Fast 6D object pose estimation of shell parts for robotic assembly

Haopeng Hu · Weikun Gu · Xiansheng Yang · Nan Zhang · Yunjiang Lou

Received: 1 March 2021 / Accepted: 29 August 2021 / Published online: 16 September 2021
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2021

Abstract
Shell parts which have symmetric structure, even surface and flat color are common in industrial manufacturing applications. Moreover, the inner and outer surface of them are of similar shape and close to each other. These features make the pose estimation process of the shell parts challenging. Aiming at 6D pose error compensation of parts in high-precision robotic assembly tasks, this work proposes a fast 6D pose estimation method tailored for the shell parts. With a binocular structured light camera to acquire the point cloud data, the proposed method consists of two phases, namely the initial pose estimation phase and local pose estimation phase. In the former phase, principle component analysis algorithm is utilized to calculate a primary pose estimation. An initial pose correction and translation offset methods serve to solve the local optimal estimation problem of the iterative closest point (ICP) algorithm. In the latter phase, the voxel sampling and a novel weighted point-to-plane ICP algorithms are applied to boost the efficiency of the pose estimation method. With two typical shell parts, a simulation and an experiment of pose estimation are conducted to validate the effectiveness of the proposed method. Experiment results prove that both the accuracy and efficiency of the pose estimation method meet the requirement of the given assembly tasks.

Keywords 6D pose estimation · Robotic assembly · Point clouds · ICP

1 Introduction
Within a robotic assembly process, the computer vision system plays an important role in various aspects [4]. For illustration, the pose compensation process, as shown in Fig. 1, is one of the keys to high-precision robotic assembly tasks [27, 32]. It is necessary to compensate the pose error through the robotic movement in advance for reliable robotic assembly [16]. Figure 2(a) demonstrates some assembly tasks such as surface mount assembly [29, 30] and peg-in-hole [26] tasks which only require pose compensation of 3-dimensional (3D) pose errors. The 3D pose error includes the position errors along the x-axis and y-axis and the angle error around the normal of the plane. Compensation of such a 3D pose error can be realized by extracting the geometrical information with 2D vision sensors. However, some assembly tasks call for 6-dimensional (6D) pose error compensation of the parts as indicated by Fig. 2(b). In those tasks, the 3D vision sensors can be utilized to capture the point cloud data (PCD) and estimate the 6D pose of parts. We focus on the 6D pose estimation problem of shell parts in this work.

As shown in Fig. 6, the shell parts studied in this work share the features of symmetric structure, even surface and flat color. There is no obvious shape or color pattern on them, as a result of which pose estimation of them turns out to be challenging. When the part is carried by the end-effector of a robot we cannot expect to obtain complete point cloud data of it. Based on those features of the shell parts, the iterative closest point (ICP) algorithm [2] is selected in this work to carry out point cloud registration process so as to estimate the parts’ 6D pose. ICP is a well-recognized point cloud registration algorithm [13, 28, 33], which is usually exploited in studies upon high-precision pose estimation with PCD. The basic idea of ICP lies in that given an initial pose estimation of the PCD and the model PCD, we build the closest point correspondences under current pose estimation and then we estimate the pose with
the built correspondence. Compared with ICP, key point-based methods [22] suffer from the symmetric structures and obscure patterns of the shell parts. Deep neural network-based methods [10, 12, 29] do not work well when the complete point cloud data are unavailable. Other methods such as [23] are kept away from the robotic assembly tasks by their low estimation accuracy.

When it comes to pose estimation of shell parts, chances are that estimation results of the classical ICP algorithm are trapped into some local optimal estimation, which results in low accuracy and inefficiency of the pose error compensation process. This is mainly owing to the similarity between the inner and outer surface of shell parts and the poorly assigned initial estimation. As indicated by Fig. 3, in the practical application, the PCD captured by the 3D camera only contains the visible side of the parts [5]. These incomplete PCD will be used to register the complete model point cloud which is obtained by sampling the CAD model. When the assigned initial estimation is inappropriate, ICP algorithm may identify incorrect corresponding points from the invisible part of the model point cloud according to the closest point principle, which leads to convergence to the local optimal estimation. In this case, both accuracy and efficiency of ICP will be negatively affected.

To handle the local optimal estimation problem of ICP algorithm some studies utilize the point pair rejection strategy [21], such as translation rejection or normal rejection. By calculating the translation matrices between the point pairs while rejecting the point pairs whose distance is larger than a given threshold, the local optimal estimation problem can be partially solved. Picky ICP [34] and Normal ICP (NICP) [25] methods exploit Euclidean translation or normal direction to prevent ICP from trapping into the local optimum. However, these methods are invalid for the shell parts. On one hand, the distances between the matched point pairs may be too short due to the captured dense PCD. On the other hand, normal vectors of the inner and outer
Fig. 3 Local optimal estimation problem: Given the shell parts, the 3D vision sensor can only capture the point cloud of its visible side. Owing to the poorly assigned initial poses and the similarity of inner and outer surfaces of shell parts, improper corresponding relationships of point pairs will be identified by ICP, which results in local optimal pose estimation.

surface are too similar and close to be distinguished. Other methods such as globally optimal ICP (Go-ICP) [31] are based on global searching without using any point cloud features. Go-ICP searches the configuration space $SE(3)$ to determine the global optimum through the branch-and-bound scheme [1]. However, the experiment shows that it takes up to $20s \sim 40s$ to search the global optimum for 1000 data points [31]. In our previous research [6], the principal component analysis (PCA) and Linemod [8, 9] algorithm are combined to calculate an initial pose estimation of the parts. To boost the pose estimation accuracy, Go-ICP is applied after ICP to search for the globally optimal pose estimation. However, inefficiency of Go-ICP makes this approach away from practical industrial assembly tasks.

Motivated by pose compensation process in robotic assembly application (Fig. 1), an efficient 6D pose estimation method tailored for shell parts is proposed. It consists of two phases, namely the initial pose estimation phase and the local pose estimation phase. Major contributions of this work go as follows. First, to tackle the problem that estimation results of the classical ICP algorithm is easily to be trapped into the local optimal one, an initial pose correction strategy is proposed for the initial pose estimation. Second, in order to boost the estimation efficiency, the weighted point-to-plane ICP is proposed that is combined with voxel sampling to accelerate convergence of ICP. Simulation and experiments with two shell parts validate effectiveness of the proposed 6D pose estimation method.

The remainder of this article is organized as follows. The proposed fast 6D pose estimation approach is introduced in Section 2. Section 2.1 describes its work flow. Two phases of the approach are introduced in detail in Sections 2.2 and 2.3 respectively. To verify the proposed approach, two typical shell parts (cellphone back cover and mouse shell) are selected for simulation and experiments in Section 3. Section 4 concludes the article with a summary of the key points.

2 Fast 6D pose estimation of shell parts

2.1 Overview

In this work a binocular structured light 3D camera [15] is employed whose parameters are listed in Table 1. As shown in Fig. 4, in the robotic assembly process intrinsic/extrinsic calibration process of the 3D camera, including the calibration of the spatial relationship between the camera frame and the robot base frame, must be carried out in advance. Given the pose of the fixed part, the expected pose of the shell part is calculated first. Afterwards, current pose of the shell part is estimated by the proposed method and adjusted to the expected one through movement of the robot. This work focus on the pose estimation process with the intrinsic/extrinsically calibrated 3D camera to calculate the 6D pose error between the current pose and the expected one. The proposed fast 6D pose estimation approach consists of two phases, namely the initial pose estimation and local pose estimation phase as exhibited in Fig. 5. In the initial pose estimation phase, a coarse estimation of

| Table 1 Parameters of the visual sensor |
|----------------------------------------|
| FOV (1000 mm distance) | 800 mm x 500 mm |
| Working distance | 150 mm ~ 1000 mm |
| Precision of depth (z-axis) | 0.01 mm |
| Sampling rate | \leq 1fps |
the 6D pose is calculated as the initial estimation of ICP while the background and outliers in scene point clouds are filtered out. In the local pose estimation phase, a pose correction strategy followed by a translation offset is applied to prevent the ICP algorithm from trapped into a local optimal estimation result. Afterwards, the target/captured PCD is transformed by the inverse matrix of the corrected initial pose. And the object CAD model is sampled to obtain the model PCD. These two PCDs are registered via the ICP algorithm to obtain a 6D pose estimation of the shell parts. To speed up the convergence of ICP, a down-sampling strategy and weighted point-to-plane ICP are applied in this work.

2.2 Initial pose estimation

Principal component analysis (PCA) algorithm followed by a template matching process is used to calculate the initial pose estimation given the point cloud of some shell part. However, as indicated by Fig. 5, point clouds of the scene and the outliers ought to be filtered out in advance. PCA can be taken as an efficient pose estimation algorithm for point clouds though is mainly used for dimension reduction [17]. Its basic idea is calculating the eigenvalues and the corresponded eigenvectors of the covariance matrix of the data and taking $k$ dimensions whose eigenvectors corresponded to the $k$ largest eigenvalues as the principal components. Taking the cellphone back cover for illustration. As shown in Fig. 6, it is convenient to assembly the shell part by fitting one side to the bottom case first. Similar structures can be found in many consumer electronic products. Therefore, similar to [3], we assign the body frame as exhibited by the upper-right graph in Fig. 5. Once axes of the objective shell part’s body frame are defined along with the eigenvectors, PCA can be applied to the covariance matrix of the PCD to obtain a coarse pose estimation.

Let $n$ be the number of data in the PCD captured by the 3D vision sensor and filtered as discussed above. To calculate the covariance matrix $\Sigma$, what ought to be done...
first are calculating the point could centroid \( \tilde{p} \) as (1) and take it as a translation vector \( t \).

\[
t \triangleq \tilde{p} = \frac{1}{n} \sum_{i=1}^{n} p_i
\]

where \( p_i \) is the \( i \)-th point in the PCD. The covariance matrix \( \Sigma \) of the point cloud is calculated by (2).

\[
\Sigma = \frac{1}{n} \sum_{i=1}^{n} (p_i - \tilde{p})(p_i - \tilde{p})^T
\]

Then through singular value decomposition (SVD), eigenvalues \( \lambda_1, \lambda_2, \) and \( \lambda_3 \) and eigenvectors \( u_1, u_2, u_3 \) can be obtained by

\[
\Sigma = U \Lambda V^T, \quad U = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}
\]

According to the eigenvalues, the corresponding relationship between the coordinate axes \( (\text{Axis}_x, \text{Axis}_y, \text{Axis}_z) \) and the eigenvectors \( (u_1, u_2, u_3) \) can be established to obtain the rotation matrix as (4).

\[
R = \begin{bmatrix} \text{Axis}_x & \text{Axis}_y & \text{Axis}_z \end{bmatrix}
\]

However, when the structure of the shell part is symmetric along some direction, such as the cellphone back cover in Fig. 5, orientation of the pose estimated by PCA may be unexpected. This problem can solved by a template matching strategy. The basic idea lies in the process where we determine the positive direction of two coordinate axes through the results of image template matching (including template category and matched pixel position), and finally obtain the third axis according to right-hand coordinate system constraints (5). Details about the template matching process can be found in our previous work [7].

\[
(\text{Axis}_x \times \text{Axis}_y) \cdot \text{Axis}_z > 0
\]

### 2.3 Local pose estimation

Given the initial pose estimation results and the PCD sampled from the CAD model of the objective shell parts, a more accurate pose estimation result is calculated through ICP within the local pose estimation phase. As exhibited in Fig. 5, before the point cloud registration process, some corrections are conducted to the estimation by PCA to acquire a better initial pose estimation and further improve both the estimation accuracy and efficiency of ICP. Those corrections are summarized as the initial pose estimation strategy and include the bounding box-based position correction and translation offset.

#### 2.3.1 Bounding box-based position correction

In the initial pose estimation phase, by PCA centroid of the captured PCD is taken as the initial position estimation as indicated by (1). Anyway, due to the self-occlusion problem there must be a position deviation between the expected position and estimated position which is depending on the viewpoint of the working vision sensor and will bring in extra time consumption to the followed point cloud registration process. Figure 7 shows three viewpoints of the vision sensor and the corresponding visible PCD. The red square therein indicates the area where the PCD are missing, which affects the calculated centroid of the captured PCD, or rather, the position estimation. To obtain a more accurate initial position estimation of the shell parts for better performance of ICP, an initial pose correction process is carried out in advance to the local pose estimation process. Key idea of pose correction is replacing the centroid of captured PCD with the center of the point cloud 3D bounding box as indicated by Fig. 8. To simplify the calculation of the 3D bounding box the original PCD are first transformed by

\[
P_T = T_{\text{init}}^{-1} P_O, \quad T_{\text{init}} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}
\]

where the \( P_O \) represents the original point cloud and \( P_T \) represents the point cloud after the transformation. \( R \) and \( t \) are exactly the orientation estimation (4) and position estimation (1) respectively. By (6) coordinate of the captured PCD \( P_O \) is aligned with the camera frame as \( P_T \), i.e., centroid of the captured PCD is transformed
Fig. 7 Data missing caused by self-occlusion problem: Three viewpoints and the corresponding visible PCD

![Image of data missing caused by self-occlusion problem]

Fig. 8 The pose correction process. $p_{\text{centroid}}$ denotes the position of the centroid of the original PCD (blue). $p_{\text{center}}$ is the center of the transformed PCD (red) through the $SE(3)$ transformation $T_{\text{init}}^{-1}$. The red cube represents the bounding box.

$\mathbf{p}_{\text{max}} = \begin{bmatrix} x_{\text{max}} \\ y_{\text{max}} \\ z_{\text{max}} \end{bmatrix}$

$\mathbf{p}_{\text{min}} = \begin{bmatrix} x_{\text{min}} \\ y_{\text{min}} \\ z_{\text{min}} \end{bmatrix}$

$\mathbf{p}_{\text{center}} = \frac{\mathbf{p}_{\text{max}} + \mathbf{p}_{\text{min}}}{2}$

where $x_{\text{min/max}}, y_{\text{min/max}}, z_{\text{min/max}}$ represent the minimum or maximum value of the coordinates of the PCD, i.e., coordinates of the vertices. Then, the correction vector is given by $\mathbf{p}_{\text{center}}$ and the corrected pose is calculated by transforming the correction to the frame defined by the original point cloud

$\begin{bmatrix} t_c \\ 1 \end{bmatrix} = T_{\text{init}} \begin{bmatrix} \mathbf{p}_{\text{center}} \\ 1 \end{bmatrix}$

where $t_c$ represents the corrected initial position estimation.

2.3.2 Translation offset

As indicated by Fig. 5, the ICP algorithm takes the transformed PCD (6) and the PCD sampled from the to $[0, 0, 0]^T$ as shown in Fig. 8. Afterwards, center of the bounding box is calculated by

$\mathbf{p}_{\text{max}} = \begin{bmatrix} x_{\text{max}} \\ y_{\text{max}} \\ z_{\text{max}} \end{bmatrix}$

$\mathbf{p}_{\text{min}} = \begin{bmatrix} x_{\text{min}} \\ y_{\text{min}} \\ z_{\text{min}} \end{bmatrix}$

$\mathbf{p}_{\text{center}} = \frac{\mathbf{p}_{\text{max}} + \mathbf{p}_{\text{min}}}{2}$

where $x_{\text{min/max}}, y_{\text{min/max}}, z_{\text{min/max}}$ represent the minimum or maximum value of the coordinates of the PCD, i.e., coordinates of the vertices. Then, the correction vector is given by $\mathbf{p}_{\text{center}}$ and the corrected pose is calculated by transforming the correction to the frame defined by the original point cloud

$\begin{bmatrix} t_c \\ 1 \end{bmatrix} = T_{\text{init}} \begin{bmatrix} \mathbf{p}_{\text{center}} \\ 1 \end{bmatrix}$

where $t_c$ represents the corrected initial position estimation.
CAD model as input while calculating the closest point as matching point. However, since there are always missed data caused by self-occlusion problem (Fig. 7) and the similar inner and outer surface are close to each other, the ICP algorithm will easily trap into the local optimal estimation result, as exhibited by Fig. 3. To further boost the accuracy of pose estimation, a local optimal solution jump-out strategy is utilized, in which a translation offset along the object-to-viewpoint orientation (Fig. 9 (left)) is applied on the captured PCD.

As shown in Fig. 9 (right), the gray block represents the model PCD and the green one represents the captured PCD after the inverse transformation of the corrected initial pose. Then the green PCD is translated by a certain translation offset along the object-to-viewpoint orientation \( V_e \) to the red one. Furthermore, the red PCD is used to register the gray one using ICP. In this way, the correct corresponding points are identified based on the closest point matching principle and the local optimal estimation problem is avoided.

The translation offset is applied in advance to ICP, i.e., the initial pose estimation is corrected by

\[
\begin{bmatrix}
R' \\
0 \\
1
\end{bmatrix}, \quad t'_c = (1 - r)t_c
\]

where the parameter \( r \) can be set as ratio of 3~5 times the thickness of the shell parts to the working distance of the camera.

### 2.3.3 Weighted point-to-plane ICP

With the pre-processed captured PCD and the model PCD, 6D pose of the shell part is estimated through ICP algorithm. To further boost the efficiency of ICP for the sake of industrial applications, both the iteration time consumption and the convergence speed are taken into account in this work.

Within each iteration of the typical ICP, processes involving determining the corresponding points and calculating the transformation matrix are time expensive. With the efficiency requirement in mind, the K-dimensional tree [18] is applied in the ICP iteration to determine the corresponding points. At the same time, to accelerate the transformation matrix calculation, the SVD method is applied to the linearized objective function [2]. Another factor for the time consumption of each iteration is the number of points in the PCD. Intuitively, the greater the number of points, the more time the ICP iteration costs and the higher the estimated pose accuracy is [11]. However, as inspired by [19], once the number of points reaches some threshold the accuracy of ICP increases slowly with the number of points. In other words, to accelerate the calculation of ICP, it is rational to down-sample the PCD without the significant loss of accuracy. In this work, the widely applied voxel sampling method is applied.

One of the factors for the convergence speed of ICP is the objective function. The average point-to-point Euclidean distance, as shown in Fig. 10(a), is used as the objective function in the classical ICP. However, objective functions based on the normal information of the point cloud surface have been proven to converge faster [14, 20, 24]. Figure 10(b) shows the case of using the normal information, which is called point-to-plane distance. Considering the structure of shell parts, the latter has a more reasonable corresponding relationship, which makes ICP converge faster. These two optimization objective functions of ICP is expressed as (7) and (8), where \( J_1 \) and \( J_2 \) represent the objective function of point-to-point ICP and point-to-plane ICP, \( n \) is point number of the source PCD, \( p_i \) and \( q_i \)
Fig. 10 Two translation metrics of the corresponding points. (a) Point-to-point translation. (b) Point-to-plane translation

is the \(i\)-th point of source PCD and destination PCD, \(\mathbf{n}_{q_i}\) denotes the normal of point \(q_i\).

\[
\min_{R,t} J_1 = \frac{1}{n} \sum_{i=1}^{n} \| q_i - (R p_i + t) \|^2 
\]

(7)

\[
\min_{R,t} J_2 = \frac{1}{n} \sum_{i=1}^{n} \| (q_i - (R p_i + t)) \cdot \mathbf{n}_{q_i} \|^2
\]

(8)

In this work we utilize the point-to-plane ICP based on the linear least squares (LLS) optimization method [14]. In addition, by extending the LLS optimization to the weighted LLS optimization, the weights are applied to each point pair to speed up the convergence of ICP. The objective function for the weighted point-to-plane ICP is expressed as

\[
\min_{R,t} J_3 = \frac{1}{n} \sum_{i=1}^{n} w_i \| (q_i - (R p_i + t)) \cdot \mathbf{n}_{q_i} \|^2
\]

(9)

where \(w_i\) is the weight of point pair \((p_i,q_i)\). Higher weights are applied to the point pairs with larger distance to speed up the convergence of ICP.

\[
w_i = \| (q_i - p_i) \cdot \mathbf{n}_{q_i} \|
\]

(10)

However, high weights may also bring in fluctuations of pose error in the later period of ICP. Hence, in this work the point-to-plane distance are used as the weights to accelerate the convergence only in the early stage, and the same weights are applied to ensure stable convergence in the later stage of ICP.

3 Experiment verification

Two shell parts, as shown in Fig. 11 and common in consumer electronic products, are selected to verify the effectiveness of the proposed 6D pose estimation approach. As discussed in Section 1, the shell parts studied in this work share the features of symmetric structure, even surface and flat color. There is no obvious shape or color pattern on them. In addition, inner and outer surface of the shell parts are of similar shape and close to each other. Both the cellphone back shell and the mouse shell studied in the simulation and experiment have these features mentioned above. For simplicity, part A refers to the cellphone back cover, and part B refers to the mouse shell. According to requirements of the selected shell parts, the accuracy of the pose estimation algorithm should be no larger than 0.5 mm and 0.5°, and the runtime is expected to be less than 1s.

3.1 6D pose estimation simulation

3.1.1 Setup

Firstly, the Monte-Carlo sampling method is used to obtain the complete PCD of the CAD model. Then, we use point cloud library (PCL) to render the PCD from several viewpoints given the CAD files. In this work, 26 sets of PCD are obtained for the simulation.

Given the same initial pose estimation results by PCA, the classical point-to-point ICP, point-to-plane ICP are taken for comparison. The iteration termination condition of ICP is set as value of the objective function declines to be less than 0.1 or the number of iterations turns to larger than 100. For accuracy comparison, the deviation between the real pose \(T_{\text{true}}\) and the estimated pose \(\hat{T}\) is calculated by

\[
\Delta T = T_{\text{true}} \cdot \hat{T}^{-1} = \begin{bmatrix} \Delta R & \Delta t \\ 0 & 1 \end{bmatrix}
\]

(11)

Fig. 11 The CAD models used in the simulation
where $T_{true}$ is assumed to be known during point cloud rendering in the simulation. Then we decompose the pose error into position error $E_d \triangleq \| \Delta t \|$ and angle error $E_a \triangleq \Delta \theta$. It can be seen that the position error is represented by the Euclidean distance. The angle error is calculated through the Rodrigues formula:

$$\Delta \theta = \cos^{-1} \left(\frac{\text{trace} (\Delta R) - 1}{2}\right)$$

(12)

In addition, the runtime of each pose estimation approach will be recorded for performance comparison.

3.1.2 Simulation results and discussion

As exhibited by Fig. 12 with both parts A and B the proposed method enjoys the least position and angle error. Table 2 lists the accuracy comparison results. It can be seen that the position error of the point-to-point ICP and point-to-plane ICP are both larger than 1 mm due to local optimal estimation problem. In addition, the average runtime of those tests are also recorded and listed in Table 3. It can be seen that the point-to-point ICP cost the longest execution time, while the runtime of the point-to-plane
Table 2  Pose errors of three pose estimation methods

| Methods             | Part A |         | Part B |         |
|---------------------|--------|---------|--------|---------|
|                     | $E_d$ (mm) | $E_a$ (°) | $E_d$ (mm) | $E_a$ (°) |
| Point-to-point ICP  | 1.21   | 0.34    | 3.20   | 2.67    |
| Point-to-plane ICP  | 1.30   | 0.20    | 1.86   | 1.67    |
| The proposed method | 0.20   | 0.18    | 0.15   | 0.11    |

ICP decreased significantly due to usage of the point cloud normal information. And our proposed method uses distance of point pairs as the weights for point-to-plane ICP, which further accelerates the convergence of ICP and shortens the runtime. Note that in some episode the point cloud registration process via classical point-to-point and point-to-plane ICP does not converge when the number of iterations is larger than 100.

3.2 6D pose estimation experiment

3.2.1 Setup

The same CAD models as used in the simulation (Fig. 11) are also selected to obtain the model PCD. Figure 13 exhibits the experiment platform. The shell part in use is fixed on a motion platform. There are two motion platforms, namely the translation and rotation platforms, but only one of them is used in each experiment episode. Hence position or orientation of the shell part is adjusted through the translation or rotation platform, respectively. And readings of the scales on these platforms serve as the ground truth. In the experiments, those platforms are adjusted 50 times to get 49 pose deviations which are the change of pose after adjusting the motion platforms. That is to say, the relative displacement of the 6D pose rather than the absolute pose are estimated. By doing so, effects of the extrinsic calibration error can be partially avoided. Position and orientation of the shell part are given by the translation platform and the rotation platform whose resolution are 0.01 mm and $35''$ ($\approx 0.097^\circ$). In each experiment episode only one motion platform is serving, i.e., we change either the position or the orientation of the shell part by adjusting those motion platforms and reading their scales. To calculate

Table 3  Runtime of the three methods

| Methods             | Runtime (s) |
|---------------------|-------------|
|                     | Part A | Part B |
| Point-to-point ICP  | 2.15   | 1.78   |
| Point-to-plane ICP  | 0.47   | 0.43   |
| The proposed method | 0.28   | 0.25   |

the pose deviation $\Delta T \in SE(3)$ more reasonably, two metrics are defined in this work. As shown in Fig. 14, in the $n$-th experiment episode, the pose $\hat{T}_n$ is obtained by the proposed 6D pose estimation method. Then by metric 1, the pose deviation $\Delta T_{1,n}$ is calculated by

$$\Delta T_{1,n} = \hat{T}_n \cdot \hat{T}_{n-1}^{-1}$$  \hspace{1cm} (13)

while by metric 2 the pose deviation turns out to be

$$\Delta T_{2,n} = \hat{T}_n \cdot \hat{T}_0^{-1}$$ \hspace{1cm} (14)

Similar to simulation in Section 3.1, $\Delta T_{1,n}$ and $\Delta T_{2,n}$ are then used to calculate the position error $E_d$ and angle error $E_a$ which play the role of evaluating indicators of the precision and accuracy performance of the pose estimation approach.

The experiments are conducted in two segments to verify the precision and accuracy of the proposed 6D pose estimation method. As indicated by Fig. 15, to evaluate the precision of the proposed method, the pose of the shell part is estimated $N$ times without any adjustment of the motion platforms. In this case, the expected pose deviation in both metric (13) and metric (14) is $I$ though it will not actually be in the experiment. Then the maximum distance/orientation error is taken to evaluate the precision of the proposed 6D pose estimation method. When it comes to accuracy evaluation, in the $n$-th experiment episode

Fig. 13  Experiment platform: The X-axis and Y-axis indicate the movement direction of the rotation and translation platforms respectively

Fig. 14  Two calculation methods of the pose deviation

$\diamond$ Springer
a pre-defined movement $\Delta T_{\text{true},n}$ is applied to the shell part through the rotation/translation platform as shown in Fig. 15. The maximum error between the estimated pose deviation, which is also calculated by (13) and (14), and the pre-defined movement of the parts is used to evaluate the accuracy performance. In addition, movements of the parts are divided into translation and rotation movements along/around the designated $x$-axis and $y$-axis, respectively. Thus, there are four sets of experiments for each shell part. At the same time, the efficiency of the approach is also evaluated by recording the runtime of each experiment episode.

### 3.2.2 Results and discussion

According to the precision evaluation experiment results, the position and angle errors of parts A and B are drawn as Fig. 16. With $N = 50$ the maximum errors calculated by the two metrics are used to represent the pose precision, which are listed in Table 4. It can be seen that the position and angle errors of part A are less than 0.24 mm and 0.16° respectively, and errors for part B are less than 0.29 mm and 0.37° respectively. Similarly, with $N = 20$ the accuracy evaluation results of parts A and B are shown as Figs. 17 and 18. For each part, the errors between the two types of movements (translation and rotation) along/around the $x/y$-axis and the measured value are plotted in four charts. The maximum error calculated by the two metrics are used to

### Table 4 Precision (maximum position error $E_d$ and angle error $E_a$)

| Part | $E_d$ (mm) | $E_a$ (°) |
|------|-----------|-----------|
| A    | 0.24      | 0.16      |
| B    | 0.29      | 0.37      |
Fig. 17 Error curves for part A in the accuracy evaluation experiment. (a) Translation along X-axis. (b) Translation along Y-axis. (c) Rotation around X-axis. (d) Rotation around Y-axis.

Fig. 18 Error curves for part B in the accuracy evaluation experiment. (a) Translation along X-axis. (b) Translation along Y-axis. (c) Rotation around X-axis. (d) Rotation around Y-axis.
evaluate the pose accuracy, which are recorded in Table 5. For shell part A, the position errors along both the x-axis and the y-axis rotation are less than 0.21 mm, and the angle error is less than 0.12°. For shell part B, the position errors are both less than 0.27 mm, and the angle error is less than 0.38°. It can be seen that estimation accuracy of the proposed method satisfies the requirement of assembling the two shell parts. The pose estimation results of shell parts A and B in the real scene are shown in Fig. 19.

In addition, the average runtime for once pose estimation is 680 ms, including data pre-processing, initial pose estimation and local pose estimation processes. Hence, estimation efficiency of the proposed method also satisfies the requirement of the robotic assembly applications. In summary, the proposed 6D pose estimation approach in this work can fulfill the requirements of shell parts assembly in terms of efficiency and accuracy.

### 4 Conclusion

For the purpose of 6D pose estimation of shell parts for high-precision robotic assembly applications, this article proposes a fast 6D pose estimation method which consists of two phases, namely the initial pose estimation phase and local pose estimation phase. To solve the local optimal solution problem of ICP in the latter phase, an initial pose correction strategy and a translation offset are applied in the former phase. To further improve the efficiency of ICP, the voxel sampling and weighted point-to-plane ICP are applied to accelerate the pose estimation process without loss of accuracy. Through simulation, the performance of three pose estimation approaches, i.e., point-to-point ICP, point-to-plane ICP and our proposed approach, are compared and our approach enjoys the highest accuracy and least runtime. The experiment results prove that the accuracy of our approach is 0.27 mm/0.38 within 680 ms. Although most shell parts are of symmetric structure and even surfaces, there may exist shell parts with complex features. In the future, the 6D pose estimation problem for these irregular shell parts may be considered.

**Author contribution** Haopeng Hu and Weikun Gu contributed more to this work. Xiansheng Yang, Nan Zhang and Yunjiang Lou contributed less to this work.

**Funding** This work was supported by NSFC-Shenzhen Robotics Basic Research Center Program Grant U1713202.

### Declarations

**Ethics approval** Not applicable

**Consent to participate** We consent.

**Consent for publication** We consent.

**Conflict of interest** The authors declare no competing interests.

### References

1. Bazin JC, Seo Y, Pollefeys M (2012) Globally optimal consensus set maximization through rotation search. In: Asian conference on computer vision. Springer, pp 539–551
2. Besl PJ, Mckay HD (1992) A method for registration of 3-d shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256
3. Chang WC (2018) Robotic assembly of smartphone back shells with eye-in-hand visual servoing. Robot Comput Integr Manuf 50:102–113
4. Cohen Y, Naseraldin H, Chaudhuri A, Pilati F (2019) Assembly systems in industry 4.0 era: a road map to understand assembly 4.0. Int J Adv Manuf Technol 105(9):4037–4054
5. Cong Y, Tian D, Feng Y, Fan B, Yu H (2018) Speedup 3-d texture-less object recognition against self-occlusion for intelligent manufacturing. IEEE Trans Cybern 49(11):3887–3897
6. Dong D, Yang X, Hu H, Lou Y (2019) Pose estimation of components in 3c products based on point cloud registration. In: IEEE international conference on robotics and biomimetics. pp 339–344

7. Gu W, Yang X, Dong D, Lou Y (2020) Pose ambiguity elimination algorithm for 3c components assembly pose estimation in point cloud. In: Chinese intelligent systems conference. Springer, pp 251–260

8. Hinterstoisser S, Cagniart C, Ilic S, Sturm P, Navab N, Fua P, Lepetit V (2011) Gradient response maps for real-time detection of textureless objects. IEEE Trans Pattern Anal Mach Intell 34(5):876–888

9. Hinterstoisser S, Lepetit V, Ilic S, Holzer S, Bradski G, Konolige K, Navab N (2012) Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: Asian conference on computer vision. Springer, pp 548–562

10. Li CHG, Chang YM (2019) Automated visual positioning and precision placement of a workpiece using deep learning. Int J Adv Manuf Technol 104(9):4527–4538

11. Li W, Song P (2015) A modified ICP algorithm based on dynamic adjustment factor for registration of point cloud and cad model. Pattern Recogn Lett 65:88–94

12. Lin C, Tsai C, Lai Y, Li S, Wong C (2018) Visual object recognition and pose estimation based on a deep semantic segmentation network. IEEE Sensors J 18(22):9370–9381

13. Lin HY, Liang SC, Chen YK (2020) Robotic grasping with multi-view image acquisition and model-based pose estimation. IEEE Sensors J 1:1–1

14. Low KL (2004) Linear least-squares optimization for point-to-plane icp surface registration. Chapel Hill Univ N C 4(10):1–3

15. Lyu C, Li P, Wang D, Yang S, Lai Y, Congying S (2020) High-speed optical 3d measurement sensor for industrial application. IEEE Sensors J 1:1–1

16. Ma Y, Liu X, Zhang J, Xu D, Zhang D, Wu W (2020) Robotic grasping and alignment for small size components assembly based on visual servoing. Int J Adv Manuf Technol 106(14):4827–4843

17. Murphy KP (2012) Machine learning: a probabilistic perspective. MIT Press, Cambridge

18. Nuchter A, Lingemann K, Hertzberg J (2007) Cached kd tree search for ICP algorithms. In: Sixth international conference on 3-D digital imaging and modeling (3DIM 2007). IEEE, pp 419–426

19. Ravishankar S, Dutt H, Gurumoorthy B (2010) Automated inspection of aircraft parts using a modified ICP algorithm. Int J Adv Manuf Technol 46(1–4):227–236

20. Rusinkiewicz S (2019) A symmetric objective function for ICP. ACM Trans Graph (TOG) 38(4):1–7

21. Rusinkiewicz S, Levoy M (2001) Efficient variants of the ICP algorithm. In: Proceedings third international conference on 3-D digital imaging and modeling. IEEE, pp 145–152

22. Rusu RB, Blodow N, Beetz M (2009) Fast point feature histograms (fpfh) for 3d registration. In: 2009 IEEE international conference on robotics and automation. IEEE, pp 3212–3217

23. Sahin C, Kouskouridas R, Kim TK (2016) Iterative hough forest with histogram of control points for 6 dof object registration from depth images. In: 2016 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 4113–4118

24. Segal A, Hachnel D, Thrun S (2009) Generalized-icp. In: Robotics: science and systems, vol 2, Seattle, WA, p 435

25. Serafin J, Grisetti G (2015) Nicp: Dense normal based point cloud registration. In: 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 742–749

26. Song HC, Kim YL, Song JB (2014) Automated guidance of peg-in-hole assembly tasks for complex-shaped parts. In: 2014 IEEE/RSJ international conference on intelligent robots and systems. IEEE, pp 4517–4522

27. Wang Y, Zhang S, Wan B, He W, Bai X (2018) Point cloud and visual feature-based tracking method for an augmented reality-aided mechanical assembly system. Int J Adv Manuf Technol 99(9-12):2341–2352

28. Wang Z, Fan J, Jing F, Liu Z, Tan M (2019) A pose estimation system based on deep neural network and ICP registration for robotic spray painting application. Int J Adv Manuf Technol 104(1–4):285–299

29. Wang Z, Xu Y, He Q, Fang Z, Xu G, Fu J (2020) Grasping pose estimation for scara robot based on deep learning of point cloud. Int J Adv Manuf Technol 108(4):1217–1231

30. Wei L, Jiao Z (2008) Visual location system for placement machine based on machine vision. In: 2008 Fifth IEEE international symposium on embedded computing. IEEE, pp 141–146

31. Yang J, Li H, Campbell D, Jia Y (2015) Go-icp: A globally optimal solution to 3d ICP point-set registration. IEEE Trans Pattern Anal Mach Intell 38(11):2241–2254

32. Zhao Y, Ye X, Wang L, Yu X, Zhang H (2019) Research on pose point cloud matching error compensation method for confocal image assembly. In: Journal of physics: Conference series, vol 1303. IOP Publishing, p 012073

33. Zhou QY, Park J, Koltun V (2016) Fast global registration. In: European conference on computer vision. Springer, pp 766–782

34. Zinßer T, Schmidt J, Niemann H (2003) A refined icp algorithm for robust 3-d correspondence estimation. In: Proceedings 2003 international conference on image processing (Cat. No. 03CH37429), vol 2. IEEE, pp II–695

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