Robot automation grinding process for nuclear reactor coolant pump based on reverse engineering

Hongyao Zhang1,2 · Lun Li1 · Jibin Zhao1

Received: 12 September 2018 / Accepted: 14 December 2018 / Published online: 5 January 2019
© Springer-Verlag London Ltd., part of Springer Nature 2019

Abstract
The nuclear reactor coolant pump (NRCP) is the heart of the nuclear power plant. This paper focuses on robot automation grinding processing for NRCP, which includes scanning, point cloud processing, grinding trajectory generation, and quality evaluation system based on reverse engineering. In this work, firstly, the point cloud of NRCP is obtained by robotic scanner system of hand-eye calibration. Secondly, the research proposes a novel method for point cloud simplification, denoising, and boundary extraction base on $k$ neighborhood octree structure. More important, the efficient trajectory generation of grinding relies on transforming point cloud into adaptive triangular mesh. Lastly, quality evaluation system can calculate the deviation between point cloud and qualified workpiece. And the further path is generated according to the deviation. Experiments show that the accuracy of “246” hand-eye calibration method is less than 0.02 mm. The method of point cloud processing has obvious efficiency advantages over other researchers’ algorithms. The final results indicate that the error of grinding is less than 3 mm and efficiency can be improved by 2.5 times compared with manual grinding.

Keywords Automation grinding · Industrial robot · Point data simplification · Trajectory planning · Quality evaluation

1 Introduction

The pump which is used to drive the coolant in the reactor coolant system is called the nuclear reactor coolant pump (NRCP). Currently, the service life of third-generation nuclear power plant is around 50 years. The NRCP needs long-term safe and reliable operation under high temperature, high pressure and radiation conditions. Traditionally, the processing of the complex and large surface workpiece entirely depended on grinding manual instead of automation. The manual grinding was inefficient, and it was difficult to achieve expected accuracy requirements for complex surface workpieces [1, 2]. Robotic machining can follow accurately complex curved path to eliminate error, and robotic machining produced high-quality parts to narrow tolerances and reject rates [3]. In the last decades, the field of robotic machining has continued to develop, especially for grinding. In [4, 5], collision-free path planning in cluttered environments is tackled. An automatic programming system for a robot equipped with a polishing tool was developed. This system is capable to generate collision-free paths for polishing workpieces that have complicated shapes. Detailed investigation of abrasive polishing of free-form surfaces using scanning paths by a robot has been done by Tam et al. [6]. In [7], authors pointed out to generate robot paths from computer-aided design data of free-form surface. Chen et al. [8] demonstrated a method for fast registration of point clouds based on the point-to-triangle distance which involves only Delaunay triangulation of a two-dimensional dataset. Zuo et al. [9] presented the research that the workpiece model is reconstructed according to the unified error between prediction model and discrete point cloud to ensure grinding precision and surface quality. Kharidege et al. [10] proposed a scheme of an automatic polishing system. And path generated is successfully applied in robotic polishing system. Wang et al. [11] outlined the method of path generation for robotic belt surface grinding. A new verification and simulation system of robotic belt grinding that uses industrial robots is presented. In [12], reverse engineering was
utilized to reconstruct models of damaged blades. The deviation was calculated according to the error between the point cloud and damaged models. Song et al. [13] developed an off-line planning method for the control grinding robot parameters based on an adaptive modeling algorithm. In [14], Xiao et al. presented synthetically an integrated polishing method for compressor blade surfaces. The precision consistency was substantially improved. In [15], Sun et al. developed a distinctive method of generating tool paths based on multiple vector fields for NURBS surfaces to maintain G1 continuity of the vector fields. Kharidege [10] et al. described an automatic planning and programming system based on data from a CAD system. In [16] a novel local subdivision tool-path generation method in finish-cut process is developed based on the offset-surface method.

Few prototype robot systems have been researched for grinding large mechanical components, and no technology has been used in the field of nuclear power plants. Despite the significant developments for robotic automation grinding, there are still gaps in comparing the robotic grinding of large mechanical components. Majority of the studies focused on the monotonous issues of processing technology such as robotic grinding tool-path generation or grinding parameter analysis. Simultaneously, almost all researches centralized on small mechanical components for example aviation blades and sanitary ware. And reverse engineering is rarely used to path plan system and quality evaluation.

In this paper, the grinding process of third generation domestically produced NRCP was taken as an example to realize the automatic grinding of robots based on reverse engineering. As shown in the Fig. 1, firstly, the robot scanner system is calibrated with high precision, and the robot scanner system is used to scan NRCP. The NRCP is placed on the CNC turntable which can realize 360° rotation for scanning. Then, the point cloud data which is scanned is denoised, simplified, and triangulated by Section 3. The algorithm is used to generate the grinding processing trajectory in Section 3. Finally, the processing experiments and the quality evaluation are performed in Section 4.

This paper proposed a novel technology of robot automation grinding the NRCP based on inverse reconstruction model. Thereinto, the point cloud of NRCP was scanned by robot scanner system. The grinding allowance could be analyzed by the Euclidean distance between the point data and qualified model. Assuredly, the point cloud was processed based on k neighborhood octree structure, then the tool path was generated and the workpiece was evaluated. After quality evaluation, the grinding allowance was achieved by the calculated between the point data and qualified 3D model. The experimental results showed that the efficiency and accuracy of grinding could be improved significantly according to the systematic method.

### 2 Hand-eye calibration and scanning

The robot scanner system can transform workpieces into point cloud which is used for the trajectory generation and quality evaluation systems. The relationship of hand-eye is a bridge between robots and scanner, so it is one of the most important criteria for industrial robots automation. The section focuses on hand-eye calibration of noncontact measurement systems. Simultaneously, high precision scanning will be described in detail in the section.

#### 2.1 Hand-eye calibration

The point cloud which is scanned by scanner is based on laser coordinate system. The laser coordinate system is constantly changing with the movement of the robot, because 3D scanner is fixed end of the robot. However the robot’s base coordinate always remains fixed. In order to allow 3D scanner to scan complex workpieces, robot and 3D laser scanner should be calibrated. In this paper, the “246” method of hand-eye calibration is proposed. This method is easy to operate and efficient.

The base coordinate system of the robot is called as $\{B\}$. The end-effector coordinate system is called as $\{E\}$, and the laser sensor coordinate system $\{S\}$ is located on the 3D laser scanner whose specific position is unknown. Meanwhile the standard ball (center coordinates $z$) is fixed on coordinate system of $\{C\}$.

The rotation matrix $R^b_z$ and translation matrix $T^b_z$ between $\{S\}$ and $\{E\}$ should be calculated. The standard sphere is used for reference through measuring position of $z$. As presented in Fig. 2, when 3D laser scanner scans the standard ball, the coordinate center $z$ in $\{S\}$ can be solved by geometric relationship. For a fixed point $z$, the relationship between $z$ in $\{B\}$ and $\{S\}$ is as follows,

$$
\begin{bmatrix}
X^b_z \\
1
\end{bmatrix} = 
\begin{bmatrix}
R^b_z & T^b_z \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
R^b_s & T^b_s \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
X^s_z \\
1
\end{bmatrix}
$$

(1)

where, matrix $R^b_z$ is rotation matrix from $\{S\}$ to $\{E\}$. The matrix $T^b_z$ is translation matrix from $\{E\}$ to $\{B\}$. $R^b_z$ and $T^b_z$ can be calculated by robot position information. The matrix $X^s_z$ keeps constant because the relationship between $z$ and $\{B\}$ are fixed. $X^s_z$ can be solved according to geometric relationship $X^s_z = \left( x_s, y_s, \pm \sqrt{R^2 - z^2} \right)$, the center of circle($x_s, y_s$) is fitted by point cloud data. Because the posture of the robot remains unchanged during scanning (translation only), that is, $R^b_z$ keeps constant. The following equation can be obtained from Eq. (1).

$$
R^b_z \left( X^b_1 - X^b_s, X^b_2 - X^b_s, \cdots, X^b_n - X^b_s \right) =
\left( R^b_z \right)^T \left( T^b_e - T^b_s, T^b_2 - T^b_s, \cdots, T^b_n - T^b_s \right)
$$

(2)
The robot’s posture is always changing during scanning, and Eq. 3 can be obtained,

\[(R_1^b R_2^b)^{-1} T_s^e = X_{z1} - X_{z2}\]  \(\cdots \) (3)

A simple, efficient and easy to operate method of “246” hand-eye calibration is put forward based on the above theory. The whole process is divided into two steps: (1) the posture of robot remains constant, \(R_e^c\) can be obtained according to four sets of data and Eq. (2). (2) The posture of robot keeps mutative, \(T_e^s\) can be solved according to six sets of data and Eq. (3).

### 2.2 Improve calibration accuracy and scanning

The error of raw robot scanner system is very large. Three ways are proposed to improve the accuracy of the hand-eye calibration system.

First, when the circle is fitted by point cloud data, there will be slightly larger differences between real circle and fitted circle due to large errors in original point cloud data. The anti-interferential algorithm is raised based on least squares and stochastic theory. The specific algorithm is implemented as follows.

- a. All points are treated as the overall sample.
- b. Eight sample points are randomly selected. The circle is fitted by least squares (eight sample points). The radius and center of circle are taken as a sample.
- c. A threshold is defined. The distance values between all points and the circle are calculated. If the distance value is less than the threshold, the corresponding point coordinate is taken as the matching sample point. If not, no treatment.
- d. Numerous samples can be obtained by repeating steps b and c enough times (e.g. \(n\)).
- e. The sample with the largest number of points is taken as the ultimate sample for all samples. The final radius and the center of the circle are fitted by points in the ultimate sample.
- f. If multiple matching samples contain almost the same number of points, the threshold should be appropriate
increased. Repeat the above algorithm until there is enough discrimination.

It is unrealistic for all the possibilities to be taken into consideration in the algorithm. If the total sample space has 300 points, a total of \(C_{300}^8 \approx 1.48 \times 10^{15}\) possibilities should be calculated. The sample space has about 300 points in this paper. The value of \(n\) is chosen between 6000 and 10,000 enough to exclude the error point.

Then, if the different position \((h)\) of the standard ball is scanned, the divergence of the laser is different. As shown in Fig. 3, the bigger the value of \(h\) is, the more severe the laser divergence is. The divergence of laser will influence scanning accuracy. The error of position \(\varepsilon_s(h)\) can be obtained by the following formula,

\[
\varepsilon_s(h) = \sqrt{\frac{\sum_{i=1}^{n} (x_i-x_s)^2 + (y_i-y_s)^2 - (R^2-h^2)}{n}}
\]  

where \((x_s, y_s)\) is the center of the circle that was fitted by \(S_a\). The relationship between accuracy and height can be described, when different heights of the calibration sphere \((R = 31.013\ \text{mm})\) are scanned. As shown in Eq. 5, the curve expressions of \(\varepsilon_s(h)\) can be obtained by Gaussian fitting. The error of position \(\varepsilon_s(h)\) can be shown in Fig. 4.

\[
\varepsilon_s(h) = 810.4 \exp\left(-\frac{(h-41.74)^2}{4.238^2}\right) + 2.548 \times 10^7 \exp\left(-\frac{(h-6284)^2}{1110^2}\right) + 0.005 \exp\left(-\frac{(h-4.979)^2}{8.294^2}\right)
\]  

The position error of \(X/Y\) coordinate can be expressed in equation \(\varepsilon_s(h)\), \(\Delta x_s = \varepsilon_s(h)\) \(\Delta y_s = \varepsilon_s(h)\). Then the position error of \(Z\) coordinate is calculated by Eq. (6),

\[
\Delta z_s = \frac{\partial z_s}{\partial r} \Delta r = \frac{1}{h} \varepsilon_s(h)
\]  

According to error synthesis theory, the total position error \(\|\Delta e\|\) is defined by Eq. (7),

\[
\|\Delta e\| = \sqrt{\Delta x_s^2 + \Delta y_s^2 + \Delta z_s^2}
\]  

The accuracy of the calibration is directly affected by the value of \(\|\Delta e\|\). And the smaller the value is, the higher the accuracy of calibration is.

The insert shows total position error \(\|\Delta e\|\) and position error of \(X/Y/Z\) coordinate. As illustrated in Fig. 5, the scanning accuracy can meet the expected requirements when the scanning line is located between 0 and 20 mm of the standard sphere.

Furthermore, the divergence of the laser beam increases as the distance between the scanner and the calibration ball increases during hand-eye calibration.

As displayed in Fig. 6, \(L\) is distance between the scanner and the standard sphere, \(D\) is width of laser beam. Double \(\alpha\) is
angle of the emission laser. The relationship between $L$ and $D$ can be described by Eq. (8),

$$D = \frac{\lambda}{2\tan \alpha} \frac{L_{\text{min}}}{L} D_{\text{min}}$$  \hspace{1cm} (8)

where, $D_{\text{min}}$ is the narrowest laser beam, $L_{\text{min}}$ is the smallest distance between the scanner and the standard sphere. And the wavelength of laser is described by $\lambda$. When $\lambda$ and $\alpha$ are fixed value, value of $D$ increases as the value of $L$ increases. If the divergence of the laser increases, the accuracy of the scanner decreases. So the scanner should be placed near the minimum depth of field. The depth of field is between 190 and 390 mm for the 3D laser scanner. The 3D laser scanner should be placed 190 mm~220 mm during hand-eye calibration.

In summary, this paper presented three methods to improve the accuracy of hand-eye calibration, point cloud preprocessing, 0–20 mm height of sphere, and the smallest distance of the scanner field. And the three ways will be approved in the next section.

### 2.3 Hand-eye calibration experiment

The hand-eye calibration system is described in Fig. 7. The system consists of a KUKA KR600 industry robot and a miyi scanCONTROL-2600-100 linear structure laser scanner. And relevant parameters of laser scanner are described as follows,

1. The start of measuring range: 190 mm
2. The end of measuring range: 390 mm
3. The resolution of $x$-axis: 0.005 mm–0.016 mm/640 points(max)
4. The resolution of $y$-axis: 0.005 mm–0.05 mm

The 3D scanner is mounted on the end of industrial robot. The industrial robot moves with the 3D scanner so that the scanner can reach more position. This standard sphere (radius 30.013 mm) is used for reference by measuring the center of the sphere.

In order to verify the accuracy of hand-eye calibration, the standard sphere was scanned by the calibrated robot scanner system. The deviation between the point cloud and the standard sphere was regarded to evaluate the accuracy of the system. Thirty experiments were conducted to evaluate calibration accuracy under unoptimized calibration circumstances. The result of mean radius is $R = 30.0832$ mm and root mean square error (RMSE) is 0.18613 mm. It shows that the unoptimized hand-eye calibration method is difficult to meet the expected measurement accuracy requirements.

To improve accuracy of hand-eye calibration and verify the theory in Section 2.2, the standard sphere was scanned there groups in the conditions which included point cloud preprocessing, 0–20 mm height and the smallest distance of the scanner field. And the standard sphere was scanned 30 times in three conditions respectively. And the standard sphere was also scanned under the combination of three optimization methods. The mean of 30 sets of radius and RMSE are described by Table. 1.

In summary, three solutions can improve respectively accuracy of hand-eye calibration according to radius and RMSE. In particular, when three methods are applied to hand-eye calibration simultaneously, the radius of sphere is closest to the actual radius and RMSE is ameliorated significantly. The comparison between raw hand-eye calibration and combination of three optimization methods is presented by Fig. 8.
The relevant matrix $R_s$ and $T_s$ of hand-eye calibration can be solved by several groups of experimental data,

$$R_s = \begin{bmatrix} -0.936944 & -0.346986 & -0.0416836 \\ 0.34002 & -0.935869 & 0.0806106 \\ -0.0669812 & 0.06123 & 0.995874 \end{bmatrix}$$

$$T_s = [166.75 \quad 87.588 \quad -46.6124]^T$$

The standard sphere was scanned 30 times repeatedly by optimized hand-eye calibration in this paper, $R = 30.01924$ mm and RMSE = 0.039654. It shows that the optimized hand-eye calibration could improve the accuracy. It can meet fully the requirements of measurement accuracy.

### 2.4 High precision scanning

Traditional scanning tool coordinate systems have limitations when scanning complex curved surfaces, because the 3D laser scanner sensor provides only position data in Cartesian space. The distance between the 3D laser scanner sensor and the complex surfaces must be kept as accurate as possible to decrease the effect of robot orientation error. (e.g., the measurement distance can always be maintained at a constant value during the scanning process). It is necessary for robot scanner system to teach a precise tool coordinate system. As illustrated in Fig. 9, the midpoint 4 of the laser beam is TCP for the scanning process. The TCP and posture of the point 4 are hard to be teached. This paper constructed an optimal tool coordinate to couple with the robot for scanning large surfaces as following steps:

a. Each of the three corner points 1, 2, 3 is regarded as TCP respectively. And the three corner points are moved respectively from 4 different directions to a reference point. TCP of three corner points are teached by the XYZ-4-points method.

b. The distance between the point 4 and the line 12 is calculated according to sample book of the scanner. The distance is represented by $H$ (18.25 mm).

c. The corner points 1, 2, 3, and 4 are coplanar. The TCP of the point 4 can be solved according to the geometric relationship.

d. As for the posture of corner point 4 can be replaced by posture of the corner point 1. The posture of point 1 can be teached by AB-2-points method. The tool coordinate system $(x, y, z, a, b, c)$ consisted of TCP and posture.

e. An appropriate distance is added to the value of $z$ axis.

In this work, the TCP coordinates of the three corner points as follows, (6.915, 46.125, 375.887), (7.155, −39.221, 376.387), (−22.827, −38.817, 376.311) were obtained. Then, the coordinate of point 4 could be calculated as (7.034, 3.452, 620.166). Finally, the posture of the tool coordinate system could be defined according to posture of the corner point 1 (81.675, 79.876, 149.303). The TCP and posture of the tool system could be input in teach pendant of KUKA KR600 industry robot.

The path of scanner can be generated according to the model of the workpieces. The equidistant offset curve construction method is adopted [17]. Of course, the influence of the deviation between model and the workpiece can be

---

**Table 1** Related parameters before and after hand-eye calibration

| Solution               | Mean of radius/mm | RMSE    |
|-----------------------|-------------------|---------|
| Not optimized         | 30.0832           | 0.186131|
| point preprocessing   | 30.06112          | 0.101132|
| Height (5–25 mm)      | 30.05035          | 0.094926|
| Distance              | 30.05211          | 0.092130|
| Overall optimized     | 30.01924          | 0.039654|

---

**Fig. 8** Result of the raw calibration method and overall optimized solution

**Fig. 9** Location of scanner’s TCP
eliminated since the scanner had 240 mm depth of field. The point cloud data of workpiece is shown in the Fig. 10. If the workpiece is a revolving body, the workpiece is rotated 30° once during scanning process.

3 Point cloud processing and path generation

The amount of original point cloud data which is acquired by the robot scanner system is very large. Only a part of the data can represent the entire image. And numerous data information is unnecessary in practical applications. Simultaneously, many noise points may generate during the scanning process because of specular reflection. The original point cloud data cannot be used for tool-path planning and quality evaluation directly because it can cause fatal errors. Meanwhile, point cloud simplification can improve the calculation speed, reduce the storage space, and highlight the model features. The point cloud boundary points are easily lost during the point cloud simplifying process. The loss of the boundary point cloud will result in model boundary defects. It will lead to the lack of trajectory during path plan process. Therefore, it is necessary to process the point cloud before tool-path planning and quality evaluation. This paper proposed a point cloud simplification algorithm of boundary data protection. The algorithm also achieved point cloud denoising.

3.1 Point cloud processing

Various point cloud reduction methods have been proposed, such as the bounding box method [18] and the triangular mesh-based mesh reduction method [19]. These methods can’t satisfy the high-density scattered and complex features point data. The point cloud simplification algorithm based on curvature is gradually developed. Lan et al. [20] achieved more practical methods that the scattered data points with complex topology and noise could be processed effectively. The method produced quality point cloud that was suited for CAD/CAM. Shi et al. [21] proposed an adaptive K-means the point cloud simplification method according to the curvature threshold. In [22], Han et al. devised an algorithm to establish the spatial topology relationship for each point. Then edge points were identified and retained using a simple but effective method. Each of these methods has shortcomings, such as low efficiency, high false positive rate, and redundant curvature estimation process. This paper established the k neighborhood of point cloud space based on the octree theory. And the curvature was used for the basic criterion of simplification to achieve point cloud reduction. The algorithm also utilized the octree theory to realize denoising and boundary extraction.

3.1.1 Establishing a neighborhood based on octree

The space partition of point cloud is generated according to octree structure. The speed and accuracy can be ensured for the point cloud boundary extraction and reduction. First, the point cloud is pre-processed to find the extreme coordinates of the scattered point cloud. The final point cloud space is a large cuboid that consists of two vertices \( u = (x_{\text{min}} - c/2, y_{\text{min}} - c/2, z_{\text{min}} - c/2) \) and \( v = (x_{\text{max}} + c/2, y_{\text{max}} + c/2, z_{\text{max}} + c/2) \). The point cloud boundary extraction and reduction. First, the point cloud is pre-processed to find the extreme coordinates of the scattered point cloud. The final point cloud space is a large cuboid that consists of two vertices \( u = (x_{\text{min}} - c/2, y_{\text{min}} - c/2, z_{\text{min}} - c/2) \) and \( v = (x_{\text{max}} + c/2, y_{\text{max}} + c/2, z_{\text{max}} + c/2) \). The point cloud boundary extraction and reduction. First, the point cloud is pre-processed to find the extreme coordinates of the scattered point cloud. The final point cloud space is a large cuboid that consists of two vertices \( u = (x_{\text{min}} - c/2, y_{\text{min}} - c/2, z_{\text{min}} - c/2) \) and \( v = (x_{\text{max}} + c/2, y_{\text{max}} + c/2, z_{\text{max}} + c/2) \). The point cloud boundary extraction and reduction. First, the point cloud is pre-processed to find the extreme coordinates of the scattered point cloud. The final point cloud space is a large cuboid that consists of two vertices \( u = (x_{\text{min}} - c/2, y_{\text{min}} - c/2, z_{\text{min}} - c/2) \) and \( v = (x_{\text{max}} + c/2, y_{\text{max}} + c/2, z_{\text{max}} + c/2) \). The point cloud boundary extraction and reduction. First, the point cloud is pre-processed to find the extreme coordinates of the scattered point cloud. The final point cloud space is a large cuboid that consists of two vertices \( u = (x_{\text{min}} - c/2, y_{\text{min}} - c/2, z_{\text{min}} - c/2) \) and \( v = (x_{\text{max}} + c/2, y_{\text{max}} + c/2, z_{\text{max}} + c/2) \). The point cloud boundary extraction and reduction. First, the point cloud is pre-processed to find the extreme coordinates of the scattered point cloud.

The large cuboid is divided into many small cuboids. Then \( x_{\text{e}}, y_{\text{e}}, z_{\text{e}} \) is the number of divisions for \( x, y, z \) direction. And \( x_{\text{e}}, y_{\text{e}}, z_{\text{e}} \) can be obtained by the following equation,

\[
\begin{align*}
  x_{\text{e}} &= \left\lceil \frac{(x_{\text{max}} - x_{\text{min}})}{d_x} \right\rceil \\
  y_{\text{e}} &= \left\lceil \frac{(y_{\text{max}} - y_{\text{min}})}{d_y} \right\rceil \\
  z_{\text{e}} &= \left\lceil \frac{(z_{\text{max}} - z_{\text{min}})}{d_z} \right\rceil 
\end{align*}
\]

where, \( d_x, d_y, d_z \) is the preset length of divisions in \( x, y, z \) direction. \( \lceil \rceil \) means round up. Three actual length of side in \( x, y, z \) direction is calculated as,

\[
\begin{align*}
  d_x &= \left\lceil \frac{(x_{\text{max}} - x_{\text{min}} + c)}{x_{\text{e}}} \right\rceil \\
  d_y &= \left\lceil \frac{(y_{\text{max}} - y_{\text{min}} + c)}{y_{\text{e}}} \right\rceil \\
  d_z &= \left\lceil \frac{(z_{\text{max}} - z_{\text{min}} + c)}{z_{\text{e}}} \right\rceil 
\end{align*}
\]

As illustrated in Fig. 11, the point cloud is divided into \( x_{\text{e}}, y_{\text{e}}, z_{\text{e}} \) parts along three directions \( x, y, z \) respectively. The length of the small cuboids is \( d_x, d_y, d_z \). The octree segmentation of the point cloud is completed.

The relationship between the small cuboid and octal number is one-to-one correspondence,

\[
S = \sum_{i=1}^{n} s_i 18^{i-1}
\]

where, \( S \) represents the position index of the small cuboid, \( s_i \) is octal code number, \( n \) represents the number of layers of the
octree. As indicated in Fig. 11, octal code of black node is \{1, 6, 13\} separately. The octal number of yellow cuboid is 130, \(S^3 = 88\). That is, the yellow cuboid is the 88th cuboid (total \(8^3 = 512\) cuboids) in the 3rd division octal tree.

The spatial index \((x_k, y_k, z_k)\) of octree partition can be calculated for a point coordinates \(P: (x_p, y_p, z_p)\),

\[
x_k = \left\lfloor \frac{x_p - x_{\text{min}} + c/2}{x_d} \right\rfloor
\]
\[
y_k = \left\lfloor \frac{y_p - y_{\text{min}} + c/2}{y_d} \right\rfloor
\]
\[
z_k = \left\lfloor \frac{z_p - z_{\text{min}} + c/2}{z_d} \right\rfloor
\]  \hspace{1cm} \text{(12)}

\(\lfloor \cdot \rfloor\) is round down, the cuboid octal number can be obtained by the spatial index \(x_k, y_k, z_k\). Meanwhile, \(x_k, y_k, z_k\) is converted to binary code,

\[
x_k = \sum_{i=1}^{n} b_{xi2^{i-1}}\quad y_k = \sum_{i=1}^{n} b_{yi2^{i-1}}\quad z_k = \sum_{i=1}^{n} b_{zi2^{i-1}} . \hspace{1cm} \text{(13)}
\]

The octal code number of the cuboid can be obtained by Eqs. (13) and (14).

\[
s_i = b_{xi2^{0}} + b_{yi2^{1}} + b_{zi2^{2}} . \hspace{1cm} \text{(14)}
\]

For \(k\) neighborhood points of point \(P: (x_p, y_p, z_p)\), the search range includes the cuboid where \(P: (x_p, y_p, z_p)\) is located and 26 cuboids which are adjacent to the cuboid (same layer, upper layer). Other 26 cuboids can be calculated according to \((x_k \pm \varepsilon, y_k \pm \varepsilon, z_k \pm \varepsilon)\) \(\varepsilon \in \{0, 1\}\).

### 3.1.2 Boundary detection

The boundary points are easily lost during point cloud simplification. Loss of boundary points will lead to trajectories missing during post-processing. Therefore, boundary points should be extracted before point cloud simplification. In this paper, the \(k\) neighborhood octree search algorithm is used to extract boundary point cloud. The specific algorithm is as follows,

**Step 1.** For \(P: (x_p, y_p, z_p)\), the \(k\) neighborhood points are obtained in 27 cuboids (the cuboid where \(P_i = (x_{p1}, y_{p1}, z_{p1})\) is located and 26 cuboids around it).

**Step 2.** The average of the distance \(\text{Mean}_p(\text{dist})\) from the point \(P_i = (x_{p1}, y_{p1}, z_{p1})\) to every point in \(k\) neighborhood is calculated.

**Step 3.** Step 1 and Step 2 are performed for all points (the total number of points \(n\)) in the point cloud space, and the total mean \(\text{Mean}(\text{dist}) = \frac{1}{n} \sum_{i=1}^{n} \text{Mean}_p\) and mean deviation \(p = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\text{Mean}_p(\text{dist}) - \text{Mean}(\text{dist}))^2}\) are calculated.

**Step 4.** If \(\text{Mean}_p(\text{dist}) \geq \text{Mean}(\text{dist}) + \Delta e\), \(P_i\) is a boundary point. Find all eligible boundary points are found. The value of \(K\) and \(\Delta e\) are set to effectively extract the boundary points according to the actual density of the point cloud.

Noted, the \(k\) neighborhood points of \(P_i = (x_{p1}, y_{p1}, z_{p1})\) are searched among the 27 cuboids. If the number of points in 27 cuboids is less than \(k\), the upper layer will be continuously searched according to the actual situation.

The traditional point cloud boundary extraction algorithm has the disadvantages that non-boundary points are incorrectly extracted, and the efficiency of extraction is low. As an example in Fig. 12, it can be observed that all boundary points are completely extracted. None of non-boundary points are extracted.

### 3.1.3 Point cloud simplification

The quality of point cloud simplification has a great influence on the accuracy of the trajectory generation. The point cloud density in the region of large curvature should be larger than that in the region of small curvature to ensure surface features. Therefore, it is not feasible to simplify point clouds for fixed Euclidean distance threshold value. The dynamic threshold
values are calculated for each segmentation region to keep the original features as much as possible.

As shown in Fig. 13, the normal vector of \( P_i : (x_i, y_i, z_i) \) is estimated according to the normal vector of triangles that are consisted of \( P_i : (x_i, y_i, z_i) \) and any two points among \( k \) neighborhood points. And the area of each triangle is regarded as the weight of the normal vector calculation.

As described in Fig. 13, \( k-1 \) triangles are composed by \( P_i \) and neighborhood points. The normal vector \( n_{i,j} \) of triangle which is composed of \( P_i, Q_j, Q_{j+1} \) can be calculated by Eq. (15),

\[
n_{i,j} = \frac{(Q_{j+1} - P_i) \times (Q_j - P_i)}{\|Q_{j+1} - P_i\| \times (Q_j - P_i)} \quad j = 1, 2, 3, \ldots, k-1
\]  

(15)

The area of the triangle \( P_i, Q_j, Q_{j+1} \) can be obtained by Eq. (16),

\[
S_{i,j} = \frac{\|Q_{j+1} - P_i\| \times (Q_j - P_i)}{2} \quad j = 1, 2, 3, \ldots, k-1
\]  

(16)

According to Eq. (17), the normal vector of \( P_i \) is calculated by area weighting formula,

\[
n_{P_i} = \frac{\sum_{j=1}^{k-1} S_{i,j} n_{i,j}}{\sum_{j=1}^{k-1} S_{i,j}}
\]  

(17)

The point cloud simplification algorithm is based on the change rate of point cloud curvature. The distribution function of the curvature is defined as,

\[
F(n_{P_i}) = \sqrt{\frac{\sum_{j=1}^{c} \left( \arccos \left( \frac{n_{P_i} \cdot \tilde{n}_P}{\|n_{P_i}\| \cdot \|\tilde{n}_P\|} \right) \right)^2}{c}}
\]  

(18)

where \( \tilde{n}_P = \frac{1}{c} \sum_{j=1}^{c} n_{P_j} \), the point cloud simplification threshold formula is obtained according to the distribution function of the curvature,

\[
\Delta T(n_{P_i}) = \alpha e^{-\frac{\pi}{\theta \cdot (\gamma \cdot n_{P_i})}}
\]  

(19)

For Eq. (19), the parameter values of \( \alpha, \theta \) are modified to accommodate the different point cloud. The algorithm of point cloud simplification is shown as,

a. The average distance \( \tilde{d} \) between \( P_i : (x_i, y_i, z_i) \) and points in the \( k \) neighborhood is calculated.

---

**Fig. 12** Example of boundary detection

**Fig. 13** Principle of normal vector

**Fig. 14** Example of point data simplification
b. The distance between $P_i$ and any point $P_{im}$ in $k$ neighborhood is recorded as $d_{im}$, if $d_{im} < d\Delta T(n_{P_i})$, then $P_{im}$ is discarded, otherwise $P_{im}$ is retained.

c. All points are judged in turn, and the simplification of point cloud is completed.

As an example in Fig. 14, the point cloud of NRCP is simplified according to the algorithm. Obviously, the bigger the change of rate curvature is, the more detailed point features are remained.

### 3.2 Trajectory planning

According to the Delaunay triangulation algorithm [23], the triangle mesh is generated by the simplified point cloud data. This paper proposed a simple and efficient algorithm for the grinding path generation. It is a grinding path generation algorithm based on intersection of the section plane and the triangular mesh surface.

As observed in Fig. 15a, $abc$ is a triangular mesh, $Z_f$ is defined as a section plane. A vertical line is drawn from the vertex $a$ to the plane $XOY$, and the point $f$ is the intersection of the vertical line $ad$ and the plane $Z_f$. The cross-section triangle $efg$ which is consisted of intersecting line $L_{i-1}$, $L_i$, $L_{i+1}$ is the intersection of the triangular pyramid $abcd$ and the sectional plane $Z_f$. The cross-section triangles of other triangular meshes can be found according to the above method. As seen in the Fig. 15b, the blue triangles represent all cross-section triangles.

First, the valid edge and the shared edges are defined separately. For a cross-section triangle, if the left (counterclockwise) of the edge is adjacent to the cross-section triangle, and the right is not adjacent to the cross-section triangle, the edge is defined as valid edge. The two edges of the same cross-section triangle are defined as shared edges. According to the cross-section triangles, the algorithm of valid edges tracking was as follows,

a. As shown in Fig. 15b, all the cross-section triangles are found.

b. As observed in Fig. 16a, for the convenience of description, the edge $L_1$ which is composed of point$_{1}$, point$_{2}$ is the initial valid edge.

c. The next valid edge $L_2$ is found when point$_2$ is taken as the starting point. And the new grinding path is consisted of $L_1$, $L_2$. As illustrated in Fig. 16b, the valid edges continue to be tracked until the starting point point$_{1}$.

d. If two valid edges are tracked simultaneously, the valid edge that cannot form shared edge with last valid edge is chosen as the grinding path. As seen Fig. 17a, two valid edges $L_i$, $L_{i-1}$ are tracked simultaneously. Because $L_i$ and $L_{i-1}$ are shared edges, the valid edge $L_{i-1}$ is selected as the grinding path.

e. As depicted in Fig. 17b, all valid edges are connected end to end to form the final grinding path.

As an example in Fig. 18, the trajectory is generated according to the algorithm of valid edges tracking. The line

---

**Fig. 15** Cross-section triangle

**Fig. 16** Trajectory planning process
spacing of the trajectory is determined by distance or angle of the section plane $Z_f$. The experiment shows that the algorithm performance is more efficient than the traditional algorithm.

### 4 Processing experiment and quality evaluation

This paper developed a new technology to complete robotic automatic grinding for the NRCP. The main design of the system is introduced as follows in Fig. 19. The heart of the system is a heavy-duty 6-axis KUKA robot—KR 600. The robot scanner system which is calibrated by the “246” calibration algorithm consists of the robot and German Miyi 3D laser scanner (Scan Control LLT 2600–100). The scanner is mounted on end of the robot. The NRCP is scanned by the robot scanner system to obtain the point cloud. The grinding trajectory planning and quality evaluation are implemented after the point cloud processing. The end of robot can switch freely between the grinding tools and the scanner through the tool-exchanger system.

The closest distance between the point cloud and the qualified model surface is calculated as the grinding allowance [24]. The removal amount of each grinding is computed according to grinding experience. The grinding times are controlled in conservative manner according to removal amount. After the initial grinding, the point cloud is scanned again. Simultaneously, the grinding allowance

---

**Fig. 17** Trajectory planning

**Fig. 18** Example of final path planning

**Fig. 19** Robotic automation grinding processing system for the NRCP. 1, KUKA Robot KR600; 2, reactor coolant pump model; 3, tool-changer system; 4, CNC turntable; 5, robot controller cabinet; 6, pressure control cabinet; 7, operator station; 8, electrical cabinet; 9, hydraulic pressure station; 10, cutting fluid tank; 11, aisle
of the workpiece is calculated. The grinding allowance is gradually removed until the deviation is smaller than the target error ($e < \text{error}$) by control of the grinding force. The whole grinding process is completed.

As shown in Fig. 20, the left picture demonstrates the quality evaluation ($e > \text{error}$) cannot meet the final quality requirements, and the NRCP needs to be reprocessed. The right picture reveals the final evaluation result. The range of closest distance between the point cloud and the qualified model is 0–4 mm. Meanwhile, the distribution of most area is less than 2 mm. The grinding result fully meets the processing requirements ($e < 5 \text{ mm}$). During the quality evaluation-grinding process, the workpiece is scanned once after the workpiece is grinded multiple times, because the removal amount model is established according to the magnitude of grinding force. And the grinding times are calculated conservatively to ensure the grinding quality. In order to get better highlight the finish grinding effect, this paper provides the final quality evaluation and quality evaluation during grinding process in Fig. 20.

The NRCP grinding experiment and final result are shown in Fig. 21. The amount of standard surface roughness parameter (Ra) is carried out using surface roughness contrast plate. The surface roughness are analyzed by repeated measurements variance analysis. Finally, the Ra of surface is less than 4.8 $\mu$m. In addition, the method can get relatively good performances on both positive curvature region and negative curvature region. Both the geometrical dimensions and the surface roughness can meet the expected accuracy requirements. At the same time, the efficiency of robot grinding is about 2.5 times that of the manual grinding workpiece for the NRCP. The consistency of grinding quality is relatively stable, and the uncontrollable grinding quality is effectively avoided.

5 Conclusion

The comprehensive investigation into robot automation grinding processing for NRCP has achieved several goals:

1. It developed that the way of ‘246’ hand-eye calibration was easy to operate and efficient. This paper achieved high precision and efficiency hand-eye calibration by the way of ‘246’ hand-eye calibration.

2. It showed that the k neighborhood of point cloud space was established based on the octree theory. The octree-based point cloud simplification algorithm had obvious advantages over other researchers’ algorithms. The algorithm can also realized denoising and boundary extraction of scattered point cloud data.

3. It confirmed that a simple and efficient method of grinding path generation was proposed based on the intersection of the cross-section plane and the triangle mesh surface.

4. It developed quality evaluation system. The quality of surface can be efficiently judged by the closest distance between the point cloud and the qualified model surface.

This paper presented pivotal technique for automatic grinding of NRCP robots based on reverse engineering. Robot automatic grinding could guarantee quality consistency. It can improve evidently grinding losses and efficiency. Although a lengthy experimental research was conducted in the present study, and a more comprehensive technology was required to cover a holonomic grinding system. The whole system can be tested at more levels to extend to complicated curved surfaces in other field.
The authors are grateful for the support provided by National Natural Science Foundation of China (grant # 51775542) and National Natural Science Foundation of China (grants # 51605475).

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

References

1. Bogue R (2009) Finishing robots: a review of technologies and applications. Ind Robot-an Int J 36(1):6–12
2. Wang W, Liu F, Liu Z, Yun C (2017) Prediction of depth of cut for robotic belt grinding. Int J Adv Manuf Technol 91:699–708
3. Lajpah L (2008) Simulation in robotics. Math Comput Simul 79(4):879–897
4. Takeuchi Y, Ge D, Asakawa N (1993) Automated polishing process with a human-like dexterous robot. IEEE International Conference on Robotics and Automation, 1993. Proceedings (Vol.3, pp.950-956 vol.3). IEEE
5. Ge DF, Takeuchi Y, Asakawa N (1995) Dexterous polishing of overhanging sculptured surfaces with a 6-axis control robot. IEEE International Conference on Robotics and Automation, 1995. Proceedings (Vol.2, pp.2090-2095 vol.2). IEEE
6. Tam H, Lui OC, Mok AC (1999) Robotic polishing of free-form surfaces using scanning paths. J Mater Process Technol 95:191–200
7. Chen S, Wu C, Xue S, Li Z (2018) Fast registration of 3D point clouds with offset surfaces in precision grinding of free-form surfaces. Int J Adv Manuf Technol 97:3595–3606
8. Berger U, Lepratti R, May M (2005) An approach for the automatic generation of robot paths from CAD-data. IEEE Conference on Emerging Technologies and Factory Automation (Vol.1, pp.7 pp.-297). IEEE
9. Zuo X, Zhang C, Li H, Wu X, Zhou X (2018) Error analysis and compensation in machining thin-walled workpieces based on the inverse reconstruction model. Int J Adv Manuf Technol 95:2369–2377
10. Khairide A, Ting DT, Yajun Z (2017) A practical approach for automated polishing system of free-form surface path generation based on industrial arm robot. Int J Adv Manuf Technol 93:3921–3934
11. Wang W, Yun C (2011) A path planning method for robotic belt surface grinding. Chin J Aeronaut 24(4):520–526
12. Lee KH, Woo H, Suk T (2001) Data reduction methods for reverse engineering. Int J Adv Manuf Technol 17(10):735–743
13. Song Y, Liang W, Yang Y (2012) A method for grinding removal control of a robot belt grinding system. J Intell Manuf 23(5):1903–1913
14. Xiao G, Huang Y, Yin J (2017) An integrated polishing method for compressor blade surfaces. Int J Adv Manuf Technol 88:1723–1733
15. Sun Y, Sun S, Xu J, Guo D (2017) A unified method of generating tool path based on multiple vector fields for CNC machining of compound NURBS surfaces. Comput Aided Des 91:14–26
16. Zhang Z, Feng Y, Ren B, Hagiwara I (2016) Exploratory study of spiral NC tool path generation on triangular mesh based on local subdivision. Int J Adv Manuf Technol 83(5–8):835–845
17. Liu Z, Li X, Song Y, Yi B (2017) Generating spiral tool paths based on spiral enter assistant line. Int J Adv Manuf Technol 92:869–879
18. Sun W, Bradley C, Zhang Y, Loh HT (2001) Cloud data modelling employing a unified, non-redundant triangular mesh. Comput Aided Des 33(2):183–193
19. Lee KH, Woo H, Suk T (2001) Point data reduction using 3D grids. Int J Adv Manuf Technol 18(3):201–210
20. Lan J, Li J, Li J, Zheng L, Hu G, He G (2013) Data reduction based on dynamic-threshold uniform grid-algorithm. Optik 124(23):6461–6468
21. Shi B, Liang J, Liu Q (2011) Adaptive simplification of point cloud using k-means clustering. Comput Aided Des 43(8):910–922
22. Han H, Han X, Sun F, Huang C (2015) Point cloud simplification with preserved edge based on normal vector. Optik 126(19):2157–2162
23. Su T, Wang W, Lv Z, Wu W, Li X (2016) Rapid Delaunay triangulation for randomly distributed point cloud data using adaptive Hilbert curve. Comput Graph 54:65–74
24. Yuwen W, Xiaoming W, Dongming G, Jian L (2009) Machining localization and quality evaluation of parts with sculptured surfaces using SQP method. Int J Adv Manuf Technol 42(11):1131–1139