J Inf Process Syst, Vol.21, No.2, pp.163~179, April 2025
https://doi.org/10.3745/JIPS.04.0342
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Research on the Tracking Registration Method of Markerless Augmented Reality for Product Assembly

Pengxia Cao* and Yibo Huang

Abstract

To solve the problems of assembly object variety, volume difference, and texture features not being obvious in the intelligent products assembly process, augmented-reality technology was employed. To improve the feasibility and robustness of the augmented-reality tracking registration algorithm in the product assembly scenarios, combined with the weak texture of product assembly operation objects and the complex scenes, a real-time and robust unmarked tracking registration method was proposed. First, the multimodal approach of LINEMOD was improved to detect the weakly textured assembly target base so that the rough pose of the camera can be obtained and the problem of reinitialization can be solved when the process of tracking registration is interrupted. Then, the iterative closest point (ICP) registration algorithm was improved to accelerate the registration process while accurately positioning the camera pose to determine the benchmark for virtual-real fusion. Finally, based on the pose relationship between the assembly and the assembly guidance information established in the offline phase, visual guidance of the assembly process was realized. A performance analysis and comparison of the proposed tracking registration method show that the proposed method has better accuracy than the LINEMOD template matching method. The visual display of the assembly guidance process shows that the proposed method can be applied to the assembly scene of products lacking texture to realize intelligent assembly.

Keywords

Augmented Reality (AR), Iterative Closest Point (ICP) Registration, LINEMOD Template Matching, Product Assembly, Tracking Registration Method

1. Introduction

Intelligent manufacturing has become a global issue and a national-level strategic issue. Many countries have made plans and deployments in the field of smart manufacturing, such as “Made in China 2025,” “Industry 4.0 Platform,” and “Industrial Internet Plan” [1]. Intelligent product assembly is the deep integration of advanced assembly technology and information technology, and has become the main trend in current product assembly development. At present, most assembly work in industrial production processes is performed manually by on-site assembly personnel according to the assembly process information file. The manual assembly of complex products has the following problems [2]. 1) The assembly process is complicated. There are many problems, such as many parts, volume difference, and texture features not being obvious, and the assembly technology is difficult. 2) The assembly process

Manuscript received December 21, 2022; first revision September 25, 2023; accepted November 18, 2023.
* **Corresponding Author:** Pengxia Cao (316657294@qq.com)
College of Physics and Electronic Engineering, Northwest Normal University, Lanzhou, China (316657294@qq.com, 514538910@qq.com)

mainly depends on the familiarity of the assembly personnel with the assembly process, which is inefficient and prone to errors. 3) During the product assembly process, assemblers must switch their attention back and forth between the operational field-of-view and the process documentation. The discontinuity of the assembly work makes it difficult for the assemblers to accumulate experience effectively. The emergence of augmented-reality technology can solve these problems. The use of augmented-reality technology to superimpose the physical characteristics of virtual information and the three-dimensional (3D) real environment in real time enables assemblers to receive specific assembly guidance information in real time in the real assembly environment [3].

Augmented-reality technology makes use of computers to enhance the digital scene with all kinds of information in the real world that users can see. It can superimpose the real assembly scene environment and virtual assembly guidance information, such as 3D models, videos, and graphics, in real time into the same picture or space, thereby improving the efficiency and quality of assembly operations [4]. An accurate 3D tracking and registration method is one of the core technologies of an augmented-reality-based product assembly assistance system. It can accurately fuse virtual assembly guidance information in the assembly environment. In recent years, several unmarked tracking registration methods have been developed. However, many problems and challenges remain when researching an unmarked augmented-reality tracking registration method for product assembly [5]. First, many types of component are involved in product assembly scenes, most of which are objects with a lack of texture on the surface. This leads to tracking jitter or disturbance when using the tracking registration method based on natural features. Second, the background environment of the product assembly scene is cluttered, and the assembly objects are easily occluded by the background or other components, leading to poor robustness when using the edge-based algorithm. The method based on the model can solve the problem of objects being difficult to detect owing to a lack of texture [6]. However, this method must accommodate a large number of reference images, making it difficult to meet the real-time requirements of product assembly auxiliary systems. Therefore, the availability of this method is reduced. The method based on point cloud registration has low requirements for environmental lighting intensity and texture features of the object surface, so it can be applied to product assembly scenes [7]. This kind of method usually uses an iterative closest point (ICP) registration algorithm to calculate all or part of the point clouds iteratively, and the algorithm can estimate an accurate pose, even in a scene with poor ambient light conditions. However, the ICP algorithm is prone to falling into local optimality and even causing “frame loss” when the depth sensor moves fast and cannot obtain the correct initial point cloud dataset, thus interrupting the tracking registration process.

To address the application scenario of product assembly, an unmarked 3D tracking registration method was developed to meet the requirements of both precision and real-time performance. This method combines the advantages and disadvantages of LINEMOD and the ICP algorithm. In the offline phase, according to the 3D model of the base of the product assembly object, reference images under different perspectives are obtained, and the gradient direction and surface normal vector features of the assembly object are obtained and saved. In the online phase, LINEMOD is improved to obtain the keyframe, the camera posture is estimated roughly, and the search space of the ICP algorithm is narrowed. The ICP algorithm is then improved to position the camera accurately, thereby determining the positioning benchmark of the virtual assembly guidance information. In the precise positioning of the camera pose, when the tracking is interrupted because of fast camera movement, LINEMOD can be used again to obtain a new keyframe and a rough pose of the assembly object.

2. System Overview

The overall framework of the product assembly system is illustrated in Fig. 1. In the offline training phase, the 3D model of the assembly is utilized in a computer-aided design (CAD) environment to generate reference images of the assembly object from multiple perspectives, thereby establishing an assembly guidance information database. During this acquisition process, the camera's positional coordinates and orientation are recorded to determine the spatial relationship between the assembly base and the assembly guidance information. The orientation gradients and normal vector features extracted from the assembly base images across different viewpoints are processed and stored during training. In the online phase, real-time video frames of the product assembly operation scene are captured by the camera. The depth image of each frame is repaired using an improved fast marching method (FMM) algorithm. Subsequently, orientation gradient features are extracted from the color images while normal vector features are derived from the processed depth images. An improved LINEMOD method is applied to identify the most similar keyframe and the corresponding camera pose from the reference images. Following this, an optimized ICP algorithm is utilized to register the model point clouds with the environment point clouds, enabling accurate camera pose estimation during the tracking and registration processes, thus determining the pose information of the assembly. Finally, leveraging the pre-established spatial relationship between the assembly base and guidance information, augmented reality visualization is achieved by superimposing assembly guidance data onto the physical environment. If virtual-real fusion fails due to excessive camera movement velocity or excessive mean distance between matched point clusters, the improved LINEMOD method can be reapplied to obtain a new keyframe and an updated initial pose.

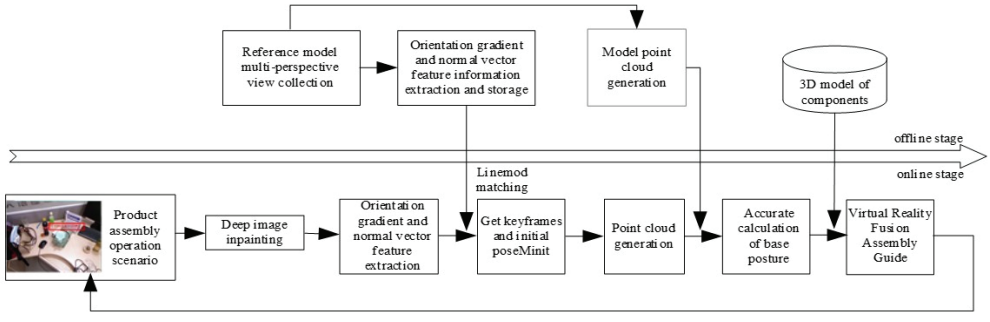


Fig. 1. Overall system block diagram.

3. Depth Image Inpainting and Point Cloud Generation

The quality of the depth images directly impacts the extraction of normal vector features and the accuracy of point cloud data generation, thereby influencing camera pose estimation results. Currently, there are two primary methods for acquiring scene depth information [8].

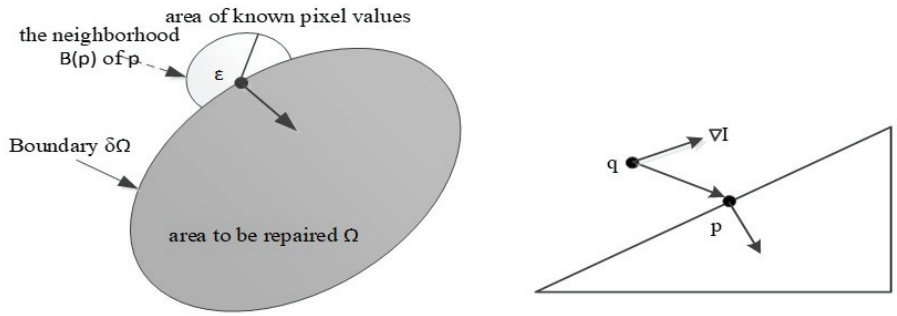
1) Stereo matching algorithms: Depth information is derived by calculating object parallax from binocular camera images using stereo matching and converting it to depth measurements through camera parameters. While theoretically viable, this approach suffers from high computational complexity and significantly degraded performance in occluded scenarios.

2) Direct depth sensing: Depth maps are acquired using dedicated ranging devices such as depth cameras. However, depth maps captured by these sensors often suffer from missing depth information on low-texture object surfaces and in the occluded regions, resulting in holes. Therefore, it is necessary to fill and inpaint these depth maps obtained from the depth camera [9]. In this paper, we employ an improved FMM to inpaint the depth map, thereby obtaining a high-quality depth map with excellent overall smoothness and well-defined edges. Based on this, we extract surface normal vector features and generate a point cloud dataset.

3.1 Improved Fast Marching Method for Depth Image Inpainting

The use of separate cameras for color and depth data acquisition introduces misalignment between the two modalities due to differing coordinate origins. This discrepancy results in imperfect spatial correspondence between the captured color images and depth maps. Prior to depth image inpainting, geometric calibration must therefore be performed to align the color and depth data within a unified coordinate system [10]. This spatial alignment is critical for enabling accurate target object identification and subsequent pose estimation [11].

The FMM operates by treating the damaged region in an image as an inpainting template. The algorithm first processes pixels along the boundary of this template and progressively propagates inward until all pixels within the template are inpainted [12]. As illustrated in Fig. 2(a), let Ω represent the damaged region, δΩ denote its boundary, and p be a specific pixel on δΩ. The following steps outline the principle of the FMM algorithm using pixel p as an exemplar.



(a) (b)

Fig. 2. Principle of FMM image inpainting: (a) the mathematical model and (b) $p-q$ point graph.

The neighborhood $B_\varepsilon(p)$ of the pixel p with radius ε in the image is selected, and N represents the gradient direction of point p. Assuming that point q in Fig. 2(b) is a pixel in the neighborhood $B_\varepsilon(p)$, the pixel value of point p is determined by the pixel q with the smallest spatial distance from p within $B_\varepsilon(p)$. The pixel value of p can be estimated via the first-order Taylor expansion, which can be expressed as Eq. (1):

$$I_q(p) = I(q) + \nabla I(q) \cdot (p - q), \tag{1}$$

where I represents the depth map to be inpainted, $I(q)$ represents the pixel value at point q, $\nabla I(q)$ is the gradient at point q, and $I_q(p)$ is the first-order Taylor approximation at p.

The mathematical model of the FMM algorithm is derived by assigning a weight $w(p,q)$ to all pixels in the neighborhood $B_\varepsilon(p)$, and normalizing the weighted sum of the first-order Taylor estimates, as shown in Eq. (2):

$$I(p) = \frac{\sum_{q\in B_\varepsilon(p)} w(p,q) I_q(p)}{\sum_{q\in B_\varepsilon(p)} w(p,q)}. \tag{2}$$

The weight $w(p,q)$ comprises three parameters: the direction parameter $dir(p,q)$, the geometric distance parameter $dst(p,q)$, and the horizontal distance parameter $lev(p,q)$.

$$w(p,q) = dir(p,q) \cdot dst(p,q) \cdot lev(p,q), \tag{3}$$

where $dir(p,q) = \frac{p-q}{\|p-q\|} \cdot N(p)$, $dst(p,q) = \frac{d_0^2}{\|p-q\|^2}$, $lev(p,q) = \frac{T_0}{1+|T(p)-T(q)|}$.

Here d_0, T_0 are empirically set to 1. The direction parameter $dir(p,q)$ ensures higher weights for pixels nearer to the normal direction of point p, the geometric distance parameter $dst(p,q)$ prioritizes spatially closer pixels, and the horizontal distance parameter $lev(p,q)$ emphasizes pixels nearer to the boundary $\delta\Omega$. T represents the distance from the pixel in the area Ω to the boundary $\delta\Omega$.

To inpaint the entire area Ω, Eq. (2) is iteratively applied to boundary pixels while updating the boundary information of Ω until all the pixels in Ω are inpainted. The FMM algorithm employs the fast-marching method to complete the boundary information of Ω, so that the points on the boundary gradually advance into Ω. In general, the repair order is rotated by solving the Eikonal equation for the pixels within Ω, which is

$$|\nabla T| = 1, \tag{4}$$

where the points on the boundary $\delta\Omega$ have a T value of 0. The FMM algorithm selects the inpainting order according to the magnitude of the T value. The smaller the T value is, the earlier the corresponding pixel is inpainted. In the FMM algorithm, all pixel values can be calculated to obtain the corresponding T value. First, all pixels are divided into three categories:

1) Boundary pixels on $\delta\Omega$ with unupdated T values.

2) Known pixels outside $\delta\Omega$ that require no repair, whose T values are predetermined.

3) Target pixels within the repair area Ω, where T values need computation.

Therefore, during depth map restoration, the initial T values are usually set as follows: Boundary pixels on $\delta\Omega$ and known pixels outside $\delta\Omega$ are set to 0, pixels within the inpainting area Ω are initialized with infinite (size is 10^6). When repairing, the algorithm will: prioritize boundary pixels with the smallest T value. Propagate the boundary inward. Update T values using the following linear difference equation:

$$max(D^{-x}T, -D^{+x}T, 0)^2 + max(D^{-y}T, -D^{+y}T, 0)^2 = 1, \tag{5}$$

where $D^{-x}T(i,j) = T(i,j) - T(i-1,j)$ represents the forward difference in the x-direction, and $D^{+x}T(i,j) = T(i+1,j) - T(i,j)$ represents the backward difference in the x-direction. The y-direction is similar to the x-direction.

The FMM algorithm determines the inpainting order based on the distance between the pixel point and the boundary $\delta\Omega$. While this method achieves good results for color images, depth images often contain large hollow regions [13]. Solely relying on spatial distance for inpainting order leads to limited accuracy. To address this, we propose weighting pixels with larger depth values more heavily and reducing weights for edge pixels. The revised weight coefficient is defined as follow equation:

$$P(p) = -T(p) + \frac{\lambda_1 D(p)}{D_{max}} - \lambda_2 |\nabla^2 I(p)|, \tag{6}$$

where $T(p)$ is the distance from the pixel p to the boundary $\delta\Omega$, $D(p)$ is the depth value at p, D_{max} is the maximum depth value in the depth image, and $\nabla^2 I(p)$ is the data coefficients obtained after applying the Laplace transform to the corresponding color image. By performing the Laplace transform on the color image, a significant output result can be achieved in the edge regions, thereby reducing the weight coefficient of pixels in these regions. Parameters λ_1 and λ_2 are used to balance the weight contributions.

3.2 Point Clouds Generation

Let $p = (x, y)$ denote a pixel in the inpainted depth map at time t. Using NVIDIA CUDA parallel computing platform, each p is back-projected into the spatial coordinate system of the depth camera to obtain the vertex map through the perspective projection model [14].

$$V_t(p) = D_t(p) M_{int\_D}^{-1} [p, 1], \tag{7}$$

where $D_t(p)$ represents the depth map at time t, and $M_{init\_D}^{-1}$ represents the depth camera intrinsic matrix.

The camera pose $M_{init} = [R_{init} | t_{init}]$ at time t can be obtained by the rough estimation method of camera pose, where $R_{init} \in SO(3)$ is the rotation matrix and $t_{init} \in R^3$ is the translation vector. Using the initial camera pose, vertices are transformed into the global coordinate system.

$$V_{3D}(p) = V_t(p) M_{init}. \tag{8}$$

Then, using the adjacent projection points, the normal vector $N_{3D}(p)$ corresponding to each vertex can be get by formula (9).

$$N_{3D}(p) = \big(V_{3D}(x+1, y) - V_{3D}(x, y)\big) \times \big(V_{3D}(x, y+1) - V_{3D}(x, y)\big). \tag{9}$$

The raw point cloud datasets are constructed by normalizing the normal vectors $N_{3D}(p)$. To accelerate registration, the template point cloud $P = \{p_1, p_2, \cdots, p_{N_p}\}$ can be generated through voxel grid down-sampling. Using the same method, the environmental point cloud $X = \{x_1, x_2, \cdots, x_{N_x}\}$ can be obtained by aligning the world coordinate system with the camera coordinate system. Then the ICP algorithm is optimized and the point cloud datasets are registered thereby achieving accurate pose estimation.

4. Rough Estimation of Camera Pose

4.1 Offline Training

During the offline training phase for product assembly, CAD models of assembly bases are applied to achieve multi-view data acquisition, generating RGB-D datasets for training [15]. The multi-view data acquisition process can be described as follows. In the CAD environment, projection views of assembly object are captured from various perspectives using a virtual camera, with each view containing distinct assembly object geometries and corresponding camera poses. In order to avoid view redundancy caused by high sampling density near hemispherical poles, sampling is performed across the surface of a regular icosahedron. Each icosahedron face undergoes iterative subdivision into four equal parts. In order to balance the operation speed and accuracy, the iterations count is set to 2, so each face can form 16

equilateral triangles. The CAD model of the assembly base is positioned at the icosahedron's centroid, with the optical axis of the virtual camera always passing through the center, and sampling viewpoints are located at the vertex of each triangle.

Fig. 3 illustrates this offline sampling process. The longitudinal sampling range spans 0°–360°, while atitudinal sampling covers 0°–90°. For each viewpoint, four rotational angles between -45° to 45° are established to address online processing mismatches caused by rotation invariance limitations in the LINEMOD multimodal template matching method.

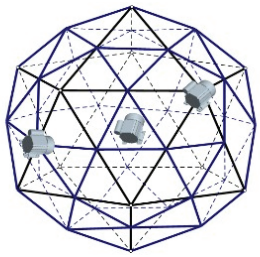


Fig. 3. Schematic of offline sampling.

4.2 Feature Descriptor Extraction

Hinterstoisser et al. [16] proposed a high-efficiency LINEMOD multimodal template matching algorithm in 2011, which integrates RGB and depth data for recognizing textureless objects in cluttered environments. The LINEMOD implementation comprises two phases:

1) Offline training phase: Reference images from multiple perspectives are obtained using the aforementioned method, with the features of each view extracted and stored.

2) Online matching phase: Similarity between video frames and template images is calculated through the similarity verification formula. Unlike a conventional template matching algorithm, LINEMOD defines template T by combining RGB gradient orientations with surface normal vectors of depth images during feature extraction [17].

$$T = (\{O_m\}_{m\in M}, P(r,m)), \tag{10}$$

where $\{O_m\}_{m\in M}$ represents multi-perspective target views, M denotes a variety of modes, specifically RGB images or depth images. P indicates feature lists at position r for modality m. So, template T is used to record the features at position r in the image. For multi-modal feature selection, LINEMOD only selects gradient directions along object contours and surface normal vectors within object boundaries. This configuration ensures robustness against illumination variations and noise. The template features of the LINEMOD algorithm are illustrated in Fig. 4.

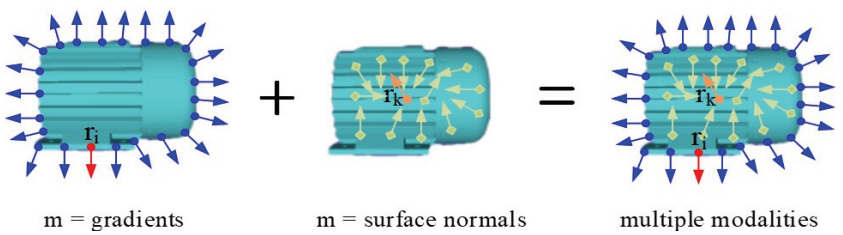


Fig. 4. Template features of LINEMOD algorithm.

4.2.1 Gradient direction feature extraction

For RGB images, maximum gradient directions across red (R), green (G) and blue (B) channels are calculated at contour positions [18]. The gradient direction of the RGB image I at the position x can be calculated by the following equation:

$$I_g(x) = ori\left(\hat{C}(x)\right) = \frac{\partial \hat{C}}{\partial x}, \tag{11}$$

where $\hat{C}(x)) = arg\ \underset{C\in\{R,G,B\}}{max} \left\|\frac{\partial C}{\partial x}\right\|$. {R,G,B} respectively represent the three color channels at position x in the color image I. $ori()$ represents gradient direction.

Due to the influence of background interference and lighting variations, the gradient direction in object contour pixels may be reversed. To quantify these gradient vectors and enhance robustness against illumination and background noise, the gradient directions can be discretized as follows (Fig. 5).

1) Collinear direction unification: Collinear directions are treated as equivalent, unifying all gradient orientations within a range of 0°–180° [19].

2) Direction quantization: The unified range is divided into 5 bins, encoded in binary. For example, the red gradient direction in Fig. 5 falls into the second bin, represented as 00010.

3) Noise mitigation: Quantized gradient directions are propagated across a 3×3 neighborhood to improve noise resistance.

4) Threshold filtering: Only gradient directions exceeding a predefined threshold are retained at each pixel position.

To accelerate online matching, gradient directions along color image contours are encoded using this binary scheme, and response mapping list needs to be calculated in advance.

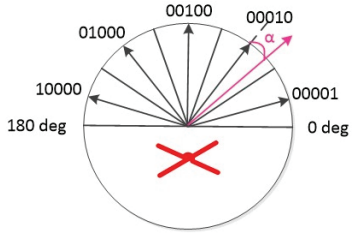


Fig. 5. Gradient direction quantization.

4.2.2 Surface normal vector feature extraction

For each pixel x in a certain depth image, the depth function $D(x)$ is approximated using a first-order Taylor expansion.

$$D(x + dx) - D(x) = dx^T \nabla D + h.o.t. \tag{12}$$

The optimal gradient $\widehat{\nabla D}$ can be obtained via the least squares method, enabling the computation of surface normal vectors for pixel x assumed to lie on a 3D plane. Assuming that this plane pass through three non-collinear points X, X_1 and X_2 [20]:

$$\begin{cases} X = \vec{v}(x)D(x) \\ X_1 = \vec{v}(x + [1,0]^T)D(x + [1,0]\widehat{\nabla D}) \\ X_2 = \vec{v}(x + [0,1]^T)D\left(x + [0,1]\widehat{\nabla D}\right). \end{cases} \tag{13}$$

In the equations above, the line of sight vector $\vec{v}(x)$ passing through pixel x is calculated using the intrinsic parameters of the depth camera that generated the depth images. The surface normal at pixel x is then obtained by calculating the cross product and normalization of vectors $\overrightarrow{XX_1}$ and $\overrightarrow{XX_2}$. As described in Section 2, depth maps from depth cameras often exhibit edge artifacts such as holes. Depth values may exhibit abrupt discontinuities at object boundaries. These discontinuities can lead to unreliable normal vector estimates at object boundaries. To address this, the LINEMOD algorithm discards pixels in a central pixel's neighborhood where depth differences exceed a predefined threshold. This method not only effectively suppresses noise in depth maps but also reliably computes corresponding normal vectors in the case of depth discontinuities.

During the offline training phase, gradient direction features from RGB images and surface normal vector features from depth maps are extracted. These features are back-projected onto the 3D assembly base model to establish their corresponding 3D space coordinates. The poses, object templates, feature vectors and 3D space coordinates corresponding to each reference view are stored in an XML file.

4.3 Rough Estimation of Camera Pose

The LINEMOD algorithm detects assembly base objects in video frames by calculating the similarity score between frame images and multi-perspective reference images. For a video frame image I and reference image T, their similarity at a certain position c can be computed using Eq. (14):

$$\varepsilon(\{I_m\}_{m\in M}, T, c) = \sum_{(r,m)\in P} \left(\max_{t\in R(c+r)} f_m\big(O_m(r), I_m(t)\big) \right), \tag{14}$$

where $f_m(O_m(r), I_m(t))$ calculates the similarity between reference images and frame image under modality m, $t \in R(c+r) = [c+r-\frac{\tau}{2}, c+r+\frac{\tau}{2}] \times [c+r-\frac{\tau}{2}, c+r+\frac{\tau}{2}]$ is a region within neighborhood region τ centered at $c+r$.

For RGB images, the similarity can be computed by the follow equation:

$$f_m\big(O_m(r), I_m(t)\big) = |O_m(r)^T I_m(t)|, \tag{15}$$

where $O_m(r)$ denotes the gradient direction at position r in the reference image, and $I_m(t)$ represents the gradient direction at position t in the video frame image. To enhance robustness against lighting variations and partial occlusions, the absolute value of the cosine between the gradient direction difference is adopted for similarity computation. Therefore, the similarity for RGB images is formulated as follows [19]:

$$\varepsilon(I, T, c) = \sum_{(r,m)\in P} \left(\max_{t\in R(c+r)} f_m\big(O_m(r), I_m(t)\big) \right) = \sum_{r\in P} \max_{t\in R(c+r)} |cos(\,ori(O,r) - ori(I,t))|, \tag{16}$$

where $ori(O,r)$ represents gradient directions at position r in reference image O. $ori(I,t)$ represents gradient directions in frame image I at position t. P denotes a list at position r, and $T = (O, P)$.

As evident from Eq. (16), the LINEMOD similarity computation inherently lacks scale invariance because it disregards relative size variations between the reference and query objects. To resolve this limitation, we integrate depth information into the similarity calculation. The enhanced similarity metric incorporates scale-aware terms derived from depth maps.

$$\begin{cases} \varepsilon(I,T,c_I') = \sum\limits_{(c_O'+r')\in p} \max\limits_{t\in R(S_I(c_I',r'))} |cos[\,ori(O,S_O(c_O',r')) - ori(I,t)]| \\ S_x(c_x',y) = c_x' + \dfrac{D(c_x')}{D(c_O')}y \end{cases}, \tag{17}$$

where $ori(O,S_O(c_O',r'))$ represent gradient directions at position $c_I' + \frac{D(c_I')}{D(c_O')}r'$ in reference image O. $t \in R(S_I(c_I',r')) = [c_I' + \frac{D(c_I')}{D(c_O')}r' - \frac{\tau}{2}, c_I' + \frac{D(c_I')}{D(c_O')}r' + \frac{\tau}{2}] \times [c_I' + \frac{D(c_I')}{D(c_O')}r' - \frac{\tau}{2}, c_I' + \frac{D(c_I')}{D(c_O')}r' + \frac{\tau}{2}]$ represents a region within neighborhood region τ centered at $c_x' + \frac{D(c_x')}{D(c_O')}r'$, $D(c_x')$ represents the depth value at the pixel point c_x'. $D(c_O')$ is the distance between the reference image O and the center of the regular polyhedron during offline phase.

With depth information integrated into the similarity calculation, the positions of the assembly base captured from multiple perspectives during the offline training phase are known. This enables estimation of the scaling factor for the assembly base using the above formula. A sliding window is then traversed across the video frame image. The similarities across all positions in list P are aggregated to ensure the similarity between the video frame and the assembly base reference image from a specific perspective. After iterating through all assembly base reference images, the keyframe should be the image that is most similar to the current video frame. This process can ensure the rough position $M_{init} = [R_{init}|t_{init}]$ of the current camera and reduce the algorithm search space for accurate calculation of pose.

5. Accurate Estimation of Camera Pose

Combined with the aforementioned point cloud dataset generation method and the initial camera pose $M_{init} = [R_{init}|t_{init}]$, this paper adopts a tracking method based on point cloud registration to refine the camera pose. Let the point cloud datasets of the assembly model and the video frame be denoted as $P = \{p_1, p_2, \cdots, p_{N_p}\}$ and $X = \{x_1, x_2, \cdots, x_{N_x}\}$. The ICP algorithm is employed to register these point clouds and compute the accurate camera pose. In order to improve the robustness of the ICP algorithm, KD-tree is utilized for nearest-neighbor search during corresponding point search, and GPU is applied to expedite point cloud registration. Besides, RANSAC eliminates outlier point pairs to improve registration accuracy. If registration fails due to the average distance between matched point pairs is too far, the LINEMOD algorithm is reinitialized to get a new keyframe and initial pose.

5.1 Classic ICP Algorithm

The ICP algorithm is a classical method for estimating relative pose using point cloud dataset [21]. The algorithm iteratively searches for nearest-neighbor points to establish correspondence pairs and minimizes an objective function constantly to estimate the relative pose between the two cloud points. Specifically, it is assumed that two point clouds to be registered is P and X, and a subset of points $\{p_i\}$ is selected from P. For each point p_i in this subset, the nearest x_i in X is identified. The objective function in Eq. (18) is minimized to compute the pose transformation that best aligns these corresponding points [22]:

$$f(R,T) = \frac{1}{k}\sum_{i=1}^{k} \|x_i - (Rp_i + T)\|^2, \tag{18}$$

where R and T respectively represent the rotation matrix and translation matrix.

The essence of ICP algorithm is to estimate the relative pose between two point clouds by calculating the optimal transformation matrix that minimizes alignment error. This is achieved through iterative adjustments to R and T. The point clouds to be registered in this work are: Template point cloud $P = \{p_1, p_2, \cdots, p_{N_p}\}$. A predefined reference model of the assembly base. Scene point cloud $X = \{x_1, x_2, \cdots, x_{N_x}\}$. Point cloud data extracted from the video frame. The classical ICP algorithm registers two point clouds through the following steps [23].

Step 1: Select a subset of points $\{p_i\}$ from the assembly base template point cloud P, namely $p_i \in P$.

Step 2: For each point p_i, search for its nearest x_i in the scene point cloud X using the shortest Euclidean distance, establishing initial correspondences between P and X.

Step 3: Apply an algorithm to eliminate erroneous correspondence pairs.

Step 4: The objective function in Eq. (18) is minimized to compute the pose transformation matrix between $\{p_i\}$ and $\{x_i\}$. Apply the calculated transformation matrix to update $\{p_i\}$, yielding a new subset of points $\{p_i'\}$, $\{p_i' = Rp_i + T, p_i \in P\}$.

Step 5: Calculate registration error using $d = \frac{1}{n}\sum_{i=1}^{n}\|p_i' - x_i\|^2$. Set a threshold ε, if $d > \varepsilon$ and the iteration count is below the maximum limit, go back to Step 2 to continue iteration. If $d \leq \varepsilon$, output R and T as optimal pose. If the maximum iterations are reached without convergence, the LINEMOD algorithm will be reinitialized to get a new keyframe and initial pose.

5.2 Improved ICP Algorithm

Compared to other registration algorithms, the ICP algorithm does not require explicit segmentation of point clouds and directly utilizes corresponding points for registration. Due to its excellent registration effect, ICP is widely adopted as a precise registration method [24]. However, it faces two key limitations.

1) The ICP algorithm requires exhaustive traversal of all points in the point cloud during nearest-neighbor searches. For dense point clouds with a large number of internal points, this severely impacts computational efficiency.

2) The algorithm is prone to incorrect correspondence pairs during nearest-neighbor matching, which can degrade registration accuracy [25].

To solve these problems, we propose the following enhancements.

1) Reduces point cloud density by using voxel rasterization method during the acquisition process.

2) A KD-tree of the point cloud data is constructed on the GPU to enhance computational efficiency during nearest-neighbor searches.

3) The RANSAC algorithm with a predefined distance threshold is used to eliminate erroneous point pairs.

The improved ICP algorithm operates as follows.

Step 1 (Data processing): Simplify the two input point cloud datasets P and X using the voxel grid method, resulting in simplified point clouds $P'$and X'.

Step 2 (Accelerated correspondence search): For each point p_i^k in $\{p_i\}$, Use a GPU-accelerated KD-tree to speed up the search for its corresponding point x_i^k in the environmental point cloud dataset, forming a correspondence pair (p_i^k, x_i^k).

Step 3 (Outlier removal via RANSAC): Apply the RANSAC algorithm with a predefined distance threshold to eliminate incorrect correspondence pairs from the candidate matches.

Step 4 (Rigid transformation estimation): Calculate the optimal rigid transformation parameters by minimizing the objective function in Eq. (18). Apply the transformation matrix to the point set p_i to obtain an updated point set $p_i' = \{p_i' = Rp_i + T, p_i \in P\}$.

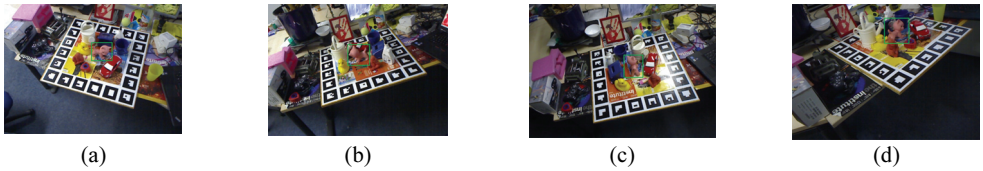
Step 5 (Termination criteria): Calculate the corresponding point pair distance according to $d = \frac{1}{n}\sum_{i=1}^{n}\|p_i' - x_i\|^2$. Set a predefined threshold ε, if $d > \varepsilon$, go back to Step 2 for further iteration. If $d \leq \varepsilon$, output the optimal relative pose. If the maximum iteration count is reached without convergence, restart the LINEMOD algorithm to acquire a new keyframe and reinitialize the camera pose.

6. Method Validation and Discussion

The proposed method was implemented using VS2017, OpenCV3.4.1, and Python3.7. The computational platform consists of an 11th Gen Intel Core i7-4600U microprocessor with a CPU frequency of 2.4 GHz, 16 GB of RAM, and an NVIDIA GeForce MX450 GPU. Color and depth images of the assembly scene were acquired using a Kinect sensor, which include an RGB sensor resolution of 640×480 pixels and a depth sensor resolution of 320×240 pixels. In the experiments, the RGB image size was set to 640×480. During the validation process, the performance of the proposed tracking registration method was first evaluated on the public weak-texture dataset introduced by Hinterstoisser et al. [16]. Subsequently the motor was used as the assembly target to demonstrate the visual guidance of the product assembly process.

6.1 Tracking Registration Method Performance Analysis

Firstly, the LINEMOD algorithm was improved to detect objects with insufficient surface texture on the open dataset. As shown in Fig. 6, the green boxes represent the detected targets under various conditions. Fig. 6(a)–6(c) depict different viewing angles, scales, and lighting conditions, while Fig. 6(d) illustrates partial occlusion of the object. So the improved LineMod algorithm can provide a relatively accurate initial pose for the registration algorithm in a cluttered environment.



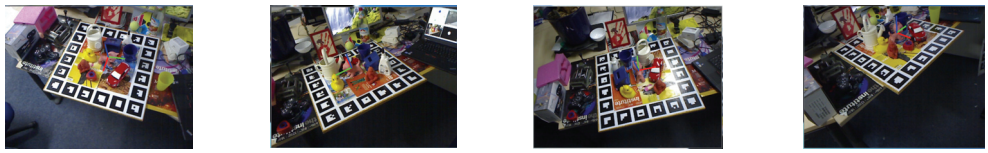
(a) (b) (c) (d)

Fig. 6. Detected objects in cluttered environment: (a–c) different viewing angles, scales, and lighting conditions, and (d) object is partially occluded.

(a) (b) (c) (d)

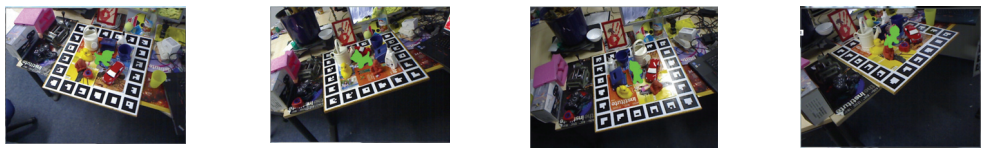
Fig. 7. The visualized effect rendered according to the initial pose: (a–c) different views and scales, and (d) the object is partially occluded.

By incorporating depth information into the computation of similarity scores, the improved LINEMOD algorithm exhibits robustness and scale invariance in cluttered environments. However, the initial pose obtained from the improved LINEMOD method does not meet the precision requirement for augmented reality product assembly systems. For instance, the green objects in Fig. 7 represent the visualization rendered based on the initial pose, which shows significant deviation. In our method, the improved LINEMOD algorithm provides a rough initial pose for the tracking algorithm based on the point cloud, followed by the application of an improved ICP algorithm for accurate pose estimation. Finally, augmented reality visualization is achieved based on the precise pose. Fig. 8 illustrates the pose information acquired using our method, and Fig. 9 depicts the visualization effect of rendering according to the exact pose. Our method achieves superior visualization results across different angles, scales, and even under partial occlusion. This is because the improved ICP method has higher registration accuracy under the condition of the rough initial pose. When tracking fails, the enhanced LINEMOD method is reactivated to get a new initial pose. Table 1 records the average time required for key steps in this method. Specifically, the target detection process is denoted as Step I, and the tracking registration process is denoted as Step II. Using the enhanced LINEMOD algorithm, it takes approximately 55.49 ms per frame to compute the initial pose on average. This process is relatively slow because it involves calculating the similarity score between the video frame image and all reference images. In contrast,



(a) (b) (c) (d)

Fig. 8. The precise pose information get by the method we proposed: (a–c) different views and scales, and (d) the object is partially occluded.



(a) (b) (c) (d)

Fig. 9. The visualized effect rendered according to the precise pose: (a–c) different views and scales, and (d) the object is partially occluded.

Table 1. Average running time per frame of the method in this paper

Process	Main steps	Time (ms)	Total time (ms)
Step I	Input the video frame image	1.05	55.49
	Feature extraction	6.75	
	Improved LINEMOD algorithm	47.69	
Step II	Input the video frame image	1.05	30.67
	Pre-processing	7.49	
	Tracking using improve ICP algorithm	13.62	
	Visualization of results	8.51	

when applying the improved ICP method, the tracking process took an average of 30.67 ms per frame. Since the augmented reality system primarily operates in the tracking state, it executes the target detection process only during initialization or upon tracking failure. Therefore, the proposed method satisfies the real-time requirements of augmented reality product assembly systems.

6.2 Assembly Guide

Using an electric motor as the assembly target, an augmented reality visualization system for product assembly was constructed. The system employs 3D Max software to model guidance information [22]. Implemented in C++ within the Python development environment, the proposed method generates dynamic link libraries callable by Python. In the experiment, a motor with its junction box and front cover removed served as the assembly base, placed in a cluttered background environment. The Kinect sensor was utilized to obtain color and depth images of the assembly scene, and the pose information of the assembly base was obtained using the proposed method. Based on the pose relationship between the assembly base and the guidance information, virtual assembly guidance information was superimposed onto the real-world scene. By moving the Kinect sensor, the registration effect of assembly guidance information can be achieved under various viewing angles and scales. Fig. 10(a) illustrates the results of the assembly base identification and registration. The green model represents the rendered and registered 3D model of the assembly base, obtained by estimating its pose using the proposed method. Fig. 10(b)–10(d) depict the guide screens for different assembly processes. The green model components in Fig. 10(b)–10(f) correspond to virtual dynamic guide information, which is utilized to determine installation positions of the components. In Fig. 10(c), the red model component provides operation animation guidance to assist assemblers in understanding the assembly process and instructions more clearly, thereby enhancing assembly efficiency. Consequently, the proposed method achieves a superior assembly guidance effect under different viewing angles and scales. Additionally, the frame size during the assembly process matches that of the aforementioned experiment. When employing the aforementioned method for assembly guidance, the average runtime per frame is 31.2 ms, slightly longer than the average runtime observed in the previous experiment. This discrepancy arises due to the larger

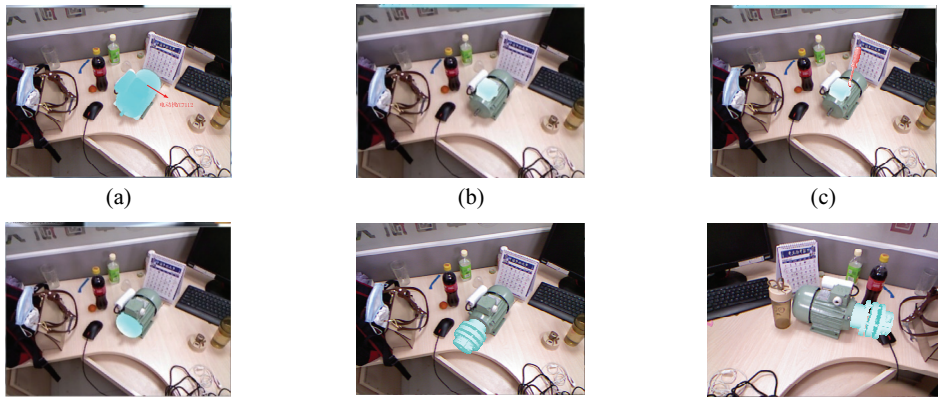


Fig. 10. Visualization of virtual assembly guidance information and assembly virtual-real fusion registration effect: (a) Motor base identification registration result. (b–d) Guide screens for different assembly processes. (e–f) The registration result of the guide information for the coupling assembly of different viewing angles and sizes.

volume of assembly objects in the product assembly experiment, which increases the computational time required for point cloud matching. Furthermore, the loading and rendering times vary depending on the complexity of the virtual information. However, the average runtime per frame during the product assembly process remains within real-time requirements.

7. Conclusion

The purpose of this study was to address the problem of poor robustness of the commonly used tracking and registration methods based on natural features, owing to the cluttered environment of product assembly scenes. The components involved were mostly weakly textured objects that are easily occluded [26]. An accurate 3D tracking registration method without markings that can satisfy real-time requirements was proposed. During product assembly, the improved linear parallel multimodal LINEMOD template matching method and improved ICP method were combined to realize the visual guidance of assembly guidance information. First, the improved LINEMOD method was applied to complete the roughness of the camera pose, which provides a more accurate initial value for the ICP registration algorithm to enhance the precision of the ICP method and reduce the search space of the ICP algorithm. Then, the improved ICP method was applied to register the assembly base model and the video frame area to determine the pose information of the assembly base. Finally, according to the pose relationship between the assembly base and the assembly guidance information, the virtual assembly guidance information was superimposed onto the assembly environment to achieve the fusion of the virtual information and the real environment. In the method verification process, the performance of the proposed tracking registration method was first verified using a public weak-texture dataset. A motor was then considered as the assembly target to achieve visual guidance for the product assembly process. The experiment shows that the proposed method has good real-time performance and robustness in such application scenarios as product assembly, where the environment is cluttered and the operating objects lack texture.

Conflict of Interest

The authors declare that they have no competing interests.

Funding

This study was supported by the Northwest Normal University Young Teachers Research Ability Enhancement Program Project (No. NWNU-LKQN2022-18).

References

[1] M. Li and G. Q. Huang, “Production-intralogistics synchronization of Industry 4.0 flexible assembly lines under graduation intelligent manufacturing system,” *International Journal of Production Economics*, vol.

241, article no. 108272, 2021. https://doi.org/10.1016/j.ijpe.2021.108272

[2] Y. Wang, S. Zhang, S. Yang, W. He, and X. Bai, "Mechanical assembly assistance using marker-less augmented reality system," *Assembly Automation*, vol. 38, no. 1, pp. 77-87, 2018. https://doi.org/10.1108/AA-11-2016-152

[3] Y. Wang, "Research on augmented reality fusion technology for product assembly guidance," Ph.D. dissertation, Northwestern Polytechnical University, Xi'an, China, 2018. https://doi.org/10.27406/d.cnki.gxbgu.2018.000140

[4] Z. Wei, Y. Guo, P. Tang, H. Li, G. Zheng, J. Pu, and R. Ji, "Research and application of augmented reality in complex product assembly," *Computer Integrated Manufacturing Systems*, vol. 28, no. 3, pp. 649-662, 2022. https://doi.org/10.13196/j.cims.2022.03.001

[5] C. Chen, Z. Tian, D. Li, L. Pang, T. Wang, and J. Hong, "Projection-based augmented reality system for assembly guidance and monitoring," *Assembly Automation*, vol. 41, no. 1, pp. 10-23, 2021. https://doi.org/10.1108/AA-02-2019-0028

[6] Y. Wang, S. Zhang, W. He, and X. Bai, "Model-Based Marker-Less 3D Tracking Approach for Augmented Reality," *Journal of Shanghai Jiaotong University*, vol. 52, no. 1, pp. 83-89, 2018. https://doi.org/10.16183/j.cnki.jsjtu.2018.01.013

[7] Y. Wang, S. Zhang, B. Wan, W. He, and X. Bai, "Point cloud and visual feature-based tracking method for an augmented reality-aided mechanical assembly system," *The International Journal of Advanced Manufacturing Technology*, vol. 99, pp. 2341-2352, 2018. https://doi.org/10.1007/s00170-018-2575-8

[8] Q. Gongye, P. Cheng, and J. Dong, "Image-based visual servoing with depth estimation," *Transactions of the Institute of Measurement and Control*, vol. 44, no. 9, pp. 1811-1823, 2022. https://doi.org/10.1177/01423312211064681

[9] L. F. Tu and Q. Peng, "Method of using RealSense camera to estimate the depth map of any monocular camera," *Journal of Electrical and Computer Engineering*, vol. 2021, article no. 9152035, 2021. https://doi.org/10.1155/2021/9152035

[10] Z. Chen, P. Liu, F. Wen, J. Wang, and R. Ying, "Restoration of motion blur in time-of-flight depth image using data alignment," in *Proceedings of 2020 International Conference on 3D Vision (3DV)*, Fukuoka, Japan, 2020, pp. 820-828. https://doi.org/10.1109/3DV50981.2020.00092

[11] C. Wang, "Research on recognition and pose estimation method of textureless and scattered workpiece based on Kinect," M.S. thesis, Guangdong University of Technology, Guangzhou, China, 2018.

[12] X. Min and J. Huang, "Simplified method for fast image restoration," *Journal of Computer Applications*, vol. 37, no. z1, pp. 169-172, 2017.

[13] H. Liu and C. Cao, "Deep image restoration algorithm based on image fusion," *Modern Electronic Technology*, vol. 43, no. 2, pp. 182-186, 2020. https://doi.org/10.16652/j.issn.1004-373x.2020.02.046

[14] P. X. Cao, W. X. Li, and W. P. Ma, "A tracking registration method for augmented reality based on multi-modal template matching and point clouds," *International Journal of Automation and Computing*, vol. 18, no. 2, pp. 288-299, 2021. https://doi.org/10.1007/s11633-020-1265-9

[15] E. Kwon, D. Pathak, H. U. Kim, P. Dahal, S. C. Ha, S. S. Lee, et al., "Structural insights into stressosome assembly," *IUCrJ*, vol. 6, no. 5, pp. 938-947, 2019. https://doi.org/10.1107/S205225251900945X

[16] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *Proceedings of 2011 International Conference on Computer Vision*, Barcelona, Spain, 2011, pp. 858-865. https://doi.org/10.1109/ICCV.2011.6126326

[17] Y. Jin, J. A. Rossiter, and S. M. Veres, "Accurate 6D object pose estimation and refinement in cluttered scenes," in *Proceedings of the 2nd International Conference on Robotics, Computer Vision and Intelligent Systems*, Valletta, Malta, 2021, pp. 31-39. https://doi.org/10.5220/0010654500003061

[18] W. Hua, J. Guo, Y. Wang, and R. Xiong, "3D point-to-keypoint voting network for 6D pose estimation," in *Proceedings of 2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Shenzhen, China, 2020, pp. 536-541. https://doi.org/10.1109/ICARCV50220.2020.9305322

[19] Y. Wang, S. Zhang, and X. Bai, "A 3D tracking and registration method based on point cloud and visual features for augmented reality aided assembly system," *Journal of Northwestern Polytechnical University*, vol. 37, no. 1, pp. 143-151, 2019. https://doi.org/10.1051/jnwpu/20193710143

[20] I. Shugurov, I. Pavlov, S. Zakharov, amnd S. Ilic, "Multi-view object pose refinement with differentiable renderer," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2579-2586, 2021. https://doi.org/10.1109/LRA.2021.3062350

[21] R. Liu, X. Fan, X. Yin, J. Wang, and X. Yang, "Combined estimation method of base and parts pose in AR-assisted assembly," *Machine Design and Research*, vol. 2018, no. 6, pp. 119-125+137, 2018. https://lib.cqvip.com/Qikan/Article/Detail?id=6100106025&from=Qikan_Search_Index

[22] C. Fotsing, N. Menadjou, and C. Bobda, "Iterative closest point for accurate plane detection in unorganized point clouds," *Automation in Construction*, vol. 125, article no. 103610, 2021. https://doi.org/10.1016/j.autcon.2021.103610

[23] J. Zhang, Y. Yao, and B. Deng, "Fast and robust iterative closest point," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3450-3466, 2022. https://doi.org/10.1109/TPAMI.2021.3054619

[24] L. Liang, "Precise iterative closest point algorithm for RGB-D data registration with noise and outliers," *Neurocomputing*, vol. 399, pp. 361-368, 2020. https://doi.org/10.1016/j.neucom.2020.02.076

[25] Z. Zhu, W. Xiang, J. Huo, M. Yang, G. Zhang, and L. Wei, "Non-cooperative target pose estimation based on improved iterative closest point algorithm," *Journal of Systems Engineering and Electronics*, vol. 33, no. 1, pp. 1-10, 2022. https://doi.org/10.23919/JSEE.2022.000001

[26] L. Zheng, X. Liu, Z. An, S. Li, and R. Zhang, "A smart assistance system for cable assembly by combining wearable augmented reality with portable visual inspection," *Virtual Reality & Intelligent Hardware*, vol. 2, no. 1, pp. 12-27, 2020. https://doi.org/10.1016/j.vrih.2019.12.002

Pengxia Cao **https://orcid.org/0000-0002-3020-1650**

She received M.Eng. degree in circuits and systems from Hunan Normal University, China, in 2015, and Ph.D. degree in electronic science and technology from Lanzhou Institute of Physics, CAST. Currently, she teaches at College of Physics and Electronic Engineering, Northwest Normal University. Her current research interests include space electronic technology, computer vision, and augmented reality.

Yibo Huang **https://orcid.org/0000-0003-1667-3114**

He received Ph.D. degree in Control Theory and Control Engineering from Lanzhou University of Technology. Currently, he is an associate professor at College of Physics and Electronic Engineering, Northwest Normal University. His current research interests include multimedia information security, authentication and retrieval.