(i+1)1}}\right\} \geq 0.5 \times L_{S_iS_{i+1}} \end{cases} \tag{6}$$

The corrected radius of the arc is given by

$$\begin{aligned} r_i &= L_{S_iP_{i,2}} \cdot \tan(\theta_i/2) \\ r_{i+1} &= L_{S_{i+1}P_{(i+1),1}} \cdot \tan(\theta_{i+1}/2) \end{aligned} \tag{7}$$

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

The kinematic constraint for the motion along the arc segment is $v_{ic}^2 = a \times r_i$. However, the actual arc velocity is determined by the minimum value among the maximum allowable velocity, the maximum velocity under the chord error constraint, and the system's rated maximum velocity, which is given by

$$v_c = \min(v_{ic}, v_{e_ch}, v) \tag{8}$$

where $v_{e_ch} = 2/t \times \sqrt{2r \times e_{ch} - e_{ch}^2}$, t is interpolation period.

Furthermore, it is necessary to determine whether the linear trajectory can meet the velocity conditions. Specifically, this involves verifying whether the endpoint velocity of the previous arc can accelerate or decelerate to the starting velocity of the next arc. If this is not feasible, adjustments are made by fixing the arc radius and corner velocity on one side while modifying the arc radius and velocity on the other side, in conjunction with the velocity planning of the straight-line segment.

The detailed trajectory planning process is described as follows. S_i represent the feature points in the toolpath trajectory, which are obtained from the NC code. A sharp corner is defined by two successive linear segments $\overrightarrow{S_{i-1}S_i}$ and $\overrightarrow{S_iS_{i+1}}$ of the toolpath, a and v represent the acceleration or deceleration and maximum speed, respectively. The small square points A, B, C, and D represent the speed change points. The robot trajectory typically follows a four-step velocity profile:

- 1. Acceleration phase to reach the maximum target cruising speed, along the segment line S_iA,
- 2. Uniform speed phase at constant speed v_i, along the segment line AB,
- 3. Deceleration phase while approaching the arc corner, along the segment line $BP_{i,1}$,
- 4. Uniform speed phase, along the curve $P_{i,1}P_{i,2}$.

Next, repeating the aforementioned steps. It is important to emphasize that if a line segment is short enough, it may bypass step 2, undergoing acceleration and direct deceleration. For example, the point E on the right of Figure 5 serves as the transition point between acceleration and direct deceleration.

In the linear interpolation, let $L_{se} = \left|P_{i,2}P_{i+1,1}\right|$ be the length of the line segment, v_s be the starting velocity, and v_e be the ending velocity of the line segment, $v_{max} = v_i$. If $L_{se} < \left|\left(v_s^2 - v_e^2\right)\right| /(2a)$, and $L_{se} < \left|\left(v_s^2 - v_e^2\right)\right| /(2a)$, then there is no constant velocity stage in the trajectory, corresponding to Figure 6(b). The maximum velocity that the trajectory can reach is

$$v_{\text{lim}} = \sqrt{L_{\text{se}} \cdot a + \frac{v_{\text{s}}^2 + v_{\text{e}}^2}{2}} \tag{9}$$

The acceleration duration t_1 and deceleration duration t_2 are respectively

$$t_1 = \frac{v_{\max} - v_{\text{s}}}{a}, t_2 = \frac{v_{\max} - v_{\text{e}}}{a} \tag{10}$$

The displacement at the interpolation time t

$$P_l(t) = \begin{cases} v_s \cdot t + \dfrac{1}{2} a \cdot t^2, & 0 \le t \le t_1 \\ L_1 + v_{\text{lim}} \cdot t - \dfrac{1}{2} a \cdot t^2, & t_1 < t \le t_1 + t_2 \end{cases} \tag{11}$$

where $L_1 = v_s \cdot t_1 + \left(a \cdot t_1^2\right)/2$.

Otherwise, the trajectory includes acceleration, uniform speed, and deceleration stages, corresponding to Figure 6(a). In this case, the acceleration duration t_1 and deceleration duration t_2 are respectively

$$t_1 = \frac{v_{\max} - v_{\text{s}}}{a}, t_2 = \frac{v_{\max} - v_{\text{e}}}{a} \tag{12}$$

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

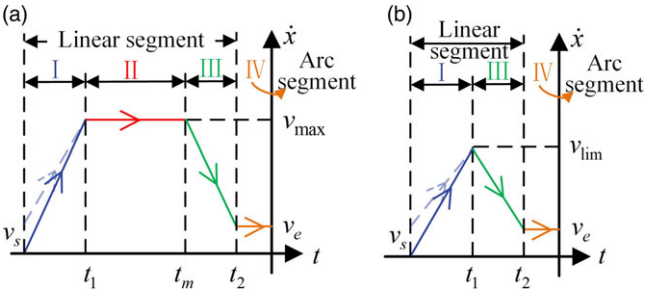


Figure 6. *Schematic diagram of the phase trajectory: (a) Long-distance positioning; (b) Short-distance positioning.*

The uniform speed duration is

$$t_{\mathrm{m}} = \frac{L_{\mathrm{se}} - \left(\frac{v_{\max}^2 - v_s^2}{2a} + \frac{v_{\max}^2 - v_e^2}{2a}\right)}{v_{\max}} \tag{13}$$

The displacement at the interpolation time t

$$P_l(t) = \begin{cases} v_s \cdot t + \dfrac{1}{2}a \cdot t^2, & 0 \le t < t_1 \\ L_1 + v_{\max} \cdot (t - t_1), & t_1 \le t < t_1 + t_{\mathrm{m}} \\ L_2 + v_{\max} \cdot (t - t_2) - \dfrac{1}{2}a \cdot (t - t_2), & t_1 + t_{\mathrm{m}} \le t \le t_1 + t_2 + t_{\mathrm{m}} \end{cases} \tag{14}$$

where $L_2 = L_1 + v_{\max} \cdot t_{\mathrm{m}}$.

For circular arc interpolation, a uniform speed planning method is employed to derive the expressions of arc length and central angle as functions of time

$$\begin{aligned} l(t) &= v_e \cdot t \\ \theta(t) &= \theta \cdot l(t)/S \end{aligned} \tag{15}$$

where S represents the arc length, θ denotes the central angle corresponding to the arc length. The interpolation points in the local coordinate system are expressed as a function of time

$$P_c(t) = [R \cdot \cos(\theta(t)), R \cdot \sin(\theta(t))] \tag{16}$$

Finally, the interpolation points are transformed from the local coordinate system to the printing base plane by the transformation matrix ${}_b^l T$, yielding the interpolation points in the printing base plane is

$$\begin{cases} P(t) = {}_b^l T \cdot P_l(t) \text{ for line} \\ P(t) = {}_b^l T \cdot P_c(t) \text{ for arc} \end{cases} \tag{17}$$

Setting the velocity of the filament feeding motor to match the velocity of the robot's end-effector, the velocity of the filament feeding motor v_{fm} is

$$v_{fm} = \frac{v_a}{v} \cdot v_m \tag{18}$$

where v is the maximum velocity for the robot in the NC code, v_a is the actual speed of the robot's end-effector, and v_m is the rated speed of the feeding motor in the robot's maximal motion velocity.

The execution logic of the planning algorithm is illustrated in Figure 7, encompassing NC code parsing, register storage, trajectory planning, and cross-interpolation of linear and circular trajectories. The sequential execution is indicated by the black arrow line, while parallel execution is represented by the red and green arrows. Notably, the red line signifies the arc interpolation input assignment, and the subsequent trajectory planning is executed in parallel with linear interpolation. The green line denotes the linear interpolation input assignment and the arc interpolation in parallel. Linear and arc interpolation are executed alternately. The pseudo-code for continuous linear arc transition is provided in Algorithm 1.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

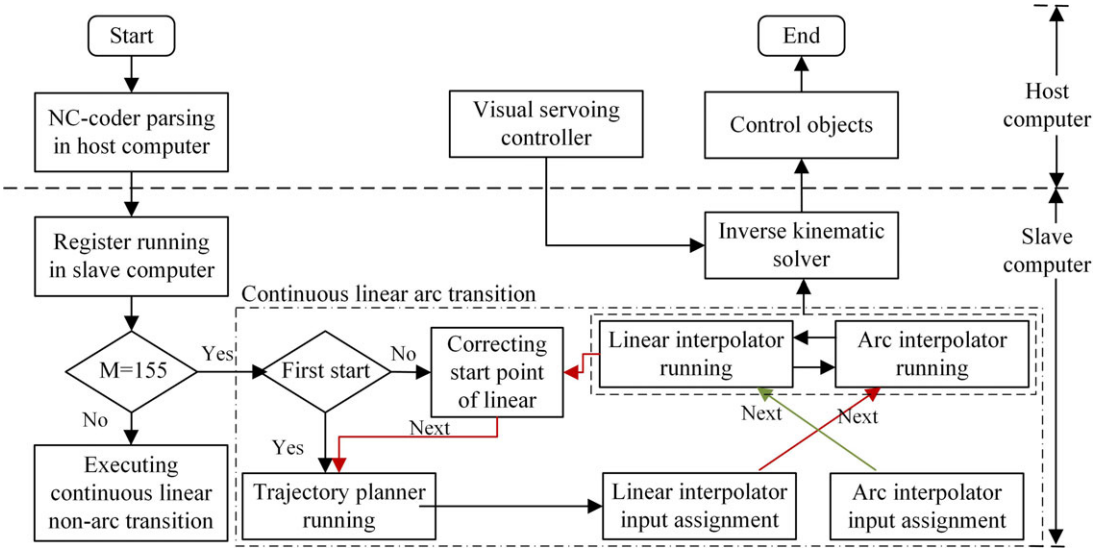


Figure 7. *Control flow chart for execution logic.*

In this algorithm, "$M = 155$" represents the continuous linear arc transition mode, $\boldsymbol{C}$ represents the register container (capacity of 10), and "planner (input)" and "planner (output)" represent inputs and outputs of the trajectory planner module, respectively. Similarly, the "ikine," "line interpolator," and "arc interpolator" represent the inverse kinematic solver, linear interpolator module, and circular interpolator module, respectively. The "para" represents an array of capacity 17 used to temporarily store the output of the planner. The "para [1:7]" represents the motion parameters of a line segment (start/end point coordinates and velocity), while the "para [8:17]" represents the motion parameters of an arc segment (start/mid/end point coordinates and velocity). t_{line} and t_{arc} represent the time of linear and circular interpolation.

2.3. Tracking control based on visual servoing

In the robotic operational space, the Cartesian-space vector $\boldsymbol{x}$ is defined as the position and orientation of the end-effector. According to the robot kinematic, the vector $\boldsymbol{x}$ is dependent on the joint configuration and can be expressed as a function of joint-space displacements as

$$\boldsymbol{x} = \boldsymbol{h}(\boldsymbol{q}) \tag{19}$$

where $\boldsymbol{h}(\bullet)$ represents a nonlinear function that computes the operational space position variables from the joint space position variables. $\dot{\boldsymbol{x}}$ denotes a velocity vector in Cartesian space, which is related to the joint-space velocity vector $\dot{\boldsymbol{q}}$ by

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_m(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{20}$$

where $\boldsymbol{J}_m(\boldsymbol{q})$ is the link Jacobian matrix.

Consider a bounded continuous trajectory, which is designed far away enough from any singularity. In the joint space, the dynamics of a serial n-link rigid robot can be written as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau} \tag{21}$$

where $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{6\times6}$ denotes the positive definite inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{6\times6}$ is the centrifugal force and Coriolis force term, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^{6\times1}$ denotes the gravity term matrix, and $\boldsymbol{\tau}$ is the joint control torque matrix.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Algorithm 1. Continuous linear arc transition mode based on the register, the planner, the interpolator, and then drive the robot.

```
1: The upper computer analyzes the NC code
2: if M = 155 then
3:     C[10] ← gain from NC code
4:     j ← 10
5:     while C[1] ≠ null do
6:         for j ← 10 to 2 do
7:             C[J − 1] ← C[j]{gain data to register until full}
8:         end for
9:     end while
10:    Planer(input) ← C[1] {Runing planner to gain line and arc paprmeters}
11:    Para ← Planer(output)
12:    C[10] ← gain from NC code {update the register}
13:    i ← 10
14:    for i ←10 to 2 do
15:        C[i − 1] ← C[i]
16:    end for
17:    line interpolator(input) ← para[1:7] {interpolation and drive the robot}
18:    if t < t_line then
19:        line interpolation
20:        ikine(input) ← line interpolator(output)
21:        Robot ← ikine(output)
22:        acr interpolator(input) ← para[8:17]
23:    end if
24:    if t < t_arc then
25:        arc interpolation
26:        ikine(input) ← arc interpolator(output)
27:        Robot ← ikine(output)
28:    end if
29: end if
```

The dynamic system in Cartesian space is the following form

$$\boldsymbol{M}_x(\boldsymbol{q})\ddot{\boldsymbol{x}} + \boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{G}_x(\boldsymbol{q}) = \boldsymbol{F}_x \tag{22}$$

where $\boldsymbol{M}_x(\boldsymbol{q}) = \boldsymbol{J}_m^{-T}(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\boldsymbol{J}_m^{-1}(\boldsymbol{q})$, $\boldsymbol{C}_x(\boldsymbol{q}) = \boldsymbol{J}_m^{-T}(\boldsymbol{q})\boldsymbol{C}(\boldsymbol{q})\boldsymbol{J}_m^{-1}(\boldsymbol{q})$, $\boldsymbol{G}_x(\boldsymbol{q}) = \boldsymbol{J}_m^{-T}(\boldsymbol{q})\boldsymbol{G}(\boldsymbol{q})$, $\boldsymbol{F}_x = \boldsymbol{J}_m^{-T}(\boldsymbol{q})\boldsymbol{\tau}$.

Property 1: The centripetal-Coriolis matrix $\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})$ possesses the following property: $\boldsymbol{x}^T(\dot{\boldsymbol{M}}_x(\boldsymbol{q}) - 2\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}}))\boldsymbol{x} = 0$.

To meet the high precision requirements of additive manufacturing, it is necessary to compensate for the inherent inaccuracies of the robotic system. A visual servoing compensation strategy is established using the BVI, with measurements converted into pose information relative to the robot base frame. Feedback signals are returned to the servo compensation controller, which generates control commands to drive the robot, thereby adjusting its pose and achieving the desired position.

In the working space, $\boldsymbol{x}_d$ is the desired position in the desired trajectory, and $\dot{\boldsymbol{x}}_d$ and $\ddot{\boldsymbol{x}}_d$ represent the desired velocity and acceleration, respectively. Define the following equation

$$\begin{aligned} \Delta\boldsymbol{x} &= \boldsymbol{x}_d - \boldsymbol{x} \\ \dot{\boldsymbol{x}}_r &= \dot{\boldsymbol{x}}_d - \mathrm{T}\Delta\boldsymbol{x} \\ \boldsymbol{r} &= \dot{\boldsymbol{x}}_r - \dot{\boldsymbol{x}} = \Delta\dot{\boldsymbol{x}} + \mathrm{T}\Delta\boldsymbol{x} \end{aligned} \tag{23}$$

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

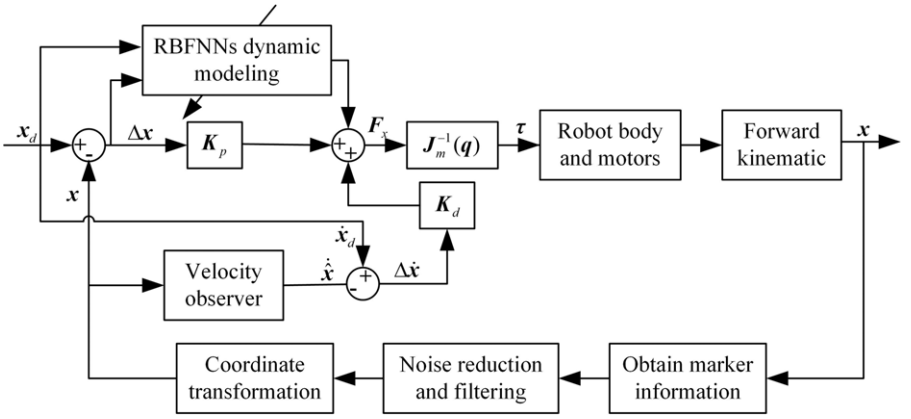


Figure 8. *The principle of the visual servoing controller.*

where, $\Delta \boldsymbol{x}$ is the pose error in Cartesian space, and $\boldsymbol{x}_d$ and $\dot{\boldsymbol{x}}_d$ are obtained by the interpolation module. In the Cartesian-space controller, the Cartesian-space feedback $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ are usually measured by the BVI, which undergoes image processing, filtering, and coordinate transformation. However, due to the presence of measurement noise, directly differentiating position values $\boldsymbol{x}$ results in significant errors in velocity $\dot{\boldsymbol{x}}$. To mitigate this issue, a velocity observer is introduced to estimate velocity using measured position data. T is a positive definite matrix.

The dynamic parameters were predicted using the RBF neural networks. The controller in Cartesian space is designed as

$$\boldsymbol{F}_x = \hat{\boldsymbol{w}}_M^T \cdot \boldsymbol{h}_M(\boldsymbol{q})\ddot{\boldsymbol{x}}_r + \hat{\boldsymbol{w}}_C^T \cdot \boldsymbol{h}_C(z)\dot{\boldsymbol{x}}_r + \hat{\boldsymbol{w}}_G^T \cdot \boldsymbol{h}_G(\boldsymbol{q}) + \boldsymbol{K}\boldsymbol{r} + k_s \operatorname{sgn}(\boldsymbol{r}) \tag{24}$$

where $(\hat{\bullet})$ is the estimation of $(\bullet)$, the first three terms of the control law are model-based control components predicted by the RBF neural networks, and the construction process is detailed in Appendix A. The $\boldsymbol{K}\boldsymbol{r}$ is the proportional derivative (PD) control term, $\boldsymbol{K}_p = \boldsymbol{K}\mathrm{T}$ and $\boldsymbol{K}_d = \boldsymbol{K}$ are diagonal and positive definite matrices, making $\mathrm{T}_{ij} = \frac{1}{K_{ij}^2}(i = j)$. The last term is a robust term to suppress the neural networks' modeling errors, and $k_s > \|\boldsymbol{E}\|$.

The control parameters $\boldsymbol{K}$ corresponding to the four stages in Figure 6 are designed as

$$\begin{cases} \boldsymbol{K}_{\boldsymbol{I}} = \bar{\boldsymbol{K}} - \dfrac{1}{2a_1}\Delta \boldsymbol{x} \cdot \boldsymbol{I} \\ \boldsymbol{K}_{\boldsymbol{II}} = \bar{\boldsymbol{K}} \\ \boldsymbol{K}_{\boldsymbol{III}} = \bar{\boldsymbol{K}} + \dfrac{1}{2a_2}\Delta \boldsymbol{x} \cdot \boldsymbol{I} \\ \boldsymbol{K}_{\boldsymbol{IV}} = \dfrac{\bar{\boldsymbol{K}}}{2} \end{cases} \tag{25}$$

where $\bar{\boldsymbol{K}}$ is the nominal value, a_1 and a_2 represent the value of acceleration and deceleration, respectively. $\boldsymbol{I}$ represents the unit matrix. In the constant acceleration phase, parameters are designed to achieve a smooth and rapid start of the motor. Once the velocity reaches its maximum, the control parameters constrain the operating speed. During deceleration, the parameters ensure a stable and accurate approach to the target position. For short-distance positioning tasks, the uniform speed phase (step 2) is omitted.

The block diagram of the visual servoing controller integrated with a velocity observer is illustrated in Figure 8. The matrix $\boldsymbol{J}_m(\boldsymbol{q})$ denotes the Jacobian matrix, and the feedback information in Cartesian space is directly transformed into the controller output of the robot in joint space using the Jacobian matrix. $\boldsymbol{\tau}$ is the torque exerted in the joint space.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Define the adaptive law of neural networks weights [29]:

$$\begin{aligned}\dot{\hat{\boldsymbol{W}}}_{Mk} &= \boldsymbol{\Gamma}_{Mk} \cdot \boldsymbol{\phi}_{Mk}(\boldsymbol{q})\ddot{\boldsymbol{x}}_r \boldsymbol{r}_k \\ \dot{\hat{\boldsymbol{W}}}_{Ck} &= \boldsymbol{\Gamma}_{Ck} \cdot \boldsymbol{\phi}_{Ck}(\boldsymbol{z})\dot{\boldsymbol{x}}_r \boldsymbol{r}_k \\ \dot{\hat{\boldsymbol{W}}}_{Gk} &= \boldsymbol{\Gamma}_{Gk} \cdot \boldsymbol{\phi}_{Gk}(\boldsymbol{q})\boldsymbol{r}_k\end{aligned} \tag{26}$$

where $\boldsymbol{\Gamma}_{MK}$, $\boldsymbol{\Gamma}_{CK}$, and $\boldsymbol{\Gamma}_{GK}$ are the symmetric positive definite matrix, and $\boldsymbol{\phi}_{Mk}(\boldsymbol{q})$, $\boldsymbol{\phi}_{Ck}(z)$, and $\boldsymbol{\phi}_{Gk}(\boldsymbol{q})$ are the Gaussian kernel functions of the neural networks, respectively.

The velocity observer is proposed to solve the velocity estimation

$$\begin{aligned}\dot{\hat{\boldsymbol{x}}} &= \boldsymbol{P} + \Lambda\tilde{\boldsymbol{x}} \\ \dot{\boldsymbol{p}} &= \boldsymbol{M}_x^{-1}(\boldsymbol{q})[\boldsymbol{F}_x - \boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\hat{\boldsymbol{x}}} - \boldsymbol{G}_x(\boldsymbol{q})]\end{aligned} \tag{27}$$

where $\boldsymbol{p}$ is an auxiliary variable, Λ is a diagonal positive definite matrix designed based on the stability requirements of the closed-loop system. The expression Λ will be presented in the stability analysis of the observer system. The dynamic coefficients are parameter values predicted using the RBF neural networks. Note that the equation is basically the robot Eq. (22) with $\dot{\boldsymbol{x}}$ replaced by $\dot{\hat{\boldsymbol{x}}}$.

Before the stability of the observer systems is analyzed, the observation error $\dot{\tilde{\boldsymbol{x}}}$ for the system is derived. By taking the time derivative of $\dot{\hat{\boldsymbol{x}}}$

$$\ddot{\hat{\boldsymbol{x}}} = \dot{\boldsymbol{p}} + \Lambda\dot{\tilde{\boldsymbol{x}}} \tag{28}$$

After pre-multiplying Eq. (28) by $\boldsymbol{M}_x(\boldsymbol{q})$, and substituting $\dot{\boldsymbol{p}}$ of Eq. (27)

$$\boldsymbol{M}_x(\boldsymbol{q})\ddot{\hat{\boldsymbol{x}}} = \boldsymbol{F}_x - \boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\hat{\boldsymbol{x}}} - \boldsymbol{G}_x(\boldsymbol{q}) + \Lambda\boldsymbol{M}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}} \tag{29}$$

Subtracting Eq. (29) from Eq. (22), we obtain the closed-loop error system for Cartesian-space velocity observation error $\dot{\tilde{\boldsymbol{x}}}$ as

$$\boldsymbol{M}_x(\boldsymbol{q})\ddot{\tilde{\boldsymbol{x}}} = -\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\tilde{\boldsymbol{x}}} - \Lambda\boldsymbol{M}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}} \tag{30}$$

Let the function V_0 be defined as follows to analyze the velocity observer stability.

$$V_0 = \frac{1}{2}\dot{\tilde{\boldsymbol{x}}}^T\boldsymbol{M}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}} \tag{31}$$

Taking the time derivative of V_0

$$\dot{V}_0 = \dot{\tilde{\boldsymbol{x}}}^T\boldsymbol{M}_x(\boldsymbol{q})\ddot{\tilde{\boldsymbol{x}}} + \frac{1}{2}\dot{\tilde{\boldsymbol{x}}}^T\dot{\boldsymbol{M}}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}} \tag{32}$$

After substituting the error system for $\dot{\tilde{\boldsymbol{x}}}$ of Eq. (30) from Eq. (32), results in

$$\begin{aligned}\dot{V}_0 &= \frac{1}{2}\dot{\tilde{\boldsymbol{x}}}^T\dot{\boldsymbol{M}}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}} + \dot{\tilde{\boldsymbol{x}}}^T(-\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\tilde{\boldsymbol{x}}} - \Lambda\boldsymbol{M}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}}) \\ &= \frac{1}{2}\dot{\tilde{\boldsymbol{x}}}^T(\dot{\boldsymbol{M}}_x(\boldsymbol{q}) - 2\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}}))\dot{\tilde{\boldsymbol{x}}} - \dot{\tilde{\boldsymbol{x}}}^T\Lambda\boldsymbol{M}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}}\end{aligned} \tag{33}$$

Applying the *property 1*, we obtain

$$\dot{V}_0 = -\dot{\tilde{\boldsymbol{x}}}^T\Lambda\boldsymbol{M}_x(\boldsymbol{q})\dot{\tilde{\boldsymbol{x}}} \le -\Lambda m_1\|\dot{\tilde{\boldsymbol{x}}}\| \le 0 \tag{34}$$

V_0 is positive definite, $\dot{V}_0$ is semi-negative definite, and the closed-loop system has progressive stabilization. When $t \to \infty$, $\dot{\tilde{\boldsymbol{x}}} \to \infty$, the observer serves as an estimator of velocity effectively.

The robot dynamics Eq. (22) is written as Eq. (35) according to Appendix A.

$$\begin{aligned}\boldsymbol{F}_x &= (\boldsymbol{F}_{MNN}(\boldsymbol{q}) + \boldsymbol{E}_M)\ddot{\boldsymbol{x}} + (\boldsymbol{F}_{CNN}(\boldsymbol{z}) + \boldsymbol{E}_C)\dot{\boldsymbol{x}} + (\boldsymbol{F}_{GNN}(\boldsymbol{q}) + \boldsymbol{E}_G) + k_s\,\mathrm{sgn}(\boldsymbol{r}) \\ &= \left[\boldsymbol{w}_M^T\boldsymbol{h}_M(\boldsymbol{q}) + \boldsymbol{E}_M\right]\ddot{\boldsymbol{x}} + \left[\boldsymbol{w}_C^T\boldsymbol{h}_C(\boldsymbol{z}) + \boldsymbol{E}_C\right]\dot{\boldsymbol{x}} + \left[\boldsymbol{w}_G^T\boldsymbol{h}_G(\boldsymbol{q}) + \boldsymbol{E}_G\right] + k_s\ \mathrm{sgn}(\boldsymbol{r})\end{aligned} \tag{35}$$

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Applying the control law from Eq. (24) to the robot dynamics in Eq. (35) and using Eq.(23) yields the tracking error equation

$$\begin{aligned}&\left[\boldsymbol{w}_M^T\boldsymbol{h}_M(\boldsymbol{q})+\boldsymbol{E}_M\right]\ddot{\boldsymbol{r}}+\left[\boldsymbol{w}_C^T\boldsymbol{h}_C(z)+\boldsymbol{E}_C\right]\dot{\boldsymbol{r}}\\&=\boldsymbol{M}\dot{\boldsymbol{r}}+\boldsymbol{C}\boldsymbol{r}=\tilde{\boldsymbol{w}}_M^T\boldsymbol{h}_M(\boldsymbol{q})\ddot{\boldsymbol{x}}_r+\tilde{\boldsymbol{w}}_C^T\boldsymbol{h}_C(z)\dot{\boldsymbol{x}}_r+\tilde{\boldsymbol{w}}_G^T\boldsymbol{h}_G(\boldsymbol{q})+\boldsymbol{E}-\boldsymbol{K}\boldsymbol{r}-k_s\,\mathrm{sgn}\,(\boldsymbol{r})\end{aligned}\tag{36}$$

where $(\tilde{\bullet})=(\bullet)-(\hat{\bullet})$ represents the estimation error of the parameter, $\boldsymbol{E}=\boldsymbol{E}_M\ddot{\boldsymbol{x}}_r+\boldsymbol{E}_C\dot{\boldsymbol{x}}_r+\boldsymbol{E}_G$.

To prove the stability of the system predicted by the RBF neural networks, the Lyapunov function is defined as

$$V_1=\frac{1}{2}\boldsymbol{r}^T\boldsymbol{M}_x\boldsymbol{r}+\frac{1}{2}\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Mk}^T\boldsymbol{\Gamma}_{Mk}^{-1}\tilde{\boldsymbol{w}}_{Mk}+\frac{1}{2}\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Ck}^T\boldsymbol{\Gamma}_{Ck}^{-1}\tilde{\boldsymbol{w}}_{Ck}+\frac{1}{2}\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Gk}^T\boldsymbol{\Gamma}_{Gk}^{-1}\tilde{\boldsymbol{w}}_{Gk}\tag{37}$$

Computing the derivative of Eq. (37) and simplifying it to get

$$\dot{V}_1=\boldsymbol{r}^T\boldsymbol{M}\dot{\boldsymbol{r}}+\frac{1}{2}\boldsymbol{r}^T\dot{\boldsymbol{M}}\boldsymbol{r}+\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Mk}^T\boldsymbol{\Gamma}_{Mk}^{-1}\dot{\tilde{\boldsymbol{w}}}_{Mk}+\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Ck}^T\boldsymbol{\Gamma}_{Ck}^{-1}\dot{\tilde{\boldsymbol{w}}}_{Ck}+\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Gk}^T\boldsymbol{\Gamma}_{Gk}^{-1}\dot{\tilde{\boldsymbol{w}}}_{Gk}\tag{38}$$

According to *property 1*, and $\dot{\tilde{\boldsymbol{w}}}_{MK}=-\dot{\hat{\boldsymbol{w}}}_{Mk}$, $\dot{\tilde{\boldsymbol{w}}}_{CK}=-\dot{\hat{\boldsymbol{w}}}_{Ck}$, $\dot{\tilde{\boldsymbol{w}}}_{GK}=-\dot{\hat{\boldsymbol{w}}}_{Gk}$. Substituted into the above formula

$$\dot{V}_1=\boldsymbol{r}^T(\boldsymbol{M}\dot{\boldsymbol{r}}+\boldsymbol{C}\boldsymbol{r})-\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Mk}^T\boldsymbol{\Gamma}_{Mk}^{-1}\dot{\hat{\boldsymbol{w}}}_{Mk}-\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Ck}^T\boldsymbol{\Gamma}_{Ck}^{-1}\dot{\hat{\boldsymbol{w}}}_{Ck}-\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Gk}^T\boldsymbol{\Gamma}_{Gk}^{-1}\dot{\hat{\boldsymbol{w}}}_{Gk}\tag{39}$$

Taking Eq. (36) into Eq. (39)

$$\begin{aligned}\dot{V}_1=&-\boldsymbol{r}^T\boldsymbol{K}\boldsymbol{r}+\boldsymbol{r}^T\boldsymbol{E}-k_s\boldsymbol{r}^T\,\mathrm{sgn}\,(\boldsymbol{r})+\boldsymbol{r}^T\tilde{\boldsymbol{w}}_M^T\cdot\boldsymbol{h}_M(\boldsymbol{q})\ddot{\boldsymbol{x}}_r+\boldsymbol{r}^T\tilde{\boldsymbol{w}}_C^T\cdot\boldsymbol{h}_C(z)\dot{\boldsymbol{x}}_r+\boldsymbol{r}^T\tilde{\boldsymbol{w}}_G^T\cdot\boldsymbol{h}_G(\boldsymbol{q})\\&-\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Mk}^T\boldsymbol{\Gamma}_{Mk}^{-1}\dot{\hat{\boldsymbol{w}}}_{Mk}-\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Ck}^T\boldsymbol{\Gamma}_{Ck}^{-1}\dot{\hat{\boldsymbol{w}}}_{Ck}-\sum_{k=1}^{n}\tilde{\boldsymbol{w}}_{Gk}^T\boldsymbol{\Gamma}_{Gk}^{-1}\dot{\hat{\boldsymbol{w}}}_{Gk}\end{aligned}\tag{40}$$

Substitute the adaptive rate Eq. (26) into the above equation and $k_s>\|\boldsymbol{E}\|$, we obtain

$$\dot{V}_1=-\boldsymbol{r}^T\boldsymbol{K}\boldsymbol{r}+\boldsymbol{r}^T\boldsymbol{E}-k_s\boldsymbol{r}^T\,\mathrm{sgn}\,(\boldsymbol{r})\le 0\tag{41}$$

In general, $V_1\ge 0$, $\dot{V}_1\le 0$, when the manipulator operates in a workspace where the kinematics mapping between Cartesian space and joint space is one-to-one, and the Jacobian matrix is nonsingular, the asymptotic stability can be concluded according to LaSalle's Invariance Theorem.

3. Experimental studies

3.1. System and control architecture

The robotic additive manufacturing platform was developed using Beckhoff industrial control software (TwinCAT), which provides an open and compatible robot control environment. By integrating a printing end-effector and a turntable, the robotic additive manufacturing system was constructed. Based on the EtherCAT bus protocol, the system is organized into three layers: the central control layer, the terminal module layer, and the controlled objects. The overall system framework and hardware compositions are depicted in Figure 9.

The central control layer comprises the master computer, the slave computer PLC (Programmable Logic Controller), and a human-computer interaction panel. The master computer is responsible for robot motion, printing planning, status collection, and hardware monitoring. The slave computer system supports the development of motion control algorithms and logic control programs. The interaction panel allows for manual debugging and automatic execution of the printing process.

The terminal module layer uses couplers and terminals to connect the central control layer with the control objects layer. The control objects include six motors of the robot, a turntable motor, and the key components of the end-effector. Motor-level control decomposes the robot into six independent motor units, which can be regulated in position, velocity, or torque mode. Additional controlled objects

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Table I. *Calibration results of the TCP.*

TCP	x_t(mm)	y_t(mm)	z_t(mm)	α_t(deg)	β_t(deg)	γ_t(deg)
Nozzle1	31.013	0.771	277.223	0.0012	0.005	0.001
Nozzle2	−30.451	0.612	277.125	0.0024	0.0022	0.006

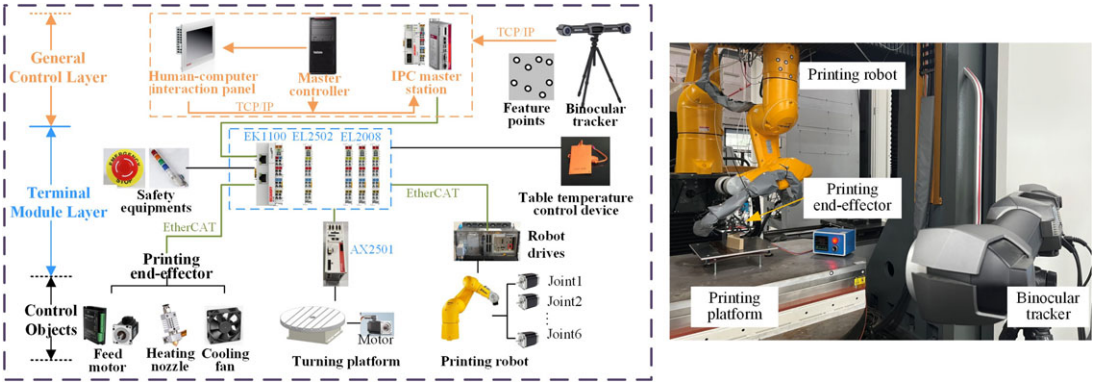


Figure 9. *The hardware configuration for the robotic additive manufacturing system.*

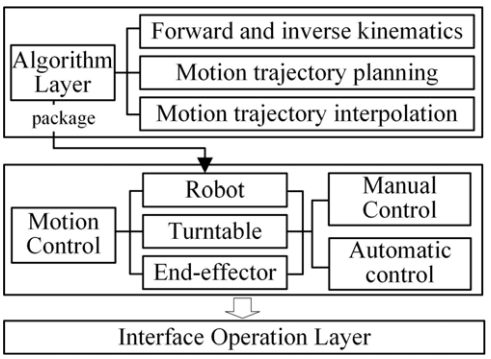


Figure 10. *Control architecture of the additive manufacturing system.*

include safety devices and a substrate heating system for the printing platform. A binocular vision system provides real-time pose measurements, which are transmitted to the master controller via TCP/IP.

The control architecture comprises the robot algorithm layer, robot motion control layer, and interface operation layer, as shown in Figure 10. The control platform features both a manual debugging mode and an automatic execution mode. The interface operation layer facilitates the switching of control modes, enabling the jogging of the robot and printing turntable and the feeding of the wire feeding motor of the printing end-effector. The algorithm layer includes forward and inverse kinematics of the robot, the trajectory planning, and the interpolation of the end-effector, developed in Matlab/Simulink and deployed to TwinCAT software.

3.2. Verification of visual servoing control

The results of the TCP in the wrist frame are shown in Table I. Nozzle 1 is located on the front, and Nozzle 2 is located on the rear side.

The performance of the proposed visual servoing controller was evaluated using a BVI (C-track 780), visual reflectors, a Staubli TX90 robot, and the developed control system. The BVI provides continuous

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Table II. *The theoretical and actual link lengths of TX90 robot.*

l_n **Lengths**	l_1	l_2	l_3	l_4	l_5	l_6
Theoretical values (mm)	478	50	425	50	425	100
Actual values (mm)	478.012	49.735	425.301	50.091	424.986	100.021

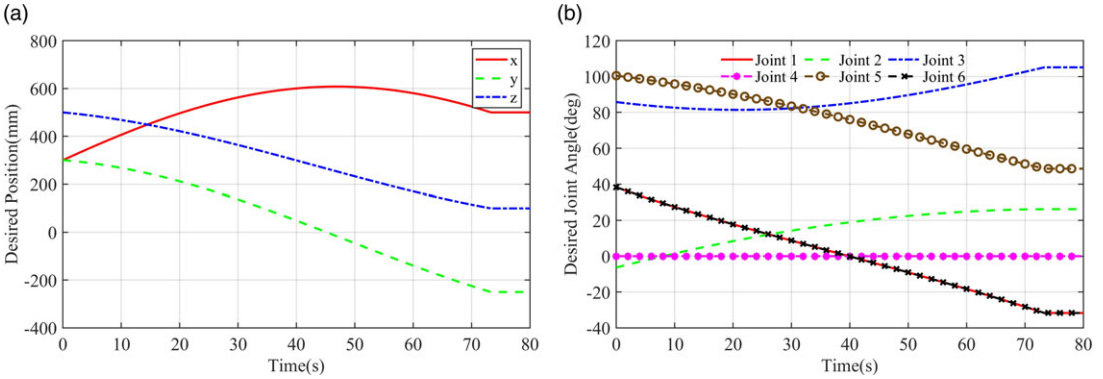


Figure 11. *Schematic diagram of the circular trajectory: (a) Interpolated curves in Cartesian space; (b) Interpolated curves in joint space.*

3D measurements of reference reflectors with a sampling period of 20 ms. Its volumetric accuracy is 0.065 mm, and repeatability is 0.0025 mm, calibrated using a reference rod and Vxelements software. The Staubli TX90 robot has a maximum working radius of 1450 mm and a payload capacity of 14 kg. Its link parameters are shown in Table II, based on the method proposed in Ref. [30], while joint limits and axis speed constraints are specified in the manufacturer's manual. Trajectory planning and control algorithms were implemented in Twincat3 on the industrial computer (C6650-0060).

To validate the trajectory accuracy, three representative motion paths were tested: arc trajectory, linear trajectory, and continuous trajectory.

The multiple gains are given by $\mathrm{T} = \mathrm{diag}\{10\}$, $\bar{\boldsymbol{K}} = \mathrm{diag}\{0.5\}$, and $k_s = 0.5$. In light of the balance between prediction accuracy and solution speed, the number of nodes in the hidden layer is designed to be 7, and the value of the Gaussian function parameter is adjusted by $\boldsymbol{C}_i = \begin{bmatrix}0.25 & 0.50 & 0.75 & 1.00 & 1.25 & 1.50 & 1.75\end{bmatrix}$, $b_i = 10$. The weight update adaptive rate parameter is taken as $\boldsymbol{\Gamma}_{MK} = \boldsymbol{\Gamma}_{GK} = \mathrm{diag}\{50\}$, $\boldsymbol{\Gamma}_{CK} = \mathrm{diag}\{2\}$.

The arc trajectory was defined with a starting point (300, 300, 500), a midpoint (600, 50, 300), and an ending point (500, −250, 100). The maximum planned tangential speed is 10 mm/s, with a tangential acceleration of 10 mm/s^2. The time evolution of Cartesian coordinates is shown in Figure 11(a), and the time evolution of the joint space is shown in Figure 11(b).

The pose tracking errors of the robot moving in an arc trajectory are displayed in Figs. 12 and 13. With visual servoing compensation, the position steady-state error is reduced to within ±0.1 mm, and the orientation error within ±0.05°. Initial pose deviations converge to zero within 6 s. Without compensation, errors are larger and more irregular. A quantitative comparison of compensated and uncompensated circular motion is summarized in Table III.

A linear trajectory was planned in the robot base frame, from (300, 300, 550) to (700, −250, 300), with a maximum velocity of 10 mm/s and an acceleration of 10 mm/s^2. Cartesian and joint evolutions are shown in Figs. 14(a) and 14(b), respectively.

The tracking errors are displayed in Figs. 15 and 16, showing position errors below ±0.1 mm and rotation errors below ±0.05°. Compared with the uncompensated case, accuracy is markedly improved. Table IV summarizes the maximum steady-state errors for both conditions.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Table III. *Comparison of motion error in the stable state for circular trajectory.*

v=10mm/s|Mix error|	No visual servoing	Visual servoing	Improvement
Distance(mm)	0.4940	0.0807	83.7%
x-error(mm)	0.4812	0.0673	86.0%
y-error(mm)	0.1982	0.0528	73.4%
z-error(mm)	0.1601	0.0468	70.7%
α-error(deg)	0.0384	0.0169	55.9%
β-error(deg)	0.1334	0.0292	78.1%
γ-error(deg)	0.0512	0.0097	81.1%

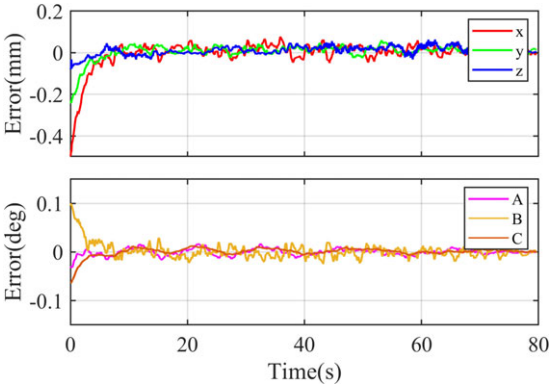


Figure 12. *Circular tracking error with visual servoing.*

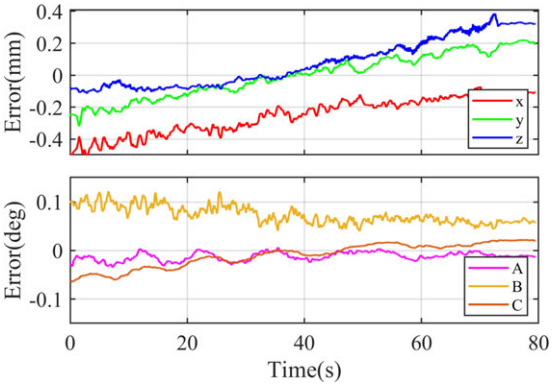


Figure 13. *Circular tracking error without visual servoing.*

To evaluate corner tracking, a continuous trajectory with sharp turns was tested. The trajectory was smoothed using circular arc transitions, with a corner tolerance of 0.05 mm. Eight straight-line segments were combined to form the letter "NU," with corner coordinates listed in Table V. The planned trajectories with and without arc transitions are illustrated in Figure 17(a). Cartesian and joint evolutions are shown in Figs. 17(b) and 17(c), and the Cartesian velocity profile in Figure 17(d). The velocity profile confirms continuity, as the end-effector did not decelerate to zero at the corners.

The corresponding tracking errors are presented in Figs. 18 and 19. With visual servoing, position errors remain below ± 0.15 mm and rotation errors below $\pm 0.05^{\circ}$. Table VI compares the errors between compensated and uncompensated trajectories.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

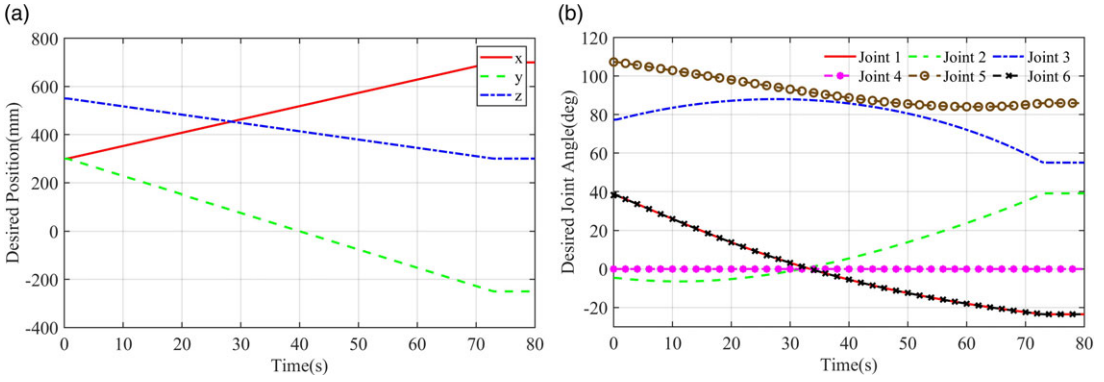


Figure 14. *Schematic diagram of the linear trajectory: (a) Interpolated curves in Cartesian space; (b) Interpolated curves in joint space.*

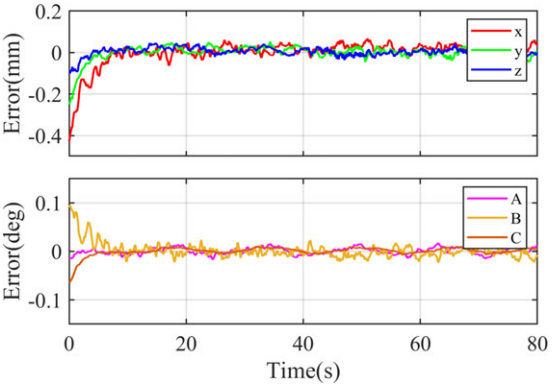


Figure 15. *Linear tracking error with visual servoing.*

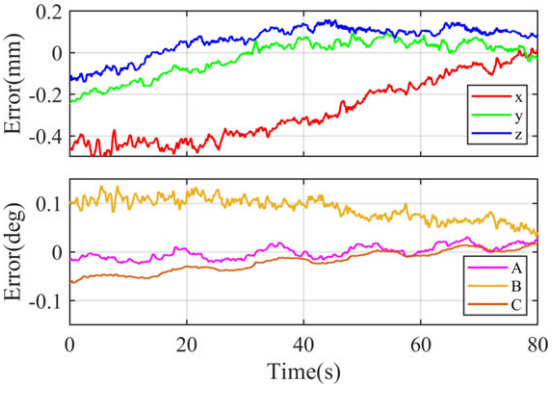


Figure 16. *Linear tracking error without visual servoing.*

After the visual servoing compensation, the trajectory accuracy of the robot was improved to 0.121 mm without load for a continuous trajectory with corners. In the FDM additive manufacturing process, the load on the robot is negligible. Therefore, the improvement of the robot accuracy achieved by the method can be directly applied to the field of robot additive manufacturing, and the effectiveness was verified by the printing experiment presented in the following section.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Table IV. *Comparison of motion error in the stable state for linear trajectory.*

v=10mm/s|Mix error|	No visual servoing	Visual servoing	Improvement
Distance(mm)	0.4868	0.0810	83.4%
x-error(mm)	0.4790	0.0772	83.8%
y-error(mm)	0.2010	0.0597	70.3%
z-error(mm)	0.3801	0.0718	81.1%
α-error(deg)	0.0281	0.0170	39.5%
β-error(deg)	0.1209	0.0295	75.6%
γ-error(deg)	0.0502	0.0124	75.3%

Table V. *The coordinate value of trajectory sharp corner points.*

points	x(mm)	y(mm)	z(mm)
P_1	300	0	400
P_2	350	160	400
P_3	420	0	400
P_4	460	160	400
P_5	460	27	400
P_6	486	0	400
P_7	520	0	400
P_8	546	27	400
P_9	546	160	400

3.3. Experiment for robotic additive manufacturing

To further validate the method, a component was designed featuring both linear and curved characteristics, where the curved features can be approximated by a series of short straight lines. The printing path of the part is shown in Figure 20. The material used was PEEK, with a melting temperature of 320–390°C. Printing parameters were as follows: layer height 0.3 mm, wire feeding speed 7.5 mm/s, robot motion speed 10 mm/s, nozzle temperature 340°C, and platform temperature 150°C.

Microscopic inspection of printed parts is shown in Figure 21. Without compensation (Figure 21(a)), corner overflow and reduced trajectory straightness are evident. With the proposed method (Figure 21(b)), trajectory deviation and overflow are significantly reduced. Dimensional accuracy was evaluated using a coordinate measuring machine, with geometric dimensions annotated in Figure 22 and measurement results summarized in Table VII. The geometric error of the printed part was within $\pm$0.15 mm.

These experiments demonstrate that visual servoing control, combined with arc transition smoothing, improves printing quality and enhances geometric accuracy. Thereby promoting the application of industrial robots in precision additive manufacturing of components.

4. Conclusion

In this article, a robotic additive manufacturing system is developed, integrating an industrial robot, a printing end-effector, and a turntable platform to enable effective collaborative control of multiple subsystems. A generalized non-parametric trajectory planning method is proposed to enhance the continuity and smoothness of robot motion, thereby enhancing the shape precision of printed components and reducing corner overflow. To address challenges related to limited NC code capacity and data transmission delays, a parallel planning architecture based on a register and planner mechanism is introduced.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

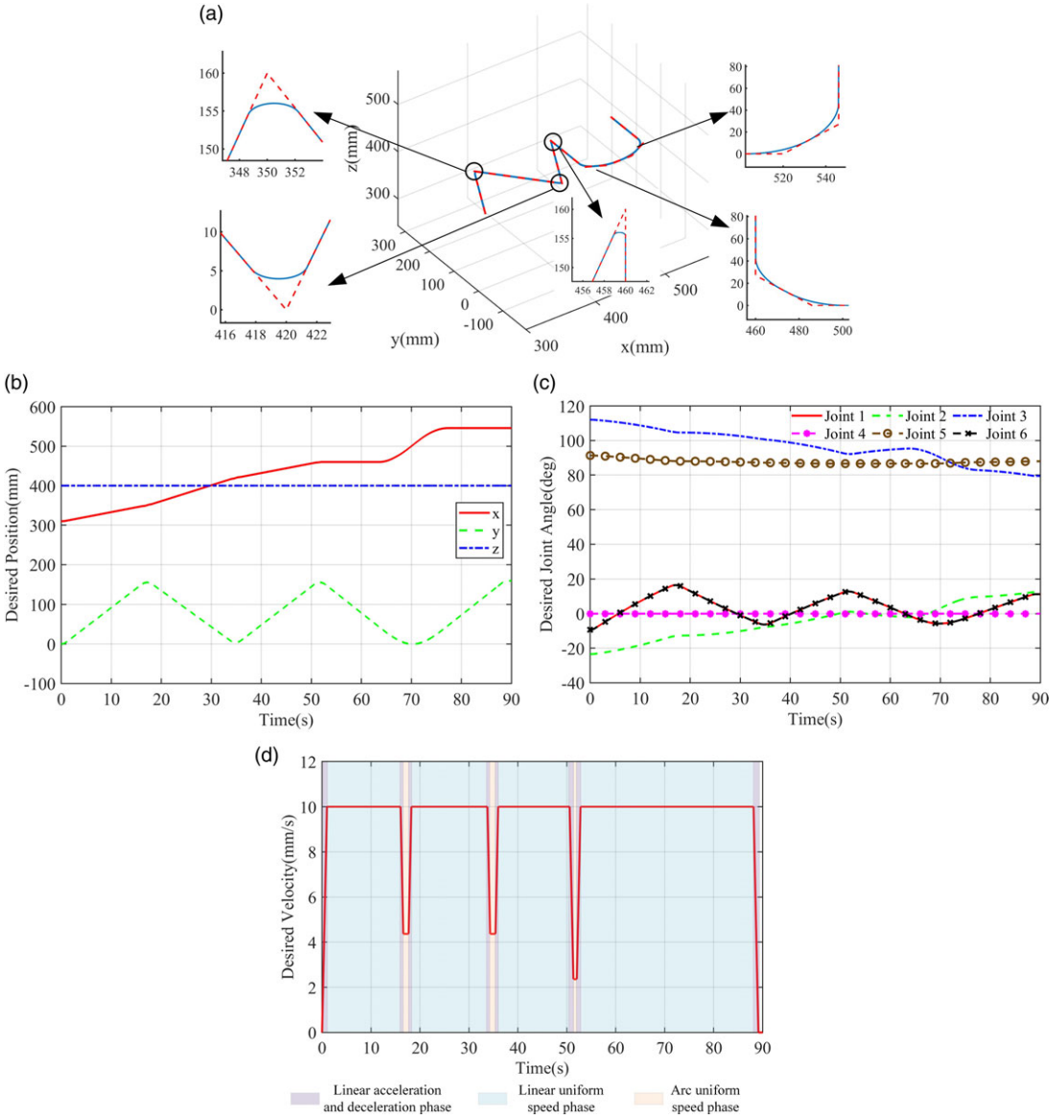


Figure 17. *Schematic diagram of the continuous trajectory: (a) The trajectory in Cartesian space with and without arc transition; (b) Interpolated curves in Cartesian space; (c) Interpolated curves in joint space; (d) The speed in Cartesian space*

On the control side, the control strategy is employed, combining RBF neural network-based dynamic feedforward control with visual servoing feedback enhanced by a velocity observer. This framework improves both responsiveness and trajectory tracking accuracy under complex additive manufacturing paths. The stability of the control system is rigorously analyzed and proven using the Lyapunov method. The experimental results demonstrate that the trajectory planning technique significantly reduces geometric and shape-related errors, while the visual servoing-based tracking control achieves a position accuracy of ± 0.15 mm and an orientation accuracy of $\pm 0.05^{\circ}$, leading to notable improvements in manufacturing precision.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Table VI. *Comparison of motion error in the stable state for continuous trajectory.*

v=10mm/s|Mix error|	No visual servoing	Visual servoing	Improvement
Distance(mm)	0.7616	0.1210	84.1%
x-error(mm)	0.6502	0.1121	81.2%
y-error(mm)	0.5074	0.1064	79.0%
z-error(mm)	0.2878	0.0632	78.0%
α-error(deg)	0.0860	0.0377	56.1%
β-error(deg)	0.1027	0.0185	81.1%
γ-error(deg)	0.0500	0.0170	66.0%

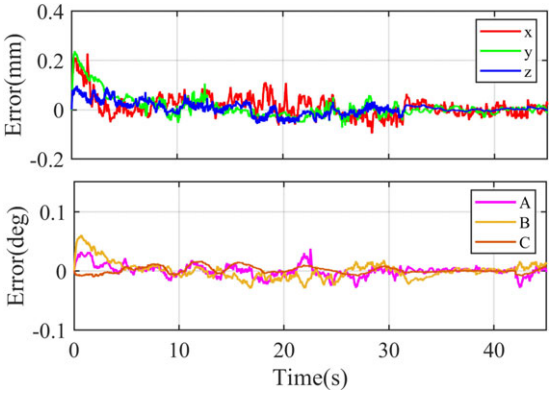


Figure 18. *Continuous trajectory tracking error with visual servoing.*

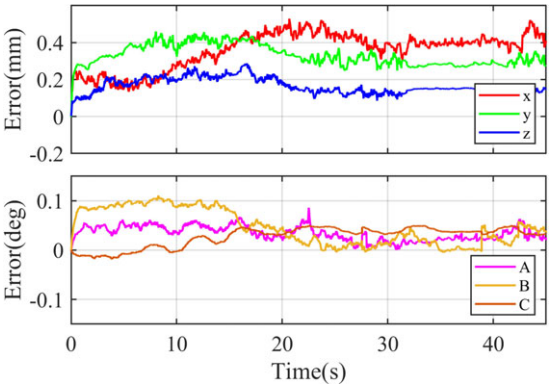


Figure 19. *Continuous trajectory tracking error without visual servoing.*

Future work will focus on improving the robustness of visual measurement under challenging environmental conditions, such as varying lighting, ambient disturbances, and partial object occlusion, to enhance the universality of the proposed method. In addition, advanced filtering strategies will be developed to improve the precision of visual servoing. Finally, tighter integration of equipment accuracy with the additive manufacturing process will be explored. These advancements are expected to facilitate broader industrial adoption of robotic systems.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Table VII. *Measurement results of printed forming precision.*

Measuring objects	Nominal value (mm)	Measuring value (mm)
DX1	204.17	204.29
DX2	185.23	185.35
DY1	19.72	19.85
DY2	19.72	19.85
DY3	79	79.13

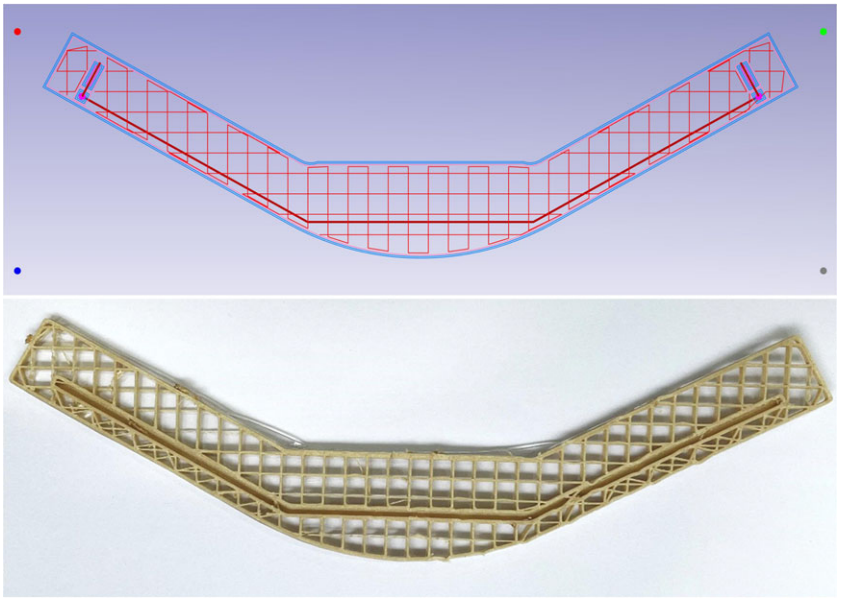


Figure 20. *Robotic printing PEEK material.*

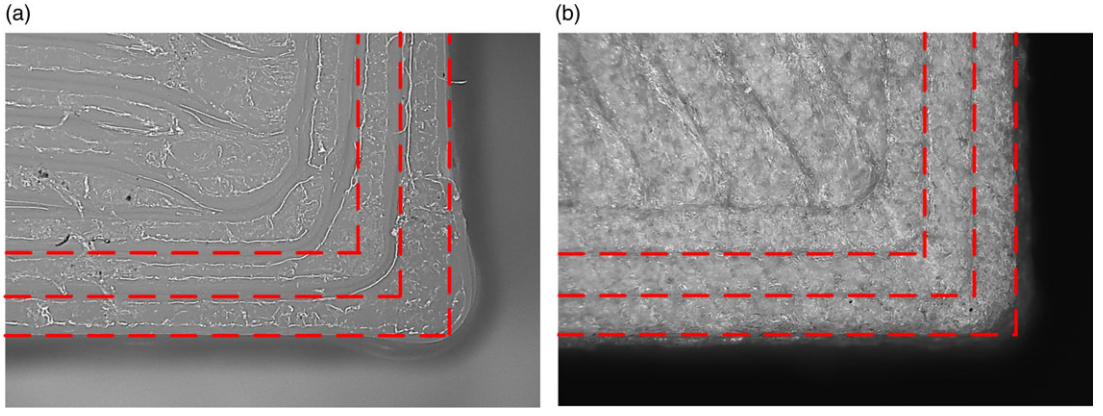


Figure 21. *Microscopic observation of the parts corner (a) without the method proposed in this paper; (b) with the method proposed in this paper.*

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

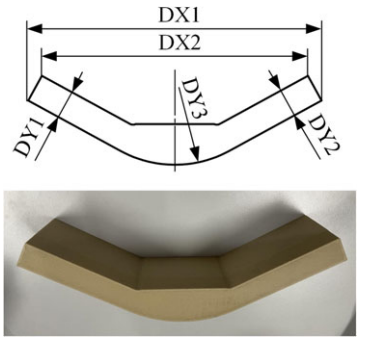


Figure 22. *Measurement results of printed forming precision.*

Author contributions. Delan Wei: formal analysis, investigation, methodology, software, validation, data curation, and writing—original draft. Pengcheng Li: conceptualization, supervision, writing review & editing, and validation. Jun Xiong: supervision, writing review & editing. Yansheng Cao: investigation, resources. Xuewen Wei: software, data curation. Wei Tian: writing review & editing, validation, and resources. Wenhe Liao: conceptualization, supervision, validation, project management.

Financial support. This research was supported by the National Natural Science Foundation of China (No. 52205530), the Natural Science Foundation of Jiangsu Province (No. BK20210299), the Defense Industrial Technology Development Program of China under Grant (No. JCKY2021204B045), and the Key R&D plan of Jiangsu Province (No. BE2021090).

Competing interests. The authors declare no conflicts of interest exist.

Ethical approval. Not applicable.

References

[1] A. Pham and H. Ahn, "Rigid precision reducers for machining industrial robots," *Int. J. Precis. Eng. Manuf.* **22**(8), 1469–1486 (2021). doi: 10.1007/s12541-021-00552-8.

[2] L. Li, A. Haghighi and Y. Yang, "A novel 6-axis hybrid additive-subtractive manufacturing process: Design and case studies," *J. Manuf. Processes* **33**, 150–160 (2018). doi: 10.1016/j.jmapro.2018.05.008.

[3] G. Zhao, G. Ma, W. Xiao and Y. Tian, "Feature-based five-axis path planning method for robotic additive manufacturing," *Proc. Inst. Mech. Eng. Part B-J. Eng. Manuf.* **233**(5), 095440541775250 (2018). doi: 10.1177/0954405417752508.

[4] Y. Bandari, J. Ding, Y. K. Bandari, S. W. Williams and F. Martina, "Additive manufacture of large structures: robotic or CNC systems," **In:** *Proceedings of the 26th Annual International Solid Freeform Fabrication Symposium* (University of Texas, Austin, 2015) pp. 17–25.

[5] X. Dai, C. Feng, Z. Xu, W. Huang, Y. He and J. Shi, "Application progress of robot 3D printing technology," *Mach. Des. Manuf. Eng.* **49**(6), 17–22 (2020).

[6] T. Song, H. Yang, S. Guo, G. Cui and Z. Yan, "Configuration selection for tip-over stability of a modular reconfigurable mobile manipulator under various application situations," *Robotica* **41**(3), 1066–1085 (2023). doi: 10.1017/S0263574722001424.

[7] L. Poudel, L. G. Marques, R. A. Williams, Z. Hyden, P. Guerra, O. L. Fowler, Z. Sha and W. Zhou, "Toward swarm manufacturing: Architecting a cooperative 3D printing system," *J. Manuf. Sci. Eng.* **114**(8), 081004 (2022). doi: 10.1115/1.4053681.

[8] X. Wang, L. Xie, M. Jiang, K. He and Y. Chen, "Kinematic calibration and feedforward control of a heavy-load manipulator using parameters optimization by an ant colony algorithm," *Robotica* **42**(3), 738–756 (2024). doi: 10.1017/S0263574723001728.

[9] R. Luo, W. Gao, Q. Huang and Y. Zhang, "An improved minimal error model for the robotic kinematic calibration based on the POE formula," *Robotica* **40**(5), 1607–1626 (2022). doi: 10.1017/S0263574721001284.

[10] Y. Shen, T. Tang, W. Ye and J. Zhang, "Geometric error modeling and source error identification methodology for a serial-parallel hybrid kinematic machining unit with five-axis," *Robotica* **42**(2), 1–21 (2024). doi: 10.1017/S0263574724002054.

[11] M. Ojer, A. Etxezarreta, G. Kortaberria, B. Ahmed, J. Flores, J. Hernandez, E. Lazkano and X. Lin, "High accuracy hybrid kinematic modeling for serial robotic manipulators," *Robotica* **42**(9), 3211–3229 (2024). doi: 10.1017/S026357472400136X.

[12] E. Uhlmann, M. Polte and J. Blumberg, "Enhanced accuracy specification of heavy-duty industrial robots and the influence of secondary encoders," **In:** *Proceedings of the Machining Innovations Conference for Aerospace Industry (MIC)* (SSRN, 2022). doi: 10.2139/ssrn.4259370.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

[13] Y. Li, W. Tian, B. Li and N. Zhang, "Accuracy control method and experiment of robot milling system," *Acta Aeronaut. Astronaut. Sin.* **43**(5), 11 (2022). doi: 10.7527/S1000-6893.2021.25815.
[14] A. Maghami and M. Khoshdarregi, "Vision-based target localization and online error correction for high-precision robotic drilling," *Robotica* **42**(9), 3019–3043 (2024). doi: 10.1017/S0263574724001255.
[15] B. Li, Y. Li, W. Tian and W. Liao, "Pose accuracy improvement in robotic machining by visually-guided method and experimental investigation," *Robot. Auton. Syst.* **164**, 104416 (2023). doi: 10.1016/j.robot.2023.104416.
[16] Y. Zhao, Y. Lin, F. Xi, S. Guo and P. Ouyang, "Switch-based sliding mode control for position-based visual servoing of robotic riveting system," *J. Manuf. Sci. Eng.* **139**(4), 041010 (2017). doi: 10.1115/1.4034681.
[17] T. Shu, S. Gharaaty, W. Xie, A. Joubair and I. A. Bonev, "Dynamic path tracking of industrial robots with high accuracy using photogrammetry sensor," *IEEE/ASME Trans. Mechatron.* **23**(3), 1159–1170 (2018). doi: 10.1109/TMECH.2018.2821600.
[18] R. Li, X. Cui, J. Lin and Y. Zheng, "An attitude measurement method of industrial robots based on the inertial technology," *Robotica* **40**(8), 2909–2931 (2020). doi: 10.1017/S0263574721002010.
[19] X. Ma, P. Zhu, X. Li, X. Zheng, J. Zhou, X. Wang and K. Wai Samuel Au, "A minimal set of parameters-based depth-dependent distortion model and its calibration method for stereo vision systems," *IEEE Trans. Instrum. Meas.* **73**, 1–11 (2024). doi: 10.1109/TIM.2024.3406802.
[20] T. Jiang, W. Deng, L. Yin, K. Yang, Y. Wang and H. Zhou, "Calibration and pose measurement of a combined vision sensor system for industrial robot grasping of brackets," *Meas. Sci. Technol.* **35**(8), 085903 (2024). doi: 10.1088/1361-6501/ad3a07.
[21] J. Jiang and Y. Ma, "Path planning strategies to optimize accuracy, quality, build time and material use in additive manufacturing: A review," *Micromachines-BASEL* **11**(7), 633 (2020). doi: 10.3390/mi11070633.
[22] H. Zhao, L. Zhu and H. Ding, "A real-time look-ahead interpolation methodology with curvature-continuous B-spline transition scheme for CNC machining of short line segments," *Int. J. Mach. Tools Manuf.* **65**, 88–98 (2013). doi: 10.1016/j.ijmachtools.2012.10.005.
[23] Z. He, H. Fang, Y. Bao, F. Yang and J. Zhang, "Smooth toolpath interpolation for a 5-axis hybrid machine tool," *Robotica* **40**(12), 4431–4454 (2022). doi: 10.1017/S026357472200100X.
[24] S. Zhang, X. Liu, B. Yan, J. Bi and X. Han, "A real-time look-ahead trajectory planning methodology for multi small line segments path," *Chin. J. Mech. Eng-en* **36**(59), 1–13 (2023). doi: 10.1186/s10033-023-00881-x.
[25] R. Comminal, M. Serdeczny, D. Pedersen and J. Spangenberg, "Motion planning and numerical simulation of material deposition at corners in extrusion additive manufacturing," *Addit. Manuf.* **29**, 100753 (2019). doi: 10.1016/j.addma.2019.06.005.
[26] H. Liu, T. Zhao, L. Li, W. Liu, T. Wang and J. Yue, "A path planning and sharp corner correction strategy for wire and arc additive manufacturing of solid components with polygonal cross-sections," *Int. J. Adv. Manuf. Technol.* **106**(11), 4879–4889 (2020). doi: 10.1007/s00170-020-04960-4.
[27] A. Zapata, A. Bendo, M. Spreitler, X. Zhao, C. Bernauer, H. F. Yoshioka and M. Zaeh, "A model-based approach to reduce kinematics-related overfill in robot-guided laser directed energy deposition," *CIRP J. Manuf. Sci. Technol.* **45**, 200–209 (2023). doi: 10.1016/j.cirpj.2023.06.014.
[28] D. Zhu, Y. He, X. Yu and F. Li, "Trajectory smoothing planning of delta parallel robot combining cartesian and joint space," *Mathematics* **11**(21), 4509 (2023). doi: 10.3390/math11214509.
[29] D. Wei, P. Li, W. Tian, X. Wei, Q. Bai, L. Zhang, C. Li and Y. Hou, "Manipulators 3d printing trajectory tracking control combined with rbfnns and visual feedback," **In:** *2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC)* (IEEE, Beijing, China, 2022) pp. 773–778. doi: 10.1109/YAC57282.2022.10023700.
[30] W. Tian, D. Mei, P. Li, Y. Zeng, P. Hong and W. Zhou, "Determination of optimal samples for robot calibration based on error similarity," *Chinese J. Aeronaut.* **28**(3), 946–953 (2015).

Appendix A

In the RBFNNs, $\boldsymbol{x} = [x_i]^T$ and $\boldsymbol{h} = [h_j]^T$ are the inputs and outputs of the input layer and hidden layer of the networks, respectively. The output of the hidden layer is expressed as

$$h_j = \exp\left(-\frac{\left\|\mathrm{x} - c_j\right\|^2}{2b_j^2}\right) \tag{A.1}$$

where $\boldsymbol{c}$ represents the coordinate vector of the center point of the Gaussian basis function of the neuron, and $\boldsymbol{b}$ is the width of the Gaussian basis function of the neurons in the hidden layer.

The output of the RBFNNs is

$$y = \boldsymbol{w}^T\boldsymbol{h} = w_1h_1 + w_2h_2 + \cdots + w_mh_m \tag{A.2}$$

where $\boldsymbol{w}$ is the network weight, $\boldsymbol{w} = [w_1, \cdots w_m]^T$.

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press

Define

$$\begin{aligned} \boldsymbol{M}_x(\boldsymbol{q}) &= \boldsymbol{F}_{MNN}(\boldsymbol{q}) + \boldsymbol{E}_M \\ \boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \boldsymbol{F}_{CNN}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{E}_C \\ \boldsymbol{G}_x(\boldsymbol{q}) &= \boldsymbol{F}_{GNN}(\boldsymbol{q}) + \boldsymbol{E}_G \end{aligned} \tag{A.3}$$

where $\boldsymbol{E}_M$, $\boldsymbol{E}_C$, $\boldsymbol{E}_G$ are the modeling error of $\boldsymbol{M}_x(\boldsymbol{q})$, $\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})$, $\boldsymbol{G}_x(\boldsymbol{q})$ by the RBFNNs, respectively.

The estimation for $\boldsymbol{F}_{MNN}(\boldsymbol{q})$, $\boldsymbol{F}_{CNN}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{F}_{GNN}(\boldsymbol{q})$ are

$$\begin{aligned} \hat{\boldsymbol{F}}_{MNN} &= \hat{\boldsymbol{w}}_M^T \cdot \boldsymbol{h}_M(\boldsymbol{q}) \\ \hat{\boldsymbol{F}}_{CNN} &= \hat{\boldsymbol{w}}_C^T \cdot \boldsymbol{h}_C(\boldsymbol{z}) \\ \hat{\boldsymbol{F}}_{GNN} &= \hat{\boldsymbol{w}}_G^T \cdot \boldsymbol{h}_G(\boldsymbol{q}) \end{aligned} \tag{A.4}$$

where $\hat{\boldsymbol{w}}_M$, $\hat{\boldsymbol{w}}_C$ and $\hat{\boldsymbol{w}}_G$ are the estimated value of the neural networks weight, and $\boldsymbol{h}_M(\boldsymbol{q})$, $\boldsymbol{h}_C(\boldsymbol{z})$ and $\boldsymbol{h}_G(\boldsymbol{q})$ are the kernel function matrix of the neural networks, where $\boldsymbol{z} = \left[\boldsymbol{q}\ \dot{\boldsymbol{q}}\right]^T$.

Cite this article: D. Wei, P. Li, J. Xiong, Y. Cao, X. Wei, W. Tian and W. Liao (2025). "Trajectory planning and tracking control for a robotic additive manufacturing system based on binocular vision guidance", Robotica **43**, 4212–4235. https://doi.org/10.1017/S0263574725102749

https://doi.org/10.1017/S0263574725102749 Published online by Cambridge University Press