The International Journal of Advanced Manufacturing Technology (2018) 99:2341–2352
https://doi.org/10.1007/s00170-018-2575-8

ORIGINAL ARTICLE

Point cloud and visual feature-based tracking method for an augmented reality-aided mechanical assembly system

Yue Wang [1] · Shusheng Zhang [1] · Bile Wan [2] · Weiping He [1] · Xiaoliang Bai [1]

Received: 15 March 2018 / Accepted: 13 August 2018 / Published online: 1 September 2018

Abstract

To improve the applicability and robustness of the three-dimensional tracking method of an augmented reality-aided assembly guiding system for mechanical products, a tracking method based on the combination of point cloud and visual feature is proposed. First, the tracking benchmark coordinate system is defined using a reference model point cloud to determine the position of the virtual assembly guiding information. Then a camera tracking algorithm combining visual feature matching and point cloud alignment is implemented. To obtain enough matching points of visual features in a textureless assembly environment, a novel ORB feature-matching strategy based on the consistency of direction vectors is presented. The experimental results show that the proposed method has good robust stability and tracking accuracy in an assembly environment that lacks both visual and depth features, and it can also achieve good real-time results. Its comprehensive performance is better than the point cloud-based KinectFusion tracking method.

Keywords Mechanical assembly · Augmented reality · Tracking · Point cloud · Visual feature

1 Introduction

Assembly is a vital part in the life cycle of the product. Assembly time and assembly quality directly affect product development costs and performance. According to statistics, in modern manufacturing enterprises, assembly workload accounts for 20–70% of the total product development workload, with an average of 45%, and assembly time takes up 40–60% of the manufacturing time [1]. To promote assembly efficiency and quality, some scholars have carried out remarkable studies in this field. Bortolini [2] and Faccio et al. [3, 4] considered component picking, feeding policy, and flexible within an assembly system, and they validated the effectiveness in improving task time performance. Faccio et al. [5] emphasized the packaging problem in a production process, and they defined a packaging strategy that provides a decision-making procedure for operation managers. Hu et al. [6, 7] aimed at product variety and manufacturing complexity in assembly systems, and they proposed a unified measure and models of complexity to assist in designing systems. Besides the above efforts, improving the efficiency and reliability of manual assembly operations is another issue worth studying and discussing.

The traditional assembly process mainly uses paper handbooks for operation guidance. This is not only weak in process guidance, but it can also hinder assembly quality due to misrecognition. With the development of science and technology, a virtual reality (VR)-aided assembly system [8–10], to a certain extent, solves the defects of using paper handbooks. It displays the assembly guiding information in three-dimensional space so that the operator's mental burden and likelihood of error are reduced. However, VR-based assembly systems require the establishment of models of a complex assembly environment, which is time-consuming and laborious. Moreover, the purely virtual assembly system lacks a direct feedback channel. It only supports users with limited "real" experience, which may pose a threat to operators' safety in the complicated and ever-changing workshop. In recent years, the rapid advances in augmented reality (AR) technology have caused revolutionary changes in the assembly area.

✉ Shusheng Zhang
zssnet@nwpu.edu.cn

[1] Key Laboratory of Contemporary Designing and Integrated Manufacturing Technology, Ministry of Education, Northwestern Polytechnical University, Youyi Road 127#, Xi'an 710072, China

[2] Beijing Institute of Spacecraft Environment Engineering, Youyi Road 102#, Beijing 100080, China

It is now possible not only to display virtual information in 3D space but also to display real-world information. AR superimposes the computer-simulated assembly guiding information (3D models, annotation, and text) onto the real world, providing operators with a mixed reality environment that combines real-world scenarios and virtual assembly guiding information. Although AR technology has some drawbacks for end users, e.g., increased cognitive load, sight-related medical issues, mental stress, and adoption willingness, it is still very popular in the mechanical assembly field.

Markerless tracking is one of the core technologies of AR-assisted assembly systems. It is the key factor in achieving accurate superimposition of virtual guiding information on the assembly environment. Markerless tracking has been studied for several years, and significant progress has been achieved. However, there are still some defects and shortcomings. For instance, the feature point-based method [11] is prone to tracking jitter or turbulence because mechanical parts for assembly are textured poorly. The edge-based method [12] is sensitive to cluttered backgrounds and not robust to occlusions during the interaction process. The model-based method [13] has been widely used for markerless tracking in AR-based assembly, but retrieving data from a large number of reference images to find the keyframe leads to a vast searching space and a heavy computational load, which greatly reduces real-time performance and system availability. The simultaneous localization and mapping (SLAM)-based method [14] is limited to estimating a relative camera pose only, which is not suitable for AR-based assembly; moreover, it is prone to tracking failure on dynamic scene. According to some studies [15, 16], point cloud-based tracking shows good robustness in an indoor environment with very dark or sparsely textured areas. However, these methods are the same as the SLAM method, and they can only estimate the relative pose of the sensor, which cannot be applied directly to assembly scenarios that require absolute location information. Moreover, they are prone to tracking failures in a mechanical assembly environment that lacks both visual and depth features. In recent years, machine learning-based methods [17–19] have achieved a huge breakthrough in camera 6-DOF tracking; these "temporal tracking by learning" approaches were demonstrated to be successful at achieving robust, real-time tracking results. However, these approaches are suitable for sequences with relatively small levels of occlusions; they fail for larger levels of occlusion.

In this work, we propose an accurate and robust tracking method combining point cloud and visual features for an AR-aided assembly system. The main contributions of our work are in the following aspects: (1) The tracking benchmark coordinate system is defined using a reference model point cloud to determine the position of virtual assembly guiding information. (2) A novel ORB feature-matching strategy based on the consistency of direction vectors is presented to obtain enough matching points of visual features in textureless assembly environments. (3) An accurate and robust tracking method for an AR-aided assembly system is presented, which avoids tracking failure in mechanical assembly environments that lack both visual and depth features.

Before explaining our approach, we introduce related work in Section 2, and then the theoretical implementation procedure of our approach in Section 3. In Section 4, the experimental results and comparative analysis with other previous approaches are presented. Finally, we close this paper with conclusions and future work in Section 5.

2 Related works

AR has been a hot topic in the field of mechanical assembly in recent years. The stability and robustness of markerless tracking algorithms directly affect the user experience and determine the performance of AR systems. Therefore, improving the robustness of the tracking algorithm is of great significance. Markerless tracking has been studied for several years in the AR community. In this section, we only focus on point cloud-based methods.

Point cloud-based tracking is typically based on variants of the iterative closest point (ICP) algorithm [20, 21]. Newcombe et al. [22–24] proposed a landmark point cloud-based tracking algorithm in the KinectFusion system. Their method obtains depth data from the handheld mobile depth sensor in real time and uses the point cloud data of adjacent frames to perform ICP registration so that the camera pose is estimated. This method can run in environments with low light intensity. However, in practice, their methods are extremely brittle when the environment lacks obvious 3D features like large-scale plate parts or planes, because a flat plane provides no constraints to ICP, causing the point clouds to drift apart. To address this deficiency, researchers explored the use of visual features to provide additional constraints to ICP to improve the robustness of point cloud-based tracking [25–27]. Whelan et al. [28] presented an improved point cloud-based camera pose tracking method that yields high-quality color surface models with few visual artifacts. Their method produced high-quality dense color maps with robust tracking in challenging environments, while still executing in low latency in real time. Henry et al. [16, 29] proposed a point cloud-based tracking algorithm named RGBD-ICP, which is an ICP variant that takes advantage of the richness feature points contained in RGB-D data for ICP initialization. Their method can generate accurate tracking result even in areas where there are no obvious 3D features. However, these methods are likely to fail when the environment is textureless. To improve the accuracy of point cloud-based tracking, researchers have also proposed the use of global pose estimation correction, including pose graph optimization [30], loop closure detection [31], incremental bundle adjustment [32, 33], or recovery from tracking failures by image- or keypoint-based relocalization [34, 35].

Although the above methods demonstrate good tracking performance on their specific systems, most of them can estimate the relative pose of the sensor only, which cannot be applied directly to assembly scenarios that require absolute camera location information. Moreover, most of these methods fail to track in a mechanical assembly environment that lacks both visual and depth features.

3 Approach overview

The basic idea and workflow of the algorithm are shown in Fig. 1. In the offline phase, the reference model point cloud is generated by its CAD model. In the online phase, the transformation relation between the reference model point cloud and the assembly environment point cloud is calculated via an ICP-based registration method. In this way, the tracking benchmark coordinate system is defined. Then, consecutive frame registration based on point cloud and visual features is executed to estimate the pose of the camera in the movement process, and then loop closures are implemented by matching the RGB frame against a subset of previously collected frames to optimize the estimated pose. Finally, the virtual guiding information is superimposed on the assembly environment based on the tracking result.

3.1 Point cloud generation

Depth image is important for point cloud generation. In this paper, an RGB-D depth sensor developed by Sony[1] is used, which captures a 720P registered RGB image and depth points at 30 frames per second. Due to hardware limitations and other environment-affecting factors (surface characteristics, illumination condition, and materials), depth images are usually noisy. To improve the quality of depth image, a weighted joint bilateral filter (WJBF) [36] is adopted for depth image denoising. Then the depth image is converted into 3D point cloud data. $u = (x, y)$ is one pixel on the depth map $D(u)$; it is back-projected to the infrared camera's coordinate 3D space with the depth sensor intrinsic parameters M_{int_D}. To accelerate this process, at time$t$, each CUDA thread operates in parallel on a separate pixel in the incoming depth map $D_t(u)$, and the back-projected points 3D vertex map $V_t(u)$ can be expressed in Eq. 1:

$$V_t(u) = D_t(u) {M_{\text{int}\_D}}^{-1}[u, 1] \tag{1}$$

Then the normal vector $N_t(u)$ of each vertex is computed through the cross-product of the neighboring re-projected points

$$N_t(u) = (V_t(x+1, y) - V_t(x, y)) \times (V_t(x, y+1) - V_t(x, y)) \tag{2}$$

$N_t(u)$ is normalized to unit length with the equation $N_t(u)/\|N_t(u)\|_2$. The normalized $N_t(u)$ and the 3D vertex map $V(u)$ form the assembly environment point cloud data $X_i = \{v_0, v_1, v_2 \ldots v_n, N_0, N_1, N_2 \ldots N_n\}$.

3.2 Benchmark coordinate system establishment

In our method, a reference point cloud model is required for Benchmark coordinate system establishment. The reference model point cloud $P_i = \{p_0, p_1, p_2 \ldots p_m, n_0, n_1, n_2 \ldots n_m\}$ is generated from its 3D model in a computer-aided design system, where p_0, p_1, $\ldots p_m$ and n_0, n_1, $\ldots n_m$ represent the 3D vertices and their corresponding normal vectors, respectively. Taking water pump shell as an example, 5558 surface vertices and normal vectors are obtained (see Fig. 2).

The goal of benchmark coordinate system establishment is to compute the camera pose with respect to the absolute coordinate system, in particular, the reference coordinate system of the point cloud model. This process is achieved by aligning the input point cloud X_i with the reference model point cloud P_i using the ICP algorithm (we assume that the difference in position and direction between the reference point cloud data and the input point cloud data are small).

To align X_i with P_i, the Euclidian distance d between the vertices must be minimized:

$$d\left(v_i, p_j\right) = \min d\left(v_i, p_j\right) \tag{3}$$

Moreover, we introduce an additional constraint to the normal vectors. The similarity between the normal vectors can be expressed by $d_{ij} = \langle n_i, N_j \rangle$, if d_{ij} is greater than a certain threshold ψ, n_i and N_j will be considered as corresponding points, and given the weight 1:

$$w_i = \begin{cases} 1, d_{ij} > \psi \\ 0, else \end{cases} \tag{4}$$

By means of Euclidean distance and normal vector measurement, the number of the point pairs N and the weights of each pair w_i can be obtained. The initial pose of the camera $M_{init} = [R|\, t]$ in the reference model coordinate system can be obtained by solving the equation:

$$e_i(R, t) = \frac{1}{N} \sum_{i=1}^{N_p} w_i \| \mathrm{x}_i - R.\mathrm{p}_i - t \|^2 \tag{5}$$

where R is the rotation matrix of the camera and t is the translation vector.

[1] https://www.sony-depthsensing.com/Depthsense/DepthsenseSensors.

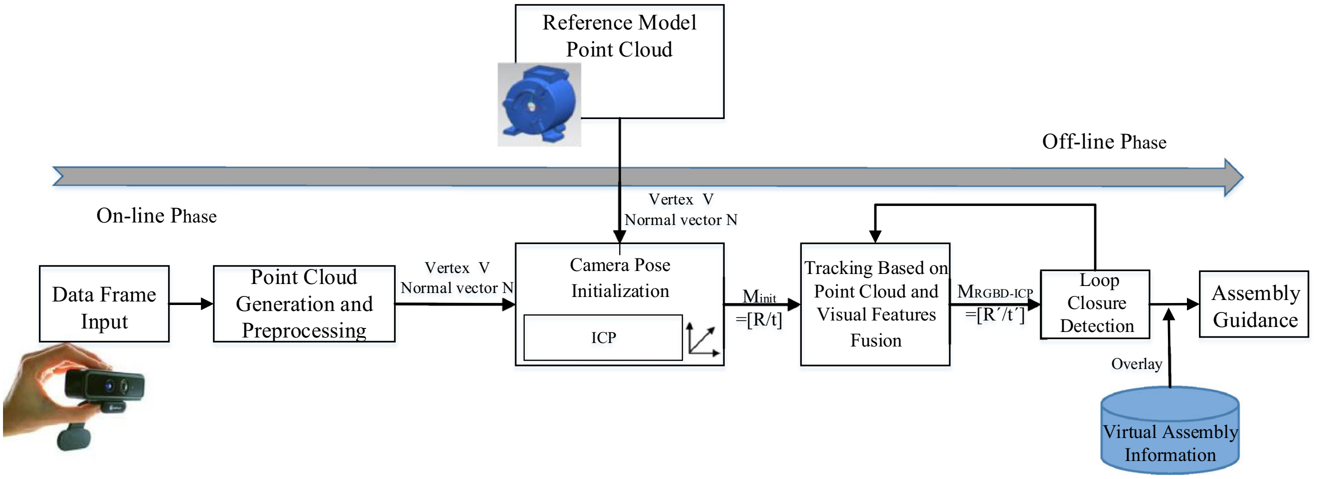

Reference Model Point Cloud
Off-line Phase
On-line Phase
Vertex V
Normal vector N
Data Frame Input
Point Cloud Generation and Preprocessing
Vertex V
Normal vector N
Camera Pose Initialization
ICP
M_init =[R/t]
Tracking Based on Point Cloud and Visual Features Fusion
M_RGBD-ICP =[R′/t′]
Loop Closure Detection
Assembly Guidance
Overlay
Virtual Assembly Information

Fig. 1 The workflow of our method

3.3 Tracking based on point cloud and visual features fusion

The goal of this step is to update the camera pose in the tracking benchmark coordinate system.

In this process, camera pose is computed via an improved ICP-based registration. Traditionally, ICP can easily fail when there are many large parallel planes. Henry et al. [29] proposed an RGBD-ICP algorithm using SIFT features on a color image to enhance the tracking performance in 3D features poorly scenarios. However, the method fails to obtain enough matching feature points in a textureless assembly environment. In this paper, we improve their method by proposing a novel feature-matching strategy. Considering the speed and stability of its feature-matching process, we utilized the widely used feature descriptor ORB [37] in our method. As with SIFT features, the original ORB feature-matching method fails to obtain enough correct and uniform distributed feature point pairs because of inadequate texture features in the assembly environment (see Fig. 3).

The reason for this apparent lack of feature point matches is not too few correct matches, but the difficulty of reliably separating the true and false matches. In addition, motion smoothness point pairs induce correspondence clusters that are highly unlikely to occur at random [38]. Inspired by this theory, we make the following assumption:

Assumption: The unit direction vector of correct matching points should stay the same as that of the matching point pairs in the neighborhood (see Fig. 4).

Assuming M is a matrix composed of the unit direction vectors of matching point pairs in the support region.

$$M = [m_1\ m_2\ m_3 \ldots m_n]^{\mathrm{T}} \tag{6}$$

The similarity between the direction vectors m_i and m_j is measured using the inner product

$$d_{ij} = \langle m_i, m_j \rangle \tag{7}$$

The similarity between $m_{i(1\times 2)}$ and all elements in $M_{(2\times n)}{}^{\mathrm{T}}$ is defined as

$$G = m_i M^{\mathrm{T}} / (n{-}1) \tag{8}$$

All the similarity results in G are sorted in descending order, if one element in G is smaller than a given threshold value δ, the matching point pair is treated as an outlier. This matching process is accelerated through gridding the image into square cells, and the algorithm searches through the input

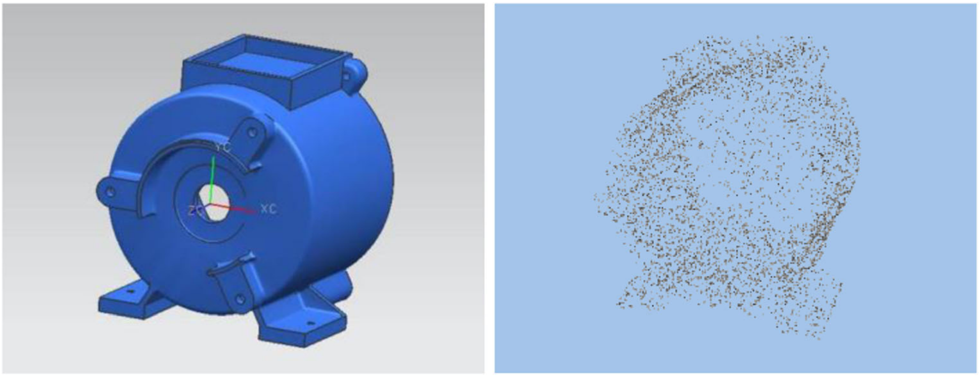
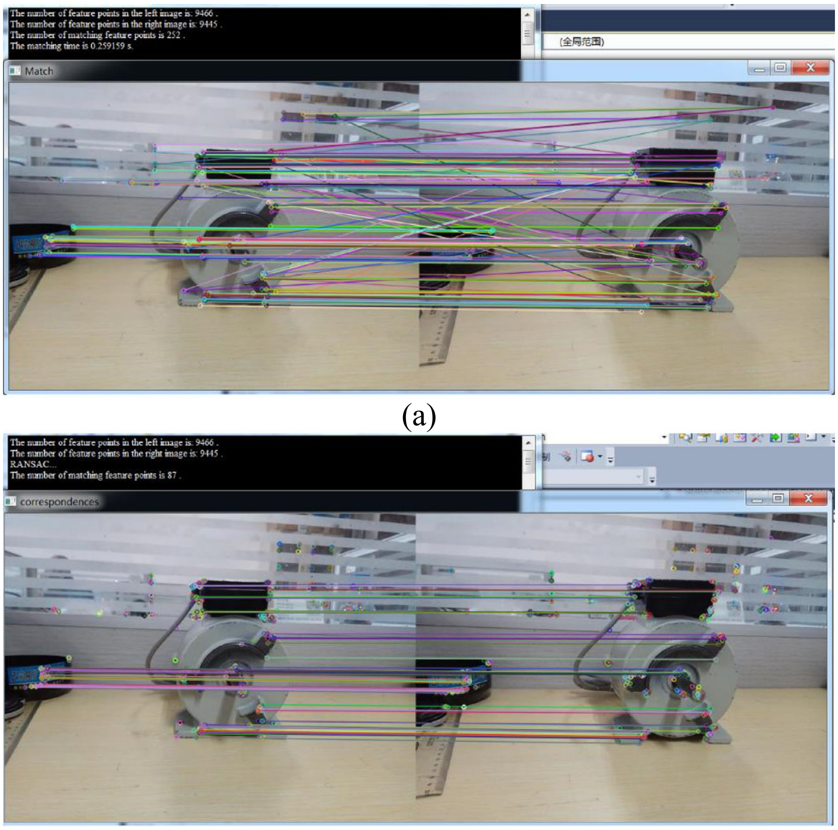


Fig. 2 Point cloud of pump shell generation. (left) 3D model, (right) reference point cloud generated from 3D model



```
The number of feature points in the left image is: 9466 .
The number of feature points in the right image is: 9445 .
The number of matching feature points is 252 .
The matching time is 0.259159 s.
```

(a)

```
The number of feature points in the left image is: 9466 .
The number of feature points in the right image is: 9445 .
RANSAC...
The number of matching feature points is 87 .
```

(b)

Fig.3 **a** Original ORB features matching. **b** Original ORB features matching with RANSAC outlier elimination. The original ORB method can extract 9400+ feature points, but the number of matching point pairs is only 250+ including a large number of mismatches. RANSAC strategy is usually used to eliminate mismatches, but this process further reduces the matching point pairs, and only 80+ matching point pairs is obtained

image with a sliding window. Finally, an adequate number of correct matching point pairs is obtained. The result can be seen in Fig. 5.

After the above preparations have been done, the improved tracking algorithm based on RGBD-ICP and our feature points matching strategy is developed. Firstly, the visual ORB features set $\{F_a, F_b\}$ is extracted from two adjacent frames I_a, I_b, and feature point correspondences $\{C_s, C_t\}$ are obtained with the improved ORB feature-matching method. After that, RANSAC is used to find the best rigid transformation $T^{'} = [R^{'} | t^{'}]$ and the matching inlier points $\{P_{s\_inliner}, P_{t\_inliner}\}$ that generate the best transformation. Then we associate the matched feature points with their corresponding depth values. In one situation, if the percentage of matching points within the best working range is above a given threshold value σ, the main ICP loop is performed to determine the associations between the input source cloud P_a and the target point cloud P_b. In the first iteration, $T^{'}$ calculated from visual RANSAC transformation is used to initialize the point cloud iteration process. And then, for each point in the input source cloud P_a, the nearest point in P_b is determined. The alignment errors of the dense point and visual feature associations are minimized using Eq. (9). The first part of the error function we use *point-to-point* error term to minimize the distance error between the sparse visual feature associations, and the second part of the error function we use *point-to-tangent plane* error

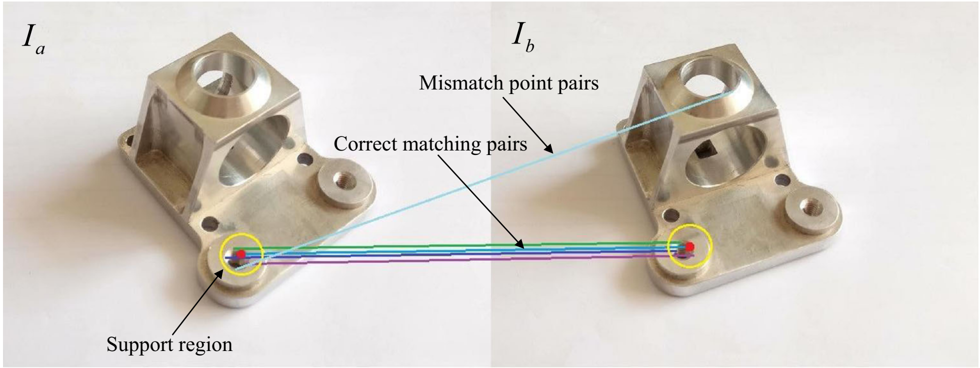
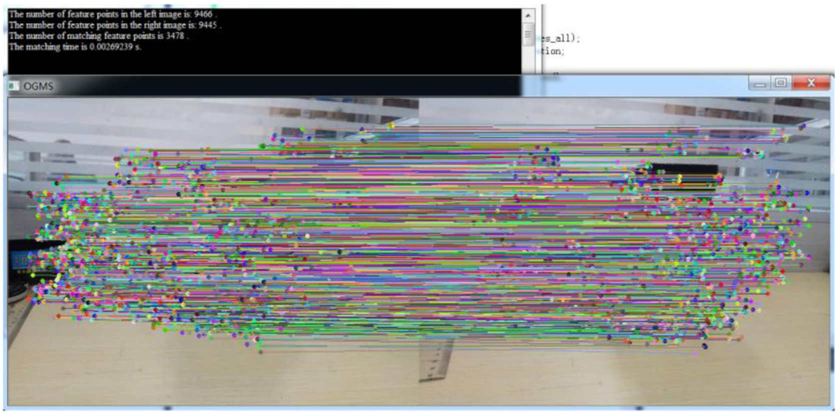


Fig. 4 Feature points matching assumption



```
The number of feature points in the left image is: 9466 .
The number of feature points in the right image is: 9445 .
The number of matching feature points is 3478 .
The matching time is 0.00269239 s.
```

Fig. 5 Our feature points matching method. The correct matching point pairs are increased from 80+ to 3100+, and point pairs are enough and uniform distributed for camera tracking

term (see Fig. 6). Finally, the two parts are weighted using a factor α(in practice, we use 0.5).

$$M^* = \operatorname{argmin} \alpha \left(\frac{1}{|P_{inliner}|} \sum_{i\in P_{inliner}} w_i \left\| \left(R' p^i_{s\_inliner} + t'\right) - p^i_{t\_inliner} \right\|^2 \right) + (1-\alpha)\left(\frac{1}{|P_d|}\sum_{j\in P_d} w_j \left\| \left[\left(R'p^j_a + t'\right) - p^j_b\right] \times n^j_b \right\|^2\right) \quad (9)$$

When the error no longer decreases significantly or the loop reaches the maximum number of iterations, the main loop exits. To reduce cumulative error in the frame alignment process, the loop closure detection and global optimization method using feature points matching those described in RGBD-ICP algorithm [18] is conducted.

In the other situation, if the percentage of matching points within the best working range fall below a given threshold value σ, we use the same features for ORB-SLAM [39] tasks: tracking, mapping, relocalization, and loop closing. We use a constant velocity motion model to predict the camera pose and to perform feature point matching with the last frame. If the tracking is lost, we query the recognition database for keyframe candidates for global relocalization and try to find a camera pose using the PnP algorithm [40]. Through coordination and cooperation between the two working mechanisms, a good tracking performance is achieved.

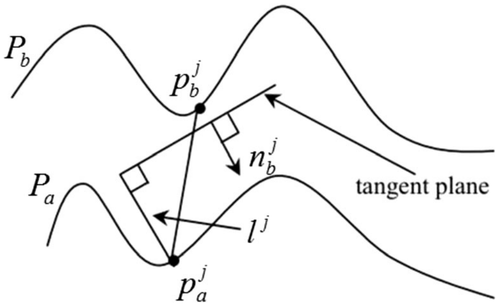


Fig. 6 Point-to-tangent plane error

4 Case study

This section presents a simulated industrial case study of a water pump assembly with AR technology. The pump shell is used for tracking benchmark coordinate system establishment (see Fig. 7a). Its maximal expansions are 120 × 120 × 120 mm in length, height, and width. The reference point sets of the pump shell were created in advance from CAD files and stored in the database with 5558 reference points. Once the tracking process is started, over 60,000+ effective points are generated from the input depth image. Then the reference points are matched with points generated from the input depth image, and the rigid transformation between them is calculated with Eq. 5. After the tracking benchmark coordinate system is established, point cloud and visual features fusion-based tracking using sequential frame image registration is performed with Eq. 9. Finally, the assembly guiding information (the green part) is superimposed onto the real assembly scenarios according to the pose calculated by the proposed method (see Fig. 7b). Then we keep the pump shell motionless, and we hold and move the Softkinetic in an arbitrary path to get 3D registration results of assembly guiding information from different angles and distances. The experimental results are shown in Fig. 7c, d. From the figures, we can see that despite the angle and distance changes, a good registration performance is still achieved.

Table 1 shows the execution time of each frame. The test platform was a desktop PC with an Intel Xeon-E3 3.5 GHz microprocessor and an NVIDIA GeForce GTX 970 8GB RAM graphics card. Softkinetic DS325 was used to capture video sequences, and all captured RGB frames were set to 640 × 480. In this work, the tracking algorithm was written in C++ and imported into unity3D as a dynamic link library. During the tracking process, the reference point cloud is first matched with the point cloud generated from the input image for tracking benchmark coordinate system establishment. This process is time consuming, and it takes almost 2 s. But because this process is executed once, and only once, the time cost of this process is not included in the total tracking time.

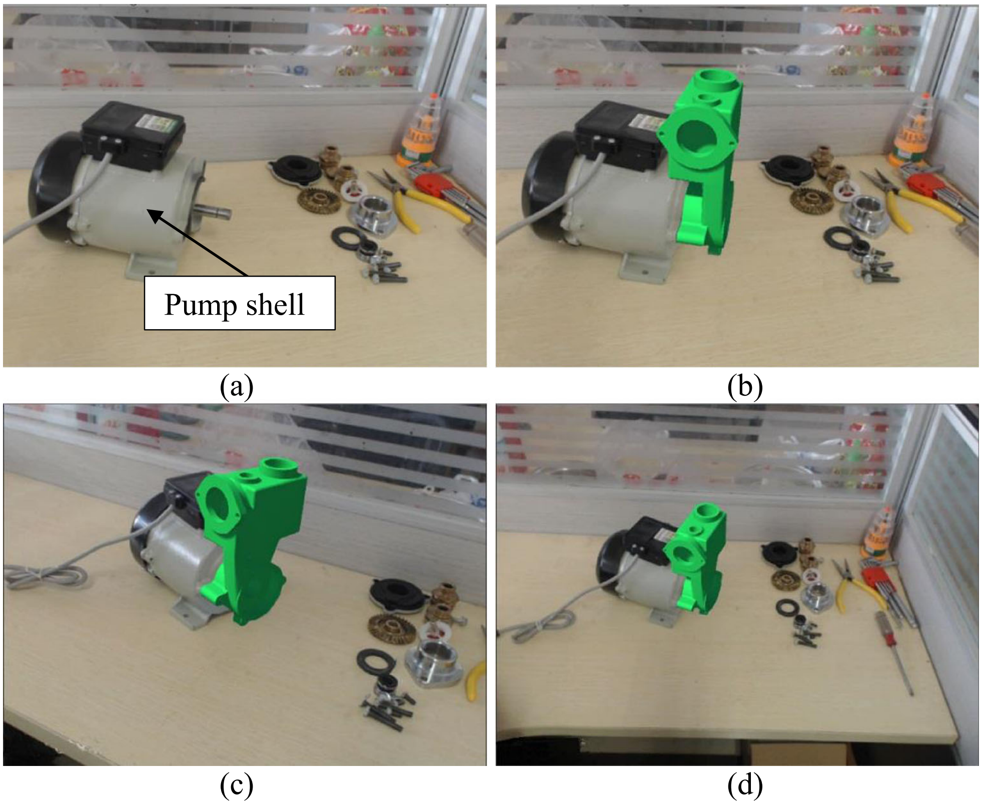


Fig. 7 Case water pump tracking. **a** Initial tracking area. **b** Assembly guiding information superimposed onto the pump shell. **c**, **d** Assembly guiding information registration result under different view angles and distances

Point cloud and visual features fusion-based tracking can be divided into four processes: data frame input, point cloud generation and preprocessing, fusion tracking, and loop closure detection. The whole tracking procedure takes about 23.19 ms (< 33 ms) for each frame; therefore, the proposed method can run at 30 frames per second (Fps), which satisfies the real-time requirement.

5 Results and discussion

5.1 Tracking accuracy analysis

The tracking accuracy of our method is evaluated using HTC Vive Tracker[2] (positioning accuracy 1.5 mm). In this experiment, a pump shell is put on the desk, and the manipulator carries and moves the depth sensor around the desk. HTC Vive Tracker is tied to the depth sensor, and its trajectory is recorded as the ground truth. At the same time, the tracking accuracy of our method is contrasted with other point cloud-based methods (KinectFusion [22] and RGBD-ICP [18]). The results of the experiment are shown in Fig. 8. It can be seen from the figure that the position error between KinectFusion and the ground truth increases as the moving distance increases, and KinectFusion tracking fails when we move the depth sensor close to the desktop, where there are no obvious 3D features visible (at about the 200th frame). RGBD-ICP performs slightly better than KinectFusion, but it also fails at about the 300th frame. This is mainly because when the depth sensor is moved much closer to the desktop, there are not enough visual features presented, resulting in not very strong constraints to the ICP. So the tracking process fails. The tracking trajectory estimated by our method is very close to the ground truth, and our method still shows robust tracking performance when environment lacks both visual and depth features. This is because a significant number of visual features obtained by our method act as an initialization for ICP, which avoids the point clouds drifting apart.

Table 2 shows the rotation error in the whole tracking process. From these results, we can see that our method

Table 1 Tracking performance for each frame

Number of steps	Process	Time/ms
1	Data frame input	2.86
2	Point cloud generation and preprocessing	13.79
3	Fusion tracking	4.41
4	Loop closure detection	2.13
5	Total time	23.19

[2] https://www.vive.com/cn/vive-tracker/.

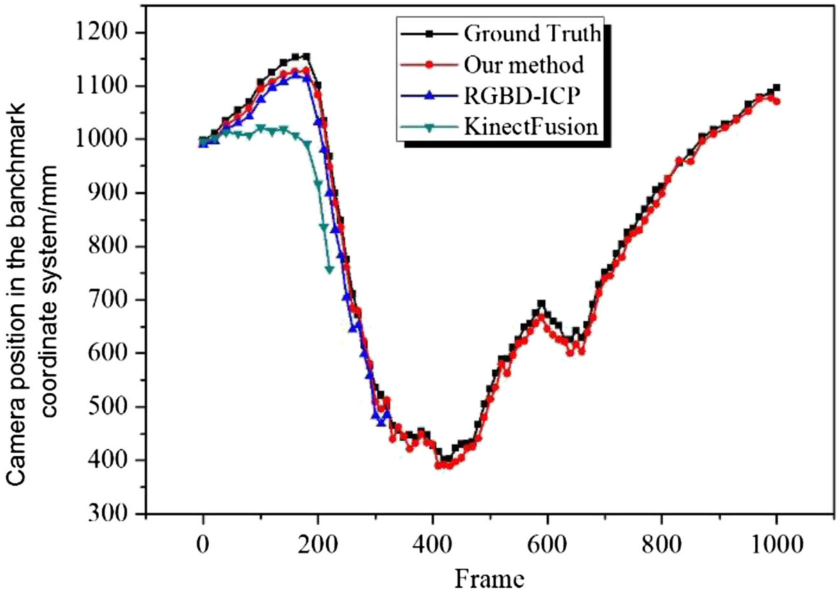


Fig. 8 Tracking trajectory contrast between different methods

outperforms other two methods and satisfies the tracking accuracy requirements of augmented reality-aided system.

5.2 Time analysis

The tracking performance of each algorithm is evaluated, and the time cost of each frame is recorded during the whole tracking process (in practice, we tested for 200 s). The test platform used was the same as "Case Study" section. The results can be seen in Table 3. From the table, we can see that the time cost of our method is greater than the other two methods, but it is still capable of operating at the sensor frame rate of 30 Hz, with average execution time below 33 ms.

5.3 Tracking performance analysis

To investigate the tracking performance of the proposed method in the manufacturing conditions (e.g., when the tracking area is partially occluded during tracking benchmark coordinate system establishment and the fusion tracking process, and the cluttered background is without a strong 3D distinction to the assemblage or the lighting condition variations), some experiments are conducted. In this experiment, a pump shell is put on the worktable, and some parts are stacked around it to simulate the manufacturing environment. The manipulator carries and moves the depth sensor around the desk to test the tracking performance under different conditions. The results can be seen in Fig. 9. From Fig. 9a, b, we can see that the proposed method can still achieve good tracking performance under the partial occlusion condition. This is mainly because the initial camera pose can be estimated with only the partial environment point cloud and reference point cloud registered. Meanwhile, the subsequent point cloud and visual features fusion-based tracking is irrelevant to the initial tracking area; it is only determined by the sequential frame image registration result. However, if most of the initial tracking area is occluded or seriously interfered with by the foreground or background objects, an error virtual object registration result will occur (see Fig. 9c, d). Figure 9e shows the tracking result of our method under low light intensity conditions. It shows that our method is robust to illumination changes. This is primarily because point cloud-based tracking and visual features-based tracking can supplement each other. Although lighting condition variations could affect ORB descriptor generation, a point cloud can still be generated using an infrared camera. Therefore, the tracking procedure is not affected.

Table 2 Rotation error of each algorithm

Methods	Average/ degree	Maximum/ degree
KinectFusion	13.9	25.2
RGBD-ICP	7.2	14.2
Our method	6.5	13.4

Table 3 Tracking performance of each algorithm

Methods	Average/ ms	Maximum/ ms
KinectFusion	23.95	33.55
RGBD-ICP	29.26	36.59
Our method	32.52	43.47

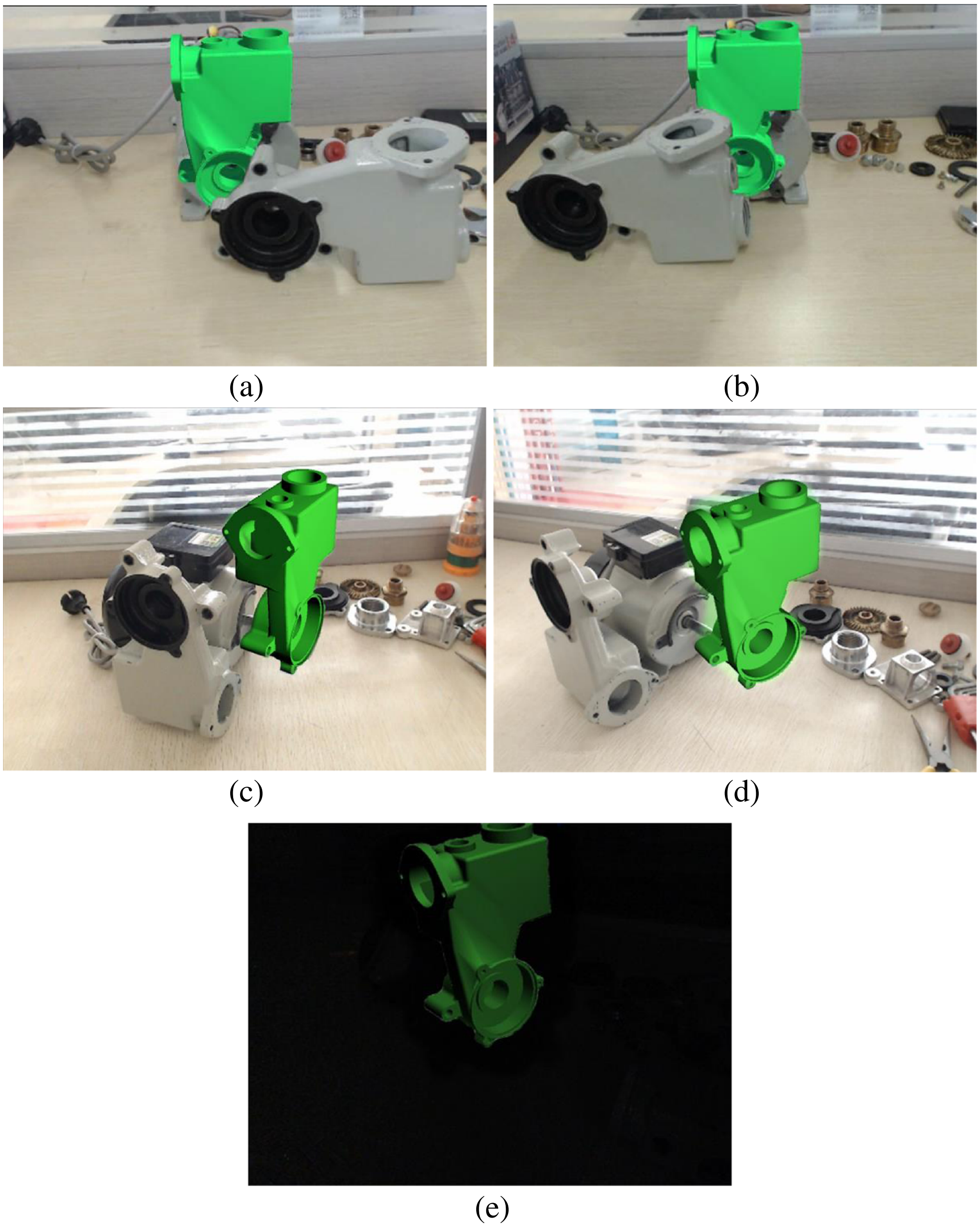
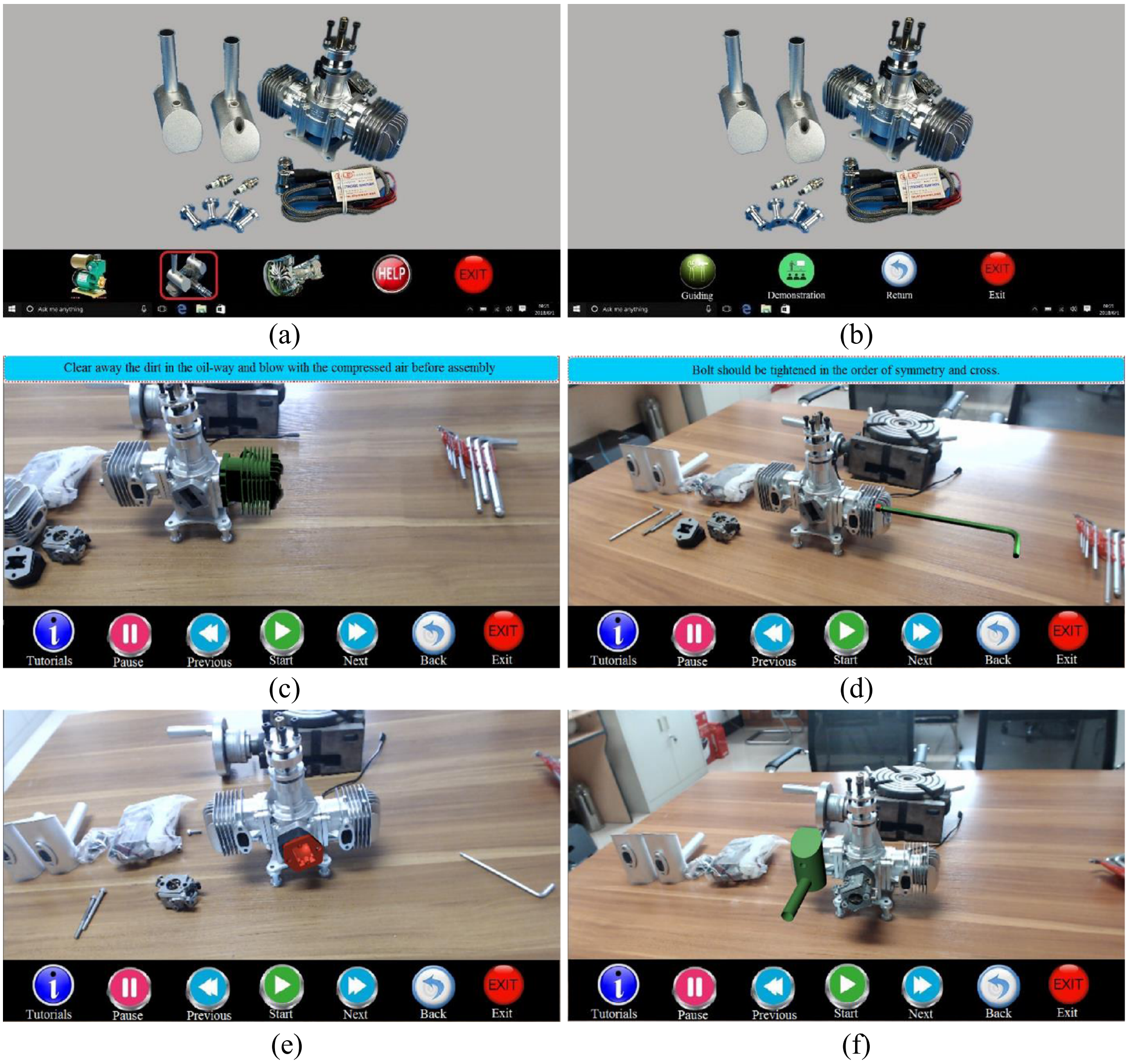


Fig. 9 Tracking performance of our method under manufacturing condition (with virtual-real objects occlusion handling). (**a**) and (**b**) show the proposed method achieves good tracking performance under the partial occlusion conditions. (**c**) and (**d**) show the error virtual object registration result due to the foreground or background objects interference during camera pose initialization. (**e**) shows the tracking result of our method under low light intensity conditions

5.4 Augmented reality assisted assembly

To verify the correctness and practicality of our algorithm, an AR-assisted unmanned aerial vehicle engine assembly system is developed. The system adopts a tablet platform for ease of use and for an intuitive guiding information display. To improve the system operating speed, assembly guiding information is stored on a remote server, and the system can access the data through a wireless network. Meanwhile, to improve the system flexibility for different tasks, different assembly tasks can be selected when operators access the system (see Fig. 10a). When selecting an assembly task, the operators can choose between two models: AR assembly animation demonstration model and AR manual assembly guiding model (see Fig. 10b). On touching the "start" button for manual assembly guiding, the operator should choose an appropriate view angle to ensure as many of the point cloud generated from the input data match the reference model point cloud, so that the initial camera pose can be calculated accurately. Then the virtual part model is superimposed onto the corresponding physical part's assembly position on the basis of the fusion tracking result to prompt the operators to assemble the part in the right position (see Fig. 10e, f). For some important assembly procedures, animations and texts are adopted at the same time to make the assembly process easily understandable for the operators (see Fig. 10c, d). According to the AR guiding steps, the operator can install the part to the target location easily. However, in the current version, the initial tracking area (the pump shell) should remain motionless after benchmark coordinate system establishment to ensure that the virtual



(a)

(b)

Clear away the dirt in the oil-way and blow with the compressed air before assembly

Bolt should be tightened in the order of symmetry and cross.

(c)

(d)

(e)

(f)

Fig. 10 Augmented reality assisted unmanned aerial vehicle engine assembly system, (**a**) different assembly tasks selection, (**b**) guiding model selection, (**c**) cylinder assembly guiding, (**d**) bolts assembly guiding, (**e**) electric plug assembly guiding, (**f**) gas vent assembly guiding

guiding information is superimposed on the right position. If the pump shell is moved during the assembly procedure, reinitialization is required to ensure the following virtual guiding information is displayed correctly.

6 Conclusions and future work

Markerless tracking is still an important but challenging problem in AR-assisted assembly systems. It is the key factor in achieving accurate superimposition of virtual guiding information on the assembly environment. This paper has presented a 3D tracking method for AR-assisted mechanical assembly systems. First, aiming at improving the applicability of the tracking method for assembly scenarios, we proposed a reference model point cloud-based tracking benchmark coordinate system definition method to determine the position of virtual assembly guiding information. Second, to improve the robustness of the point cloud and visual features fusion-based tracking method, a novel ORB feature-matching strategy based on the consistency of direction vectors was presented.

In addition, the tracking accuracy and practical property of the algorithm were presented through a pump assembly case study. The results showed that our tracking method could run at 30 *Fps* and that good tracking performance was still achievable despite the angle and distance changes. To investigate the performance of the proposed method in manufacturing conditions further, a series of comparison experiments took place. The results revealed that the tracking trajectory estimated by the proposed method was very close to the ground truth; it was superior to the KinectFusion method. Moreover, the proposed method also demonstrated robust tracking performance when the environment lacks both visual and depth features, which is effective and practical for AR-assisted assembly systems.

There are still some limitations to our study; for example, the running speed of the proposed method needs further improvement, and the assemblage must remain motionless after the tracking benchmark coordinate system is established. In our future research, we will adopt multithreading technology for fusion tracking, mapping and loop closure detection, and global optimization processes to improve the algorithm efficiency. Moreover, we will explore the fast camera pose recovery method to improve the tracking robustness further.

Acknowledgements We would like to acknowledge the experimental assistance provided by Xiaozhou Dong and language editing assistance provided by Marc D. Baldwin, PhD.

Funding information This study received financial support provided by "The Fundamental Research Funds for the Central Universities" of China (3102015BJ(II)MYZ21).

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

References

1. Liu JH (2011) Digital assembly technology in military industry. Def Manuf Technol 4:5–7
2. Bortolini M, Faccio M, Gamberi M, Pilati F (2017) Multi-objective assembly line balancing considering component picking and ergonomic risk. Comput Ind Eng 112:348–367
3. Faccio M (2014) The impact of production mix variations and models varieties on the parts-feeding policy selection in a JIT assembly system. Int J Adv Manuf Technol 72(1–4):543–560
4. Finetto C, Faccio M, Rosati G, Rossi A (2014) Mixed-model sequencing optimization for an automated single-station fully flexible assembly system (F-FAS). Int J Adv Manuf Technol 70(5–8):797–812
5. Faccio M, Gamberi M, Pilati F, Bortolini M (2015) Packaging strategy definition for sales kits within an assembly system. Int J Prod Res 53(11):3288–3305
6. Hu SJ, Ko J, Weyand L, ElMaraghy HA, Lien TK, Koren Y, Bley H, Chryssolouris G, Nasr N, Shpitalni M (2011) Assembly system design and operations for product variety. CIRP Ann Manuf Technol 60(2):715–733
7. Hu SJ, Zhu X, Wang H, Koren Y (2008) Product variety and manufacturing complexity in assembly systems and supply chains. CIRP Ann Manuf Technol 57(1):45–48
8. Wang QH, Huang ZD, Ni JL, Xiong W, Li JR (2016) A novel force rendering approach for virtual assembly of mechanical parts. Int J Adv Manuf Technol 86(1–4):977–988
9. Liu Z, Tan J (2007) Constrained behavior manipulation for interactive assembly in a virtual environment. Int J Adv Manuf Technol 32(7–8):797–810
10. Chen J, Mitrouchev P, Coquillart S, Quaine F (2017) Disassembly task evaluation by muscle fatigue estimation in a virtual reality environment. Int J Adv Manuf Technol 88(5–8):1523–1533
11. Kyriazis N, Argyros A (2014) Scalable 3D tracking of multiple interacting objects. IEEE Conference on Computer Vision and Pattern Recognition, pp 3430–3437
12. Tombari F, Franchi A, Stefano L D. (2014) BOLD features to detect texture-less objects. IEEE International Conference on Computer Vision, pp 1265–1272
13. Wang Y, Zhang S, Yang S, He W, Bai X, Zeng Y (2017) A linemod-based markerless tracking approach for AR applications. Int J Adv Manuf Technol 89(5–8):1699–1707
14. Engel J, Stückler J, & Cremers D (2015) Large-scale direct SLAM with stereo cameras. IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 1935–1942
15. Mengyin F, Xianwei L, Tong L, Yi Y, Li X, Yu L (2015) Real-time slam algorithm based on RGB-D data. Robot 6(37):683–692
16. Henry P, Krainin M, Herbst E, Ren X, & Fox D (2014) RGB-D mapping: using depth cameras for dense 3D modeling of indoor environments. In the 12th International Symposium on Experimental Robotics, pp 647–663
17. Garon M, Lalonde JF (2017) Deep 6-DOF tracking. IEEE Trans Vis Comput Graph 23(11):2410–2418
18. Tan D J, & Ilic S (2014) Multi-forest tracker: a chameleon in tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1202–1209
19. Joseph Tan D, Tombari F, Ilic S, & Navab N (2015) A versatile learning-based 3d temporal tracker: scalable, robust, online. In Proceedings of the IEEE International Conference on Computer Vision, pp 693–701
20. Besl PJ, Mckay ND (2002) Method for registration of 3-D shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256
21. Rusinkiewicz S, Levoy M (2001) Efficient variants of the ICP algorithm. 3DIM. IEEE Computer Society, pp 145
22. Newcombe R A, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison A J, … & Fitzgibbon A (2011) KinectFusion: real-time dense surface mapping and tracking. 10th IEEE international symposium on Mixed and augmented reality (ISMAR), pp 127–136
23. Izadi S, Kim D, Hilliges O, Molyneaux D, Newcombe R, Kohli P, … & Fitzgibbon A (2011). KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In Proceedings of the 24th annual ACM symposium on User interface software and technology, pp 559–568
24. Izadi S, Newcombe R A, Kim D, Hilliges O, Molyneaux D, Hodges S, … & Fitzgibbon A (2011) Kinectfusion: real-time dynamic 3d surface reconstruction and interaction. In ACM SIGGRAPH, pp 23
25. Audras C, Comport A I, Meilland M, & Rives P (2016) Real-time dense RGB-D localisation and mapping. Australian Conference on Robotics and Automation, pp 1–10
26. Stuckler J, Behnke S (2012) Integrating depth and color cues for dense multi-resolution scene mapping using RGB-D cameras. Multisensor Fusion and Integration for Intelligent Systems, pp 162–167
27. Endres F, Hess J, Engelhard N, Sturm J, Cremers D, & Burgard W (2012) An evaluation of the RGB-D SLAM system. 2012 IEEE International Conference on Robotics and Automation (ICRA), pp 1691–1696
28. Whelan T, Johannsson H, Kaess M, Leonard J J, & McDonald J (2012) Robust tracking for real-time dense RGB-D mapping with Kintinuous. Technical Report, (Query date: 5-13-2018.)
29. Henry P, Krainin M, Herbst E, Ren X, Fox D (2014) RGB-D mapping: using depth cameras for dense 3D modeling of indoor environments. Experimental Robotics. Springer, Berlin Heidelberg, pp 647–663
30. Steinbrucker F, Kerl C, & Cremers D (2013) Large-scale multi-resolution surface reconstruction from RGB-D sequences. In Proceedings of the IEEE International Conference on Computer Vision, pp 3264–3271
31. Whelan T, Kaess M, Leonard J J, & McDonald J (2013) Deformation-based loop closure for large scale dense RGB-D SLAM. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 548–555
32. Whelan T, Leutenegger S, Salas-Moreno R, Glocker B, & Davison A (2015) ElasticFusion: dense SLAM without a pose graph. Robotics: Science and Systems, pp 1–9
33. Fioraio N, Taylor J, Fitzgibbon A, Di Stefano L, & Izadi S (2015) Large-scale and drift-free surface reconstruction using online

subvolume registration. IEEE Conference on Computer Vision and Pattern Recognition, pp 4475–4483

34. Glocker B, Shotton J, Criminisi A, Izadi S (2015) Real-time RGB-D camera relocalization via randomized ferns for keyframe encoding. IEEE Trans Vis Comput Graph 21(5):571–583

35. Valentin J, Nießner M, Shotton J, Fitzgibbon A, Izadi S, & Torr P H (2015) Exploiting uncertainty in regression forests for accurate camera relocalization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4400–4408

36. Matsuo T, Fukushima N, Ishibashi Y (2013) Weighted joint bilateral filter with slope depth compensation filter for depth map refinement. VISAPP 2:300–309

37. Rublee E, Rabaud V, Konolige K, & Bradski G (2011) ORB: an efficient alternative to SIFT or SURF. 2011 IEEE International Conference on Computer Vision (ICCV), pp 2564–2571

38. Bian J, Lin WY, Matsushita Y, Yeung SK, Nguyen TD, & Cheng MM (2017) GMS: grid-based motion statistics for fast, ultra-robust feature correspondence. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2828–2837

39. Mur-Artal R, Montiel JMM, Tardos JD (2015) ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Trans Robot 31(5):1147–1163

40. Li S, Xu C, Xie M (2012) A robust O (n) solution to the perspective-n-point problem. IEEE Trans Pattern Anal Mach Intell 34(7):1444–1450