IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 12, 2020, accepted March 1, 2020, date of publication March 4, 2020, date of current version March 13, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2978255

3D Object Recognition and Pose Estimation From Point Cloud Using Stably Observed Point Pair Feature

DEPING LI[1,2], HANYUN WANG[3], NING LIU[1,2], XIAOMING WANG[1], AND JIN XU[1,2]

[1]College of Information Science and Technology, Jinan University, Guangzhou 510632, China
[2]Robotics Research Institute, Jinan University, Guangzhou 510632, China
[3]Institute of Surveying and Mapping, Information Engineering University, Zhengzhou 450000, China

Corresponding author: Ning Liu (tliuning@jnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61775172, in part by the Natural Science Foundation of Guangdong Province, China, under Grant 2018030310482, and in part by the Jinan University Special Fund for Science and Technology Platform Construction under Grant 17817003.

ABSTRACT Recognition and pose estimation from 3D free-form objects is a key step for autonomous robotic manipulation. Recently, the point pair features (PPF) voting approach has been shown to be effective for simultaneous object recognition and pose estimation. However, the global model descriptor (e.g., PPF and its variants) that contained some unnecessary point pair features decreases the recognition performance and increases computational efficiency. To address this issue, in this paper, we introduce a novel strategy for building a global model descriptor using stably observed point pairs. The stably observed point pairs are calculated from the partial view point clouds which are rendered by the virtual camera from various viewpoints. The global model descriptor is extracted from the stably observed point pairs and then stored in a hash table. Experiments on several datasets show that our proposed method reduces redundant point pair features and achieves better compromise of speed vs accuracy.

INDEX TERMS 3D object recognition, 3D pose estimation, point cloud, point pair feature.

I. INTRODUCTION

3D object recognition is a popular topic in computer vision with numerous applications including robotics, biometrics, navigation, remote sensing, medical diagnosis, entertainment, and education. The aim of 3D object recognition and pose estimation includes correctly recognizing objects presented in a scene and estimating their pose containing 3 degrees of rotation and 3 degrees of translation. With the availability of low-cost 3D data acquisition devices such as Microsoft's Kinect and Intel's RealSense, research on 3D object recognition and pose estimation algorithms has been extensively developed over the last two decades. However, 3D object recognition and pose estimation are still a challenging task in the case of perturbations,such as noise, occlusions and clutter.

Many different methods have been proposed for 3D object recognition and pose estimation [1], such as 3D feature descriptors based methods [2]–[13], template matching based methods [16]–[18] and deep learning based methods [19]–[26]. Usually, 3D feature descriptors based methods can broadly be classified into two categories: global feature-based algorithms and local feature-based algorithms. The global feature-based algorithms have efficiency in the aspects of computation time and memory consumption. However, they are sensitive to occlusion and clutter, and require the objects of interest to be segmented beforehand from the background. Compared with global feature-based algorithms, local feature-based algorithms are more robust to occlusion and clutter [14], [15]. But local feature-based algorithms incur additional computational time in the subsequent stages of matching and hypothesis verification. Template matching based methods is able to detect textureless objects but is sensitive to occlusion. More recently deep learning based methods have been introduced into 3D object recognition and pose estimation [19]–[24] and has good performance in public datasets. However, deep learning based methods require massive computational power and a large amount of time to prepare annotated datasets.

The associate editor coordinating the review of this manuscript and approving it for publication was Jianqing Zhu.

PPF [27] has shown very successful object recognition performance on different 3D datasets and has been applied in many practical scenarios [44]. Because of its superior performance, PPF has been improved and extended by many authors [28]–[31]. However, PPF and its variants share a common problem: all point pair features of the model are extracted and stored in a hash table. If there are too many redundant point pair features in the hash table, the recognition performance degrades. Some point pair features in the input scene match with some point pair features in the model that are unlikely to appear in the input scene, which increases the matching time. To address these limitations, we extract point pair features that are stably observed in some viewpoints and store them in a hash table, and reliable recognition is achieved by using such point pair features. We calculate the observability of 3D points on the object model through a multi-view rendering strategy to form the visible model part and extract point pair features from stably observed points.

Our main contributions are described as follows. 1) We propose a multiview rendering strategy to sample the visible model points; 2) By defining the observability of the model point and the observability of the point pair feature, the stably observable point pair features are extracted from the 3D model; 3) We demonstrate significant performance improvements of our approach compared with original PPF [27] benchmark on several datasets.

The rest of this paper is organized as follows: Section II introduces related work. Section III describes the PPF algorithm. Section IV describes our proposed algorithm. Section V presents the experimental results. Section VI concludes this paper.

II. RELATED WORK

In this section we review point pair features and their modifications.

PPF was first introduced to recognize free-form 3D objects in point clouds by Drost *et al.* [27]. Because of its superiority, many modifications based on PPF have been proposed. Choi and Christensen [29] introduced a color point pair feature (CPPF) to selectively encode the geometric surface shape and the photometric color characteristics based on the traditional 4-dimensional point pair feature. Drost and Ilic [30] proposed a multimodal point pair feature to use both geometric object edges in the intensity image and depth image. Birdal and Ilic [31] first segmented into clusters, and then PPFs are computed for each segment. Tuzel *et al.* [32] proposed a machine learning-based approach to identify and rank important model features for voting-based 3D pose estimation and object recognition tasks. To reduce the influence of clutter and sensor noise, Hinterstoisser *et al.* [33] introduced different point sampling and voting schemes. Wang *et al.* [34] proposed a new clustering strategy combined with DBSCAN and PCA in PPF to further improve the pose hypothesis result for the mismatching region. Vidal *et al.* [35] introduced a novel preprocessing step based on PPF, which considers the discriminative value of surface information. Recently, Lilita *et al.* [36] presented a complete performance evaluation of the original four-dimensional PPF [27] and its variants including a detailed comparison with the most popular local features.

For the visibility issues of PPF, Kim and Medioni [37] enhanced the discriminative ability of PPF by incorporating the visibility context in the range image. They defined three different space types in terms of visibility: visibility-space, surface, and invisible space. The experiments demonstrated that incorporating information from the visibility context improves the PPF matching performance. However, their method is based on RGB-D data, and it is difficult to define the visible and invisible spaces for the point cloud data used in our method. Choi and Christensen [38] combined the 3D surface point and a point on the geometric edge to improve the robustness to occlusion and clutter in the online matching stage. However, in the offline stage, all point pair features of the model must be extracted and stored. This leads to redundant point pair features in the hash table, some of which are not necessarily observable in the input scene. Birdal and Ilic [39] proposed a multiview rendering strategy to remove hidden/invisible geometries and introduced a sparse voxel-based algorithm to address the global distribution of resulting vertices and normals using Poisson disk sampling and normal space sampling. Their method is related to our approach and the main difference is that our goal is to reduce redundant point pair features. However, the goal of their method was to create a bias-free, sparser model. To reduce the useless point pair information, Liu *et al.* [40] proposed a point pair feature-based descriptor named boundary-to-boundary-using-tangent-line boundary (B2B-TL) and a model description method based on multiple edge appearance models. Their method is very effective even for industrial parts whose point clouds lack key details. However, it is not suitable for general objects. Unlike previous approaches, we propose a general solution to reduce redundant point pair features by defining observability.

III. PPF METHOD

The PPF approach is a voting-based algorithm combining a hash table and efficient voting. In this section, we give a brief description of the PPF method. The details of the PPF method refer to [27].

The point pair feature $F(m_1, m_2)$ that describes the relative position and orientation of the oriented points is invariant to rigid motions. It's defined as:

$$\begin{aligned} F(m_1, m_2) &= F(f_1, f_2, f_3, f_4) \\ &= \left(\|d\|_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2)\right) \end{aligned} \quad (1)$$

where m_1 and m_2 are the oriented points, n_1 and n_2 are their normals, $d = m_2 - m_1$ and $\angle(a, b) \in [0, \pi]$ denote the angle between two vectors a and b, as shown in Fig. 1 (h).

The PPF method can be divided into two stages: global modeling and local matching. At global modeling stage, all model point pair features are extracted and quantized. The

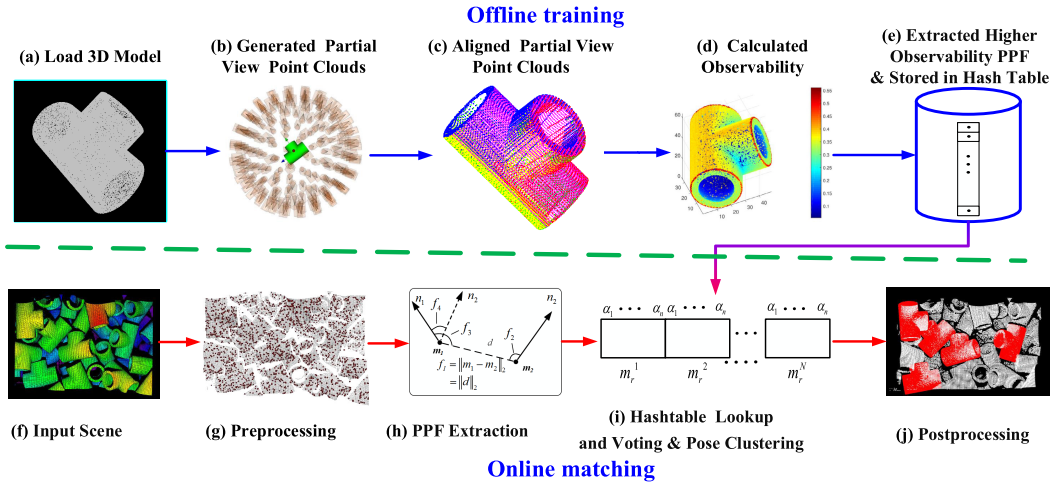

FIGURE 1. The framework of the proposed method. In the offline stage, the loaded CAD models (a) are rendered by the virtual camera and partial view point clouds are generated (b). The partial view point clouds are aligned to the model (c) (different colors represent partial point clouds with different viewpoints). Then, the observability is calculated (d) and the higher observability PPF features are extracted and stored in hash table (e). In the online stage, PPF features are extracted (h) from the input scene (f) after preprocessing (g). Then PPF features are matched to the hash table and generate candidate poses by voting and pose clustering(i), each candidate pose is deal with postprocessing (j).

point pair features are stored in a hash table by using their quantized descriptors as the hash keys. The hash table also stores the angles α^m of every point pair. α^m describes the angle between the positive *y–axis* vector and the *yz–plane* projection of a vector $\vec{a}$. The vector $\vec{a}$ is obtained by transforming the second model points m_2 with a transformation $T_{m\to g}$ which translates m_1 into the origin and rotates its normal n_1 onto the *x–axis*.

At local matching stage, a set of reference points is sampled from a scene. A given scene reference point s_r paired with another scene point s_i to form a point pair feature. Then every scene point pair feature is matched with the model point pair features using the hash table. For each of the potential match (m_r, m_i), the angle $\alpha = \alpha^m - \alpha^s$ is computed and then votes are cast in the 2D space of (m_r, α). α^s is similar to α^m, the different is that the vector $\vec{a}$ is obtained by transforming the second scene point s_2 with a transformation $T_{s\to g}$ which translates the first scene point s_1 into the origin and rotates its normal onto the *x–axis*. After all the matchings are voted, valid pose candidates supported by a certain number of votes are selected. Then the selected poses are clustered and the poses with high scores output as the object poses.

IV. PROPOSED ALGORITHM

In this section, we present the details of the proposed improvement based on the well-known point pair feature voting approach [27] for 3D object recognition and pose estimation on point cloud data. The proposed method pipeline, as shown in Fig. 1, can be divided into an offline training stage and an online matching stage.

A. OFFLINE TRAINING

In the offline training stage, all point pair features of a model are extracted and stored in a hash table to create a global model description according to the original PPF introduced in Section III. The global model description by this approach contains some redundant point pair features that never appear in the input scene. For example, as shown in Fig. 2, the point p_1 and p_2 never appear in the input scene because these two points cannot be observed in the same viewpoint due to self-occlusion. The redundant point pair features not only increase searching time in the voting stage but also increase the matching error.

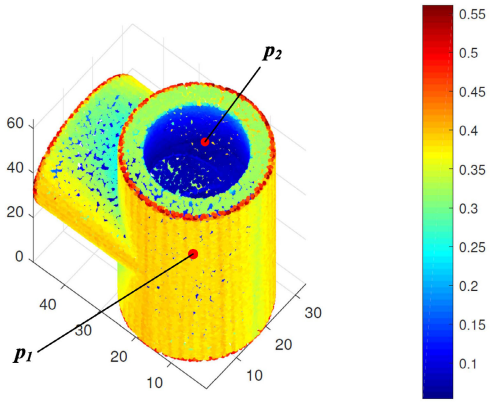

FIGURE 2. The example of two points that cannot appear in the same viewpoint.

We propose an efficient solution to reduce the redundant point pair features based on the observability of points inspired by Akizuki and Hashimoto [43]. The scene point cloud data are always acquired from a certain viewpoint by different techniques, such as stereo vision, structured light, or time-of-flight (TOF). Only partial points of the object can be observed in the scene, and other points are always occluded by itself or other objects. Thus, we extract point pair

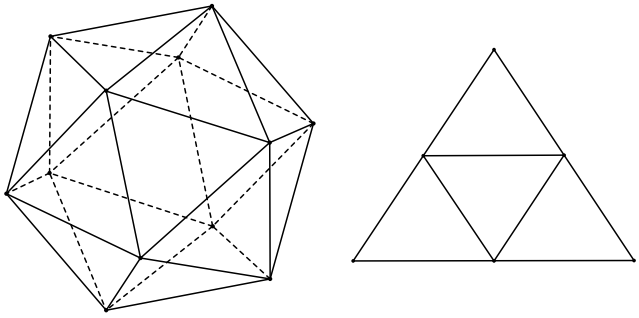

FIGURE 3. The icosahedron and the four equilateral triangles; (a) icosahedron, (b) four equilateral triangles.

features that can be stably observed in the same viewpoint by computing the observability of points in an object model.

For the object model represented by point cloud data, we define "observable points" as points that can be observed from a certain viewpoint. The observability of an object's surface is affected by (1) The point cloud measurement method, (2) The internal constitution of a 3D sensor, (3) The distance from the 3D sensor to the detected object, (4) The viewpoint direction, and (5) The object's shape. The factors of (1) to (3) are not easy to acquire before the recognition process is performed, and we do not consider them in this research. Thus, we calculate the observability by using only factors (4) and (5). To extract observable points in a certain viewpoint, we place several virtual cameras on a bounding sphere around the CAD model of the object, as shown in Fig. 1 (b). The CAD model of the object is then rendered from each viewpoint to obtain a depth buffer. The depth buffer, known as Z-Buffer, is used to determine the visible elements and observable points of the object. To obtain uniform sampling around the object, the sphere is generated using an icosahedron as the starting shape, which subdivides each triangular face with four equilateral triangles, as shown in Fig. 3. This process is implemented recursively for each face until the desired level is reached, which determines how many triangles are used to approximate spheres. The resulting triangles are used to place the camera at their barycenter or vertex, and the pose of the virtual camera and the point cloud of the corresponding viewpoint are obtained by sampling from the depth buffer of the graphic card. There are three main parameters that govern this process: 1) the resolution of the synthetic depth images, 2) the level of recursion and 3) whether to use the vertices or triangle centers of the tessellated sphere as the camera position. In our experiments, a resolution of 450×450 gives an appropriate level of detail over the object. The level of recursion determines the min number of viewpoints. With the level of recursion increasing, the number of viewpoints increases. When using the triangle centers of the tessellated sphere as the camera position, the number of viewpoints is higher than using the vertices of the tessellated sphere as the camera position. For example, when using the vertices of the tessellated sphere as the camera position, and the level of recursion is setting one, 42 partial views point clouds are generated. In the next section, we test the effect of different numbers of partial views.

The observability of the model points is calculated using the pose of the virtual camera and the point cloud of the corresponding viewpoint. We first align these partial view point clouds to the model according to the corresponding pose of the virtual camera. To reduce the computational complexity, the model is subsampled based on voxelgrid method. Then, for any model point p after subsampled, we find the closest point p_c from the partial view point cloud after alignment. If the distance between p and p_c is lower than a setting threshold d_{th}, the model point p is observed in the viewpoint V_k of the partial view point cloud. Otherwise, this model point p is not observed in this viewpoint V_k. In this paper, the distance threshold d_{th} is set to the resolution of the model point after subsampled. The model is uniformly downsampled and the density of the model point is lower than the partial view point clouds. So, the model point which is observed will have at least one point of the partial view point clouds within the distance threshold d_{th}. If the model point isn't observed, the corresponding points within the distance threshold d_{th} in partial view point cloud will not appear. At last, the observability of the model point p is calculated according to (2).

$$Obs(p) = \frac{1}{K} \sum_{k=0}^{K-1} \delta(p, o_k) \tag{2}$$

Here, K represents the number of viewpoints. o_k represents the direction of the viewpoint vector, and the function δ returns "1" when the model point p is observed. Fig. 4 shows the object model and the observability maps for different number of viewpoints.

Fig. 4(a) shows the object model, and Fig. 4(b–f) shows the observability maps under different numbers of viewpoints. The blue and red colors in Fig. 4(b–f) indicate low and high observability, respectively. The outside contour (i.e., the outer circle with highlight) of the object has the highest observability, and the convex surface (i.e., the smooth surface area) of the object has higher observability. Due to self-occlusion, the internal point (i.e, the blue point) of the object has the lowest observability. When the number of viewpoints is small, the distribution of viewpoints is scattered and is not uniform around the surface. Therefore, the observability values of model points are distributed in a broad range, as shown in Fig. 4(b–d). As the number of viewpoints increases, the distribution of viewpoints is relatively uniform around the surface. Therefore, the observability values of model points are relatively uniform and focus in a narrow range, as shown in Fig. 4(e–f).

Using the same approach as calculating the observability of model points, we can calculate the observability of point pair feature $Obs(F(m_r, m_i))$. Because m_r and m_i are observed simultaneously, $Obs(F(m_r, m_i))$ is calculated by (3).

$$Obs(F(m_r, m_i)) = \frac{1}{K} \sum_{k=0}^{K-1} \delta(m_r, o_k)\delta(m_i, o_k) \tag{3}$$

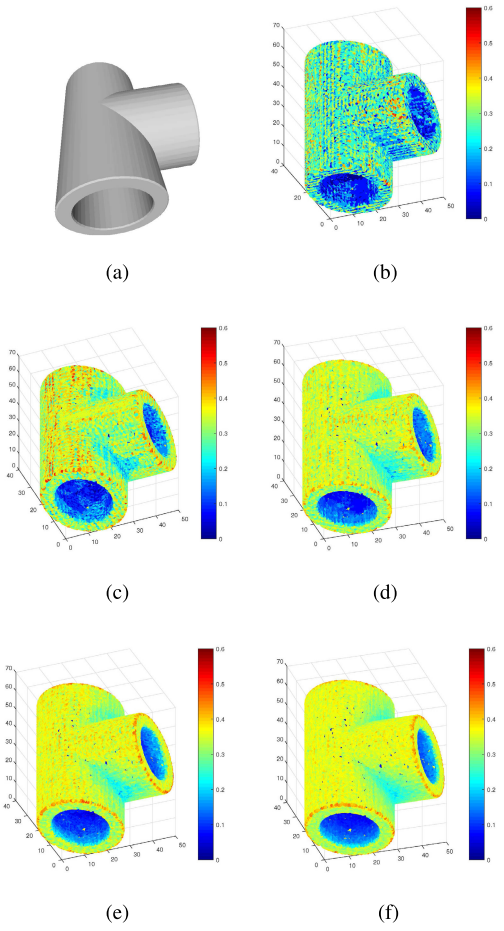

FIGURE 4. Object model and observability map for different viewpoints. (a) the object model; (b) observability map for 12 viewpoints; (c) observability map for 20 viewpoints; (d) observability map for 42 viewpoints;(e) observability map for 80 viewpoints;(f) observability map for 162 viewpoints.

Based on the observability of the point pair feature $Obs(F(m_r, m_i))$, we can set an observability threshold τ_{obs}. When $Obs(F(m_r, m_i)) > \tau_{obs}$, the point pair feature $F(m_r, m_i)$ is computed according to (1). When $Obs(F(m_r, m_i)) \leq \tau_{obs}$, the point pair feature $F(m_r, m_i)$ is set to NaN. Then, the point pair features with observability larger than the given threshold are grouped together based on their similarity and then stored in a hash table.

B. ONLINE MATCHING

The online matching stage includes preprocessing, PPF extraction, pose voting, pose clustering and postprocessing.

1) PREPROCESSING

The number of point pair features is a quadratic of the number of points, which increases the computational complexity. Therefore, a downsampling method is applied in the input scene to reduce the number of points. The input point cloud is divided into multiple voxels with the desired resolution. For each voxel, the point nearest to the centroid of the voxel is selected to represent all points inside the voxel.

As described in Section III, the PPF voting method relies heavily on the normal. Thus, in this manuscript, we use the method proposed in [41] to compute the point's normal, based on the analysis of the eigenvectors and eigenvalues of a covariance matrix created from the nearest neighbors of the query point. To improve the robustness of normal estimation, we find the nearest neighbors of the query point from the point cloud before downsampling.

2) PPF EXTRACTION

During the online stage, we choose one point for every n points as the reference point in the input scene. The fraction of the points selected as the reference is controlled by the parameter P_{ref} (typically, 1/5th or 1/10th in the subsampled scene is used). For each selected reference point, the point pair features with respect to all other points are computed in the subsampled scene. If the distance between the two points is larger than the model diameter $diam(M)$, these two points belong to different objects and therefore should be excluded from the point pair features. The selected reference point should only be paired with other scene points within the model diameter $diam(M)$ from the reference point.

3) POSE VOTING

As described in section III, an object is detected via a voting scheme that is similar to the generalized Hough transform. For each scene point pair (s_r, s_i), k corresponding model point pairs $\{(m_r^1, m_i^1), (m_r^2, m_i^2), \cdots, (m_r^k, m_i^k)\}$ can be retrieved from the hash table, and the corresponding rotation angles $\{\alpha^1, \alpha^2, \cdots, \alpha^k\}$ are calculated based on the (s_r, s_i) and $\{(m_r^1, m_i^1), (m_r^2, m_i^2), \cdots, (m_r^k, m_i^k)\}$. Then, each local coordinate $\{(m_r^1, \alpha^1), (m_r^2, \alpha^2), \cdots, (m_r^k, \alpha^k)\}$ casts a vote in a two-dimensional accumulator array. Because the number of model points N and the discretized number of rotation angles n are known before voting, we use a one-dimensional accumulator array to speed up voting, as shown in Fig. 1 (i). After all point pair features for a scene reference point cast votes, the peak point in the accumulator array is extracted and treated as the optimal local coordinate, from which a global rigid transformation is computed.

4) POSE CLUSTERING

As described above, a candidate pose is estimated for each reference point, and multiple candidate poses may exist for all scene reference points. To merge similar candidate poses, we first sort the candidate poses by the number of votes they received. This ensures that the most likely candidate poses are merged first. Then, all votes are checked in order, and two votes are group together if the rotation and translation between these two votes are smaller than the predefined thresholds th_{tra} and th_{rot}. Finally, for each cluster, a new candidate pose is calculated by averaging the poses contained in the cluster, and individual scores are summed as the score of the new candidate pose.

5) POSTPROCESSING

Finally, a list of candidate poses is sorted according to the voting scores. To reject the false estimation of pose, the hypothesis verification method proposed by Papazov *et al.* [42] is used. The hypothesis verification method evaluates how well a model hypothesis fits into the scene by an acceptance function μ. The acceptance function μ measures the quality of a hypothesis and consists of a support term and a penalty term. The support term is the number of transformed model points falling within a certain distance from a scene point. The penalty term is the number of model points that would occlude other scene points. A hypothesis is accepted as a valid hypothesis when the support term is higher than a predefined and the penalty term is lower than a predefined. A conflict graph is built when two valid hypotheses are conflicting if the intersection of the point sets they explain is nonempty. The better hypotheses are then selected by performing a nonmaximum suppression on the conflict graph.

V. EXPERIMENTAL RESULTS

In this section, we first evaluate the effect of parameters on the UWA dataset [9]. Then we compare the proposed method with the original PPF method on the UWA dataset, synthetic dataset and real dataset. For all experiments $1/5th$ of the points in the downsampled scene are used as reference points. The normal and the distance of the model are stored in a hash table with 30 bins and 20 bins, respectively. The $diam(M)$ is the model diameter. The pose clustering thresholds th_{tra} and th_{rot} are set to be $th_{tra} = diam(M)/10$ and $th_{rot} = 12°$, respectively. The algorithm presented in this paper is implemented with the point cloud library (PCL) [45], and the tests are performed on a PC with a 3.2 GHz Intel®Core(TM) i7-8700 CPU and 16 GB DDR III RAM. All methods are used OpenMP to speed up matching process.

A. THE DATASET

1) THE UWA DATASET

The UWA dataset contains five object models with full viewpoints and fifty scenes at specific viewpoints. The scene data are scanned with a Minolta VIVID 910 scanner with a resolution of 640 × 480 and saved as a 3D point cloud in the ply file format. Each scene is generated by randomly placing four or five objects together. For each object in a scene, the ground truth pose is a 4 × 4 transformation matrix.

The rhino model is usually excluded from evaluation due to large holes during the scanning procedure. The total number of scene instances is 188 (50 for the object chef, 48 for the object chicken, 45 for the object Parasaurolophus and 45 for the object T-rex) in the UWA dataset. The detected objects are depicted in Fig. 5.

2) SYNTHETIC DATASET

The synthetic dataset contains 3 types of target parts, as shown in Fig. 6. Each target part has 30 scenes. The diameter of the PartA, PartB and PartC is 87 mm, 73 mm and 36 mm, respectively. For each scene, 20 same target parts are simulated that drop down one by one from a defined position above a bin with a random orientation. The scene data are scanned with a virtual camera and saved as the 3D point cloud with a PCD file. For each part in a scene, the ground truth pose is a 4×4 transformation matrix.

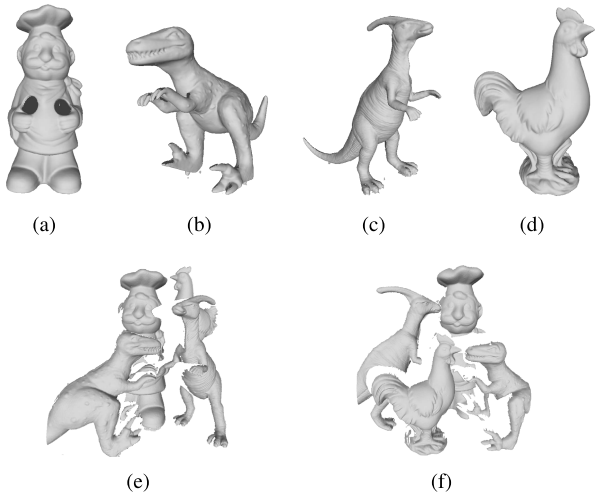

FIGURE 5. The object model and two random scenes in the UWA dataset, (a) Chef; (b) T-Rex; (c) Parasaurolophus; (d) Chicken; (e-f) two random scenes.

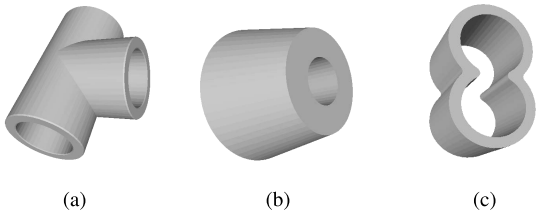

FIGURE 6. The objects model of the synthetic dataset. (a) PartA. (b) PartB. (c) PartC.

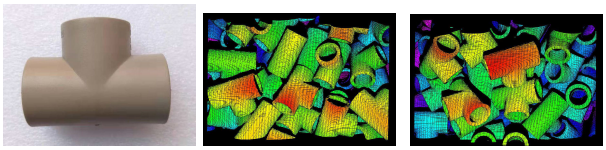

FIGURE 7. The target part and two random scenes in the real scene dataset; (a) the target part; (b-c) two random scenes.

3) REAL SCENE DATASET

The real scene dataset is captured by a 3D acquisition system that consists of a digital micromirror device projector (TI DLP lightcrafter 4500) and a monochrome camera (Point-Grey Flea3 FL3-U3-13Y3) equipped with a c-mount lens (Computar 8mm F1.4 M0814-MP2). The real data contain one target part (PartA) and twenty scenes at specific viewpoints. Each scene includes more than five instances. Fig. 7 shows the target part and two random scenes.

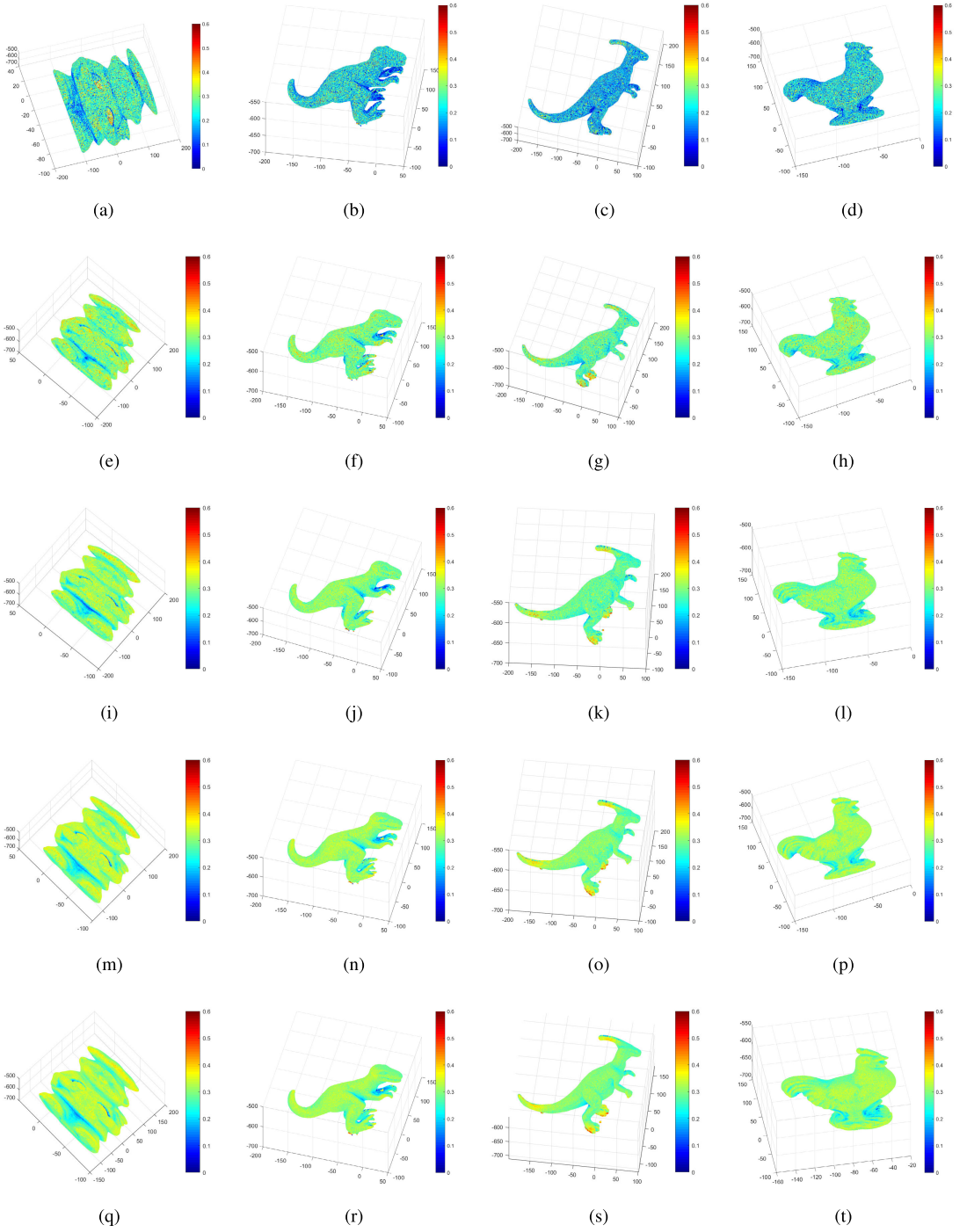

FIGURE 8. The observability map of the model in UWA dataset under different numbers of viewpoints. The four columns from left to right represent observability maps of chef, T-rex, parasaurolophus and chicken, respectively. The five rows from top to bottom represent observability maps of 12,20,42,80 and 162 viewpoints, respectively.

B. EVALUATION CRITERIA

In this paper, we adopt the average point distance (ADD) [46] as the pose error metric. Given the ground truth pose $\bar{\mathbf{P}} = (\bar{\mathbf{R}}, \bar{\mathbf{t}})$ and the estimated pose $\hat{\mathbf{P}} = (\hat{\mathbf{R}}, \hat{\mathbf{t}})$, ADD is calculated as follows:

$$\text{ADD} = \underset{x \in \mathcal{M}}{\text{avg}} \|(\bar{\mathbf{R}}x + \bar{\mathbf{t}}) - (\hat{\mathbf{R}}x + \hat{\mathbf{t}})\|_2 \tag{4}$$

Here, $\mathcal{M}$ is the set of 3D model points, x is the model point. Based on [19], we use the area under the accuracy-threshold curve (AUC) for pose evaluation. The maximum threshold is set to 10 cm in the UWA dataset.

If the object model is indistinguishable under different views, the object pose is ambiguous due to object symmetries or occlusions [47]. Based on [46], the error is measured

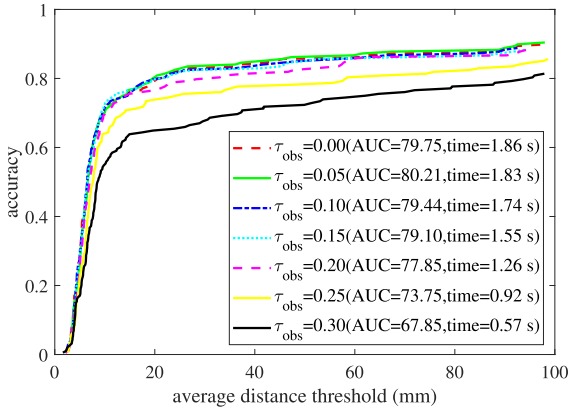

FIGURE 9. The accuracy-threshold curve under different observability threshold τ_{obs} when the viewpoints $K = 162$.

according to the average closest point distance:

$$\text{ADD-S} = \underset{x_1 \in \mathcal{M}}{\text{avg}} \min_{x_2 \in \mathcal{M}} \|(\bar{\mathbf{R}}x_1 + \bar{\mathbf{t}}) - (\hat{\mathbf{R}}x_2 + \hat{\mathbf{t}})\|_2 \quad (5)$$

Here, $\mathcal{M}$ is the set of 3D model points, x_1, x_2 is the closest model point in the camera coordinate system. For the synthetic dataset, we use ADD-S as the error of the estimated pose.

The fitness of the aligned object surface is the main indicator of pose estimation quality. If the ground truth pose is not available, we define the average distance *ADE* between the visible model and the nearest point of the scene as the error of the estimated pose. For the real scene dataset, we use *ADE* as the error of the estimation pose.

C. PARAMETER ANALYSIS

In this subsection, the number of viewpoints K and the observability threshold τ_{obs} are analyzed. We first qualitatively analyze the effect of the number of viewpoints according to the observability map of the object model in the UWA dataset. From Fig. 8, we find that when the number of viewpoints is small, the observability distribution of model points is not uniform. Some model points have very high observability values, but the observability values near them are very low and the overall observability values are relatively small, such as shown in Fig. 8(a-h). It can't reflect the observability distribution of model points well. When the number of viewpoints is large, the observability distribution of model points is relatively uniform, such as shown in Fig. 8(m-t). So, the number of viewpoints K is set to 162 in our paper.

Then, we quantitatively analyze the observability threshold τ_{obs} according to the recognition performance on the UWA dataset. In this experiment, d_{voxel} is set to $0.025 \times diam(M)$. Because the scenes contain models with different diameters, and Drost *et al.* [27] only reported the average number of points in the subsampled scene to be $S \approx 1690$. The scenes are downsampled with 8 mm and the average number of points in the subsampled scene is $S \approx 1750$ in this paper. We vary the observability threshold τ_{obs} from 0.0 to 0.30 with step 0.05. Fig. 9 shows the results of the detailed evaluation. We can see that when the observability threshold τ_{obs} increases, the matching time decreases, and the AUC first increases and then decreases. This is because the hash table which is low the observability threshold contains some redundant point pair features which affect the performance. And with the observability threshold increasing, some importance point pair features are removed. When $\tau_{obs} \leq 0.15$, AUC has relatively good performance. The observability threshold τ_{obs} is related to the shape of the model, and the optimal value is between 0 and 0.15. In the following experiment, the viewpoints and the observability threshold are set to be $K = 162$ and $\tau_{obs} = 0.05$, respectively.

TABLE 1. Area under accuracy-threshold curve for pose evaluation using ADD metric from [19] on the UAW dataset.

Object	Original PPF [27]		Our method	
	AUC	time(s)	AUC	time(s)
Cheff	**91.20**	3.72	91.07	3.64
Chicken	89.28	1.58	**91.24**	1.58
T-Rex	**83.70**	1.44	83.66	1.43
Parasaurolophus	58.95	0.51	**58.99**	0.51
Mean	80.78	1.81	**81.24**	1.79

D. COMPARISON WITH THE ORIGINAL PPF

In this section, we compare our proposed method with the original PPF method [26] in the UWA dataset, the synthetic dataset and the real dataset.

1) COMPARISON WITH ORIGINAL PPF IN THE UWA DATASET

Table 1 shows the detailed evaluation results for all the 4 objects in the UWA Dataset. We show the area under the accuracy-threshold curve (AUC) using the ADD metric [47] and the maximum threshold is set to 10 cm. In this experiment, the models are downsampled with with $d_{voxel} = 0.025 \times diam(M)$. For the object of chicken and parasaurolophus, our proposed method outperforms the original PPF method [27] with the same matching time. From Table 1, we can observed that our proposed method outperforms the original PPF method in the UWA dataset with less matching time. In table 2, the PPF number of the object model and the average reductions number of matching PPF in each scene are also analyzed. It is clearly seen that our proposed method can greatly reduce the PPF number of the object model. The Table 1 and Table 2 demonstrate that our proposed method achieves better performance by removing the point pair features with lower observability.

2) COMPARISON WITH ORIGINAL PPF IN SYNTHETIC DATASET

In this subsection, we compare our proposed method with the original PPF method [27] on synthetic dataset. The estimated pose is compared with the ground truth pose. Because the target parts are symmetrical and their rotation direction is not unique, we use ADD-S to evaluate the accuracy of the estimated pose. The model and the scene are downsampled

TABLE 2. The PPF numbers of object model on the UWA dataset; MN1 is the PPF numbers of object model in the original PPF method; MN2 is the PPF numbers of object model in the our proposed method; RN is the average number of the matching PPF reductions in each scene by our proposed method.

Object	MN1	MN2	RN
Cheff	4361832	**3087036**	11045
Chicken	2648756	**1946916**	6123
T-Rex	2308880	**1745438**	2855
Parasaurolophus	762998	**643700**	360
Mean	2502616	**1855772**	5095

TABLE 3. The evaluation results on the synthetic datasets.

Object	Original PPF			Our method		
	AUC	RR(%)	Time(s)	AUC	RR(%)	Time(s)
PartA	4.18	94.29	1.17	4.18	**94.74**	0.94
PartB	4.25	81.82	1.52	**4.56**	**86.11**	1.27
PartC	2.81	100	0.62	2.81	100	0.57
Mean	3.74	92.03	1.10	**3.85**	**93.62**	0.92

with $0.05 \times diam(M)$ for improving the overall matching speed. We report the ADD–S AUC ($< diam(M)/10$) and the recognition rate (RR). The recognition rate equals the number of true positive divided by the number of output poses. If the error is within the range of $10\% \times diam(M)$, we count the case as a true positive; otherwise, it is regarded as a false positive. Tables 3 shows the evaluation results on the synthetic dataset. For our proposed method, the mean of AUC, recognition rate and the matching time are 3.85, 93.62% and 0.92s, respectively. For original PPF method, the mean of AUC, recognition rate and the matching time are 3.74, 92.03% and 1.10s, respectively. Thus our proposed method achieves better performance with less computational time in synthetic dataset.

3) COMPARISON WITH ORIGINAL PPF IN REAL SCENE DATASET

To show the performance of our algorithm concerning real scene data, we capture the 3D data of real scenes using a self-built structured light senor. In this experiment, we deliberately did not perform any postprocessing of acquired point clouds, such as outlier removal and smoothing. The model and the scene are downsampled with $0.05 \times diam(M)$. Because there is no ground truth pose for the real scene part, we quantify the pose estimation error with the average distance between the visible model and the nearest point of the scene. We first transform the model into a real scene by the estimated pose. Then, a visible model can be generated via Z-buffering. The distance between the visible model and the nearest point of the scene can be obtained by an efficient Kd-tree structure. Table 4 shows the AUC($< diam(M)/10$), the recognition rate and the matching time in real scene data. The recognition rate of our proposed method is 69.60% which is higher than the original PPF method 66.99%. The matching time of our proposed method is 0.66s which is lower than the original PPF method 0.82s. Thus our proposed method outperforms the original PPF method [27]. Fig. 10 shows the test scenes and the qualitative results.

TABLE 4. The results of the algorithms for real scene data.

Methods	AUC	RR(%)	Time(s)
Original PPF [27]	3.41	66.99%	0.82
Our proposed method	**3.55**	**69.60%**	**0.66**

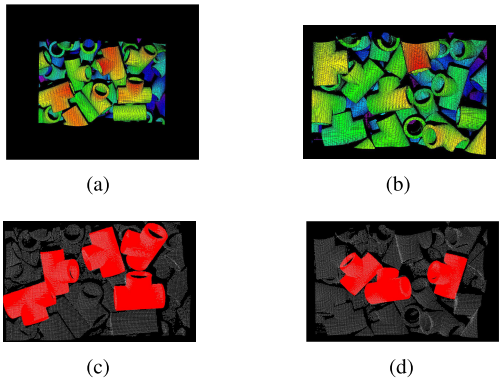

FIGURE 10. The recognition result on the real scene dataset, the first row shows the real scenes, the second row shows the recognition results with our proposed method.

VI. CONCLUSION

In this paper, we propose a novel strategy to build a global model description based on point pair features and Hough-like voting. We analyze the observability of the model points according to the multiview rendering strategy and extract point pair features from stably observed points.

The experimental results on UWA, synthetic and real scene datasets demonstrate that the proposed method achieves a better trade-off between speed and recognition performance.

In future work, we will test its feasibility for the application of bin-picking robots.

REFERENCES

[1] M. Fujita, Y. Domae, A. Noda, G. A. G. Ricardez, T. Nagatani, A. Zeng, S. Song, A. Rodriguez, A. Causo, I. M. Chen, and T. Ogasawara, "What are the important technologies for bin picking? Technology analysis of robots in competitions based on a set of performance metrics," *Adv. Robot.*, pp. 1–15, Dec. 2019, doi: 10.1080/01691864.2019.1698463.

[2] S. Boubou, H. J. Asl, T. Narikiyo, and M. Kawanishi, "Real-time recognition and pursuit in robots based on 3D depth data," *J. Intell. Robotic Syst.*, vol. 93, nos. 3–4, pp. 587–600, Jan. 2018.

[3] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the viewpoint feature histogram," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, Oct. 2010, pp. 2155–2162.

[4] A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R. B. Rusu, and G. Bradski, "CAD-model recognition and 6DOF pose estimation using 3D cues," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCV Workshops)*, Barcelona, Spain, Nov. 2011, pp. 585–592.

[5] A. Aldoma, "OUR-CVFH—Oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6DOF pose estimation," in *Proc. Joint DAGM (German Assoc. Pattern Recognit.) OAGM Symp.*, 2012, pp. 113–122.

[6] D. Li, N. Liu, Y. Guo, X. Wang, and J. Xu, "3D object recognition and pose estimation for random bin-picking using partition viewpoint feature histograms," *Pattern Recognit. Lett.*, vol. 128, pp. 148–154, Dec. 2019.

[7] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3D object classification," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Karon Beach, Phuket, Dec. 2011, pp. 2987–2992.

[8] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 433–449, May 1999.

[9] A. S. Mian, M. Bennamoun, and R. Owens, "Three-dimensional model-based object recognition and segmentation in cluttered scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1584–1601, Oct. 2006.
[10] F. Tombari, S. Samuele, and L. D. Stefano, "Unique signatures of histograms for local surface description," in *Proc. ECCV*, Sep. 2010, pp. 356–369.
[11] Y. Lei, M. Bennamoun, and A. A. El-Sallam, "An efficient 3D face recognition approach based on the fusion of novel local low-level features," *Pattern Recognit.*, vol. 46, no. 1, pp. 24–37, Jan. 2013.
[12] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3D local surface description and object recognition," *Int. J. Comput. Vis.*, vol. 105, no. 1, pp. 63–86, Apr. 2013.
[13] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. IEEE Int. Conf. Robot. Autom.*, Kobe, Japan, May 2009, pp. 3212–3217.
[14] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.
[15] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok, "A comprehensive performance evaluation of 3D local feature descriptors," *Int. J. Comput. Vis.*, vol. 116, no. 1, pp. 66–89, Apr. 2015.
[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 *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 858–865.
[17] S. Hinterstoisseret, "Dominant orientation templates for real-time detection of texture-less objects," in *Proc. CVPR*, 2010, pp. 2257–2264.
[18] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of texture-less objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 876–888, May 2012.
[19] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arxiv:1711.00199*. [Online]. Available: https://arxiv.org/abs/1711.00199
[20] W. Kehl, "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again," in *Proc. ICCV*, 2017, pp. 1521–1529.
[21] G. Billings and M. Johnson-Roberson, "SilhoNet: An RGB method for 6D object pose estimation," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3727–3734, Oct. 2019.
[22] C. Wang, D. Xu, Y. Zhu, R. Martin-Martin, C. Lu, L. Fei-Fei, and S. Savarese, "DenseFusion: 6D object pose estimation by iterative dense fusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3343–3352.
[23] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4561–4570.
[24] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "PVN3D: A deep point-wise 3D keypoints voting network for 6DoF pose estimation," 2019, *arXiv:1911.04231*. [Online]. Available: http://arxiv.org/abs/1911.04231
[25] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3D point cloud matching with smoothed densities," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4561–4570.
[26] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global context aware local features for robust 3D point matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018.
[27] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Francisco, CA, USA, Jun. 2010, pp. 998–1005.
[28] C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam, "Voting-based pose estimation for robotic assembly using a 3D sensor," in *Proc. IEEE Int. Conf. Robot. Autom.*, Saint Paul, MN, USA, May 2012, pp. 1724–1731.
[29] C. Choi and H. I. Christensen, "RGB-D object pose estimation in unstructured environments," *Robot. Auto. Syst.*, vol. 75, pp. 595–613, Jan. 2016.
[30] B. Drost and S. Ilic, "3D object detection and localization using multimodal point pair features," in *Proc. 2nd Int. Conf. 3D Imag., Modeling, Process., Vis. Transmiss.*, Zurich, Switzerland, Oct. 2012, pp. 9–16.
[31] T. Birdal and S. Ilic, "Point pair features based object detection and pose estimation revisited," in *Proc. Int. Conf. 3D Vis.*, Lyon, France, Oct. 2015, pp. 527–535.
[32] O. Tuzel, "Learning to rank 3D features," in *Proc. ECCV*, 2014, pp. 520–535.
[33] S. Hinterstoisser, "Going further with point pair features," in *Proc. ECCV*, Sep. 2016, pp. 834–848.
[34] Z. Wang, "Pose estimation with mismatching region detection in robot bin picking," in *Proc. ICIRA*, Aug. 2017, pp. 36–47.
[35] J. Vidal, C.-Y. Lin, X. Lladó, and R. Martí, "A method for 6D pose estimation of free-form rigid objects using point pair features on range data," *Sensors*, vol. 18, no. 8, p. 2678, Aug. 2018.
[36] L. Kiforenko, B. Drost, F. Tombari, N. Krüger, and A. G. Buch, "A performance evaluation of point pair features," *Comput. Vis. Image Understand.*, vol. 166, pp. 66–80, Jan. 2018.
[37] E. Kim and G. Medioni, "3D object recognition in range images using visibility context," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Francisco, CA, USA, Sep. 2011, pp. 3800–3807.
[38] C. Choi and H. I. Christensen, "3D pose estimation of daily objects using an RGB-D camera," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vilamoura, Portugal, Oct. 2012, pp. 3342–3349.
[39] T. Birdal and S. Ilic, "A point sampling algorithm for 3D matching of irregular geometries," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Vancouver, BC, Canada, Sep. 2017, pp. 6871–6878.
[40] D. Liu, S. Arai, J. Miao, J. Kinugawa, Z. Wang, and K. Kosuge, "Point pair feature-based pose estimation with multiple edge appearance models (PPF-MEAM) for robotic bin picking," *Sensors*, vol. 18, no. 8, p. 2719, Aug. 2018.
[41] R. R. Bogdan, "Semantic 3D object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, Aug. 2010.
[42] C. Papazov and D. Burschka, "An efficient RANSAC for 3D object recognition in noisy and occluded scenes," in *Proc. ACCV*, Nov. 2010, pp. 135–148.
[43] S. Akizuki and M. Hashimoto, "Position and pose recognition of randomly stacked objects using highly observable 3D vector pairs," in *Proc. 40th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Dallas, TX, USA, Oct. 2014, pp. 5266–5271.
[44] C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka, "Rigid 3D geometry matching for grasping of known objects in cluttered scenes," *Int. J. Robot. Res.*, vol. 31, no. 4, pp. 538–553, Mar. 2012.
[45] *Point Cloud Library (PCL)*. Accessed: May 15, 2018. [Online]. Available: http://pointclouds.org/
[46] T. Hodaň, J. Matas, and V. S. Obdržálek, "On evaluation of 6D object pose estimation," in *Proc. ECCV*, Nov. 2016, pp. 606–619.
[47] S. Hinterstoisser, "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes," in *Proc. ACCV*, 2012, pp. 548–562.

DEPING LI received the B.S. degree from the College of Physics and Photoelectric Engineering, Guangdong University of Technology, Guangzhou, China, in 2011, and the Ph.D. degree from the Hefei Institute of Physical Science, Chinese Academy of Sciences, Hefei, China, in 2016.

In 2016, he joined the College of Information Science and Technology, Jinan University, Guangzhou, where he is currently a Postdoctoral Researcher. His research interests include 3D object recognition, shape matching, robotic vision, and semantic segmentation.

HANYUN WANG received the Ph.D. degree in electronic science and engineering from the National University of Defense Technology, in 2015. He was a Visiting Ph.D. Student with Xiamen University, from 2011 to 2014. His research interests include computer vision, machine learning, pattern recognition, mobile laser scanning, and point cloud processing.

NING LIU received the B.S. degree in machinery manufacturing and automation from the Huazhong University of Science and Technology, in 1985, and the master's degree in mechanics and the Ph.D. degree from the South China University of Technology, in 2000 and 2007, respectively. He is currently a Professor with the Department of Electronic Engineering, Jinan University, China. His research interests include industrial robots, machine vision, and numerical control technology.

XIAOMING WANG received the B.Sc. degree from the Harbin Institute of Technology, China, and the Ph.D. degree from Nankai University, China. She is currently a Professor with the Department of Computer Science, Jinan University, China. Her research interests include security and privacy in networks and distributed systems, such as wireless sensor networks and cloud computing, with a focus on security protocol designs and access control.

JIN XU received the B.S. degree in electronic information engineering from Jinan University, Guangzhou, China, in 2012, where he is currently pursuing the degree in signal and information processing with the College of Information Science and Technology. His main research interests include computer 3D vision, point cloud processing, target pose estimation, and robotic bin-picking applications.