Efficient MSPSO Sampling for Object Detection and 6D Pose Estimation in 3D Scenes

Xing, Xuejun; Guo, Jianwei; Nan, Liangliang; Gu, Qingyi; Zhang, Xiaopeng; Yan, Dong Ming

DOI
10.1109/TIE.2021.3121721

Publication date
2022

Document Version
Final published version

Published in
IEEE Transactions on Industrial Electronics

Efficient MSPSO Sampling for Object Detection and 6-D Pose Estimation in 3-D Scenes

Xuejun Xing, Jianwei Guo, Liangliang Nan, Qingyi Gu, *Member, IEEE*, Xiaopeng Zhang, *Member, IEEE*, and Dong-Ming Yan

***Abstract*—The *point pair feature* (PPF) is widely used in manufacturing for estimating 6-D poses. The key to the success of PPF matching is to establish correct 3-D correspondences between the object and the scene, i.e., finding as many valid similar point pairs as possible. However, efficient sampling of point pairs has been overlooked in existing frameworks. In this article, we propose a revised PPF matching pipeline to improve the efficiency of 6-D pose estimation. Our basic idea is that the valid scene reference points are lying on the object's surface and the previously sampled reference points can provide prior information for locating new reference points. The novelty of our approach is a new sampling algorithm for selecting scene reference points based on the multisubpopulation particle swarm optimization guided by a probability map. We also introduce an effective pose clustering and hypotheses verification method to obtain the optimal pose. Moreover, we optimize the progressive sampling for multiframe point clouds to improve processing efficiency. The experimental results show that our method outperforms previous methods by 6.6%, 3.9% in terms of accuracy on the public DTU and LineMOD datasets, respectively. We further validate our approach by applying it in a real robot grasping task.**

***Index Terms*—3-D point cloud, 6-D pose estimation, multisubpopulation particle swarm optimization (MSPSO), point pair features (PPF).**

I. Introduction

OBJECT detection and 6-D pose estimation is a key component of various applications in intelligent manufacturing and industrial robotics. The purpose of 6-D pose estimation is to determine an object pose that is represented by a 3×3 rotation matrix R and a 3-D translation vector $\mathbf{t}$. In particular, the object pose is the transformation that converts a 3-D point $\mathbf{p}_o$ in the local object coordinate system into the corresponding 3-D point $\mathbf{p}_s$ in the scene coordinate system, i.e., $p_s = Rp_o + \mathbf{t}$.

Researchers in the machine vision and robotics communities have presented many methods on 6-D pose estimation. In general, the key technologies are based on three directions.

A. Template Matching

The template matching method converts the object into a series of templates and searches the position of each template on the entire scene to obtain the object pose [1]–[7]. These methods have the advantages of strong real-time performance and easy implementation [5], [7]. However, their performance of instances detection and pose estimation in complex scenes is compromised due to changes in light and scene occlusion [4], [8]–[10].

B. Deep-Learning-Based Methods

In recent years, deep learning has been developed rapidly in the field of image processing and analysis [11]–[15]. Similarly, many pose estimation methods based on deep learning are proposed for point cloud [16]–[19], RGB [7], [10], [20]–[24], or RGB-D images [25]–[32]. These methods show great potential in 6-D pose estimation. However, so far there are few methods focusing on unstructured point clouds. Hagelskjaer *et al.* [16] performed semantic segmentation on the input point cloud based on PointNet, then used traditional methods to estimate the 6-D pose of the segmented instances. The methods of [17]–[19] require a segmented point cloud as input, which depends on the segmentation using RGB images. Furthermore, the accuracy of these methods is still lower than that of deep learning models based on RGB-D images and is not yet saturated [18], [19]. On the other hand, the combination of deep learning models and

Manuscript received June 2, 2021; revised August 12, 2021 and September 16, 2021; accepted October 10, 2021. Date of publication October 27, 2021; date of current version May 2, 2022. This work is supported in part by the National Key R&D Program under Grant 2018YFB2100602, in part by the National Natural Science Foundation of China under Grant 62172416, Grant 62172415, Grant 61802406, and Grant 61972459, in part by the Open Research Fund Program of State Key Laboratory of Hydroscience and Engineering, Tsinghua University under Grant sklhse-2020-D-07, in part by the Scientific Instrument Developing Project of the Chinese Academy of Sciences under Grant YJKYYQ20200045, and in part by the Open Research Projects of Zhejiang Lab under Grant 2021KE0AB07 and Project TC210H00L/42. *(Corresponding author: Jianwei Guo.)*

Xuejun Xing is with School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China, and NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: xingxuejun2019@ia.ac.cn).

Jianwei Guo, Qingyi Gu, and Dong-Ming Yan are with Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: jianwei.guo@nlpr.ia.ac.cn; qingyi.gu@ia.ac.cn; dongming.yan@nlpr.ia.ac.cn).

Liangliang Nan is with the Delft University of Technology, 2628BL Delft, Netherlands (e-mail: liangliang.nan@tudelft.nl).

Xiaopeng Zhang is with NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and Zhejiang Lab (e-mail: xiaopeng.zhang@ia.ac.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIE.2021.3121721.

Digital Object Identifier 10.1109/TIE.2021.3121721

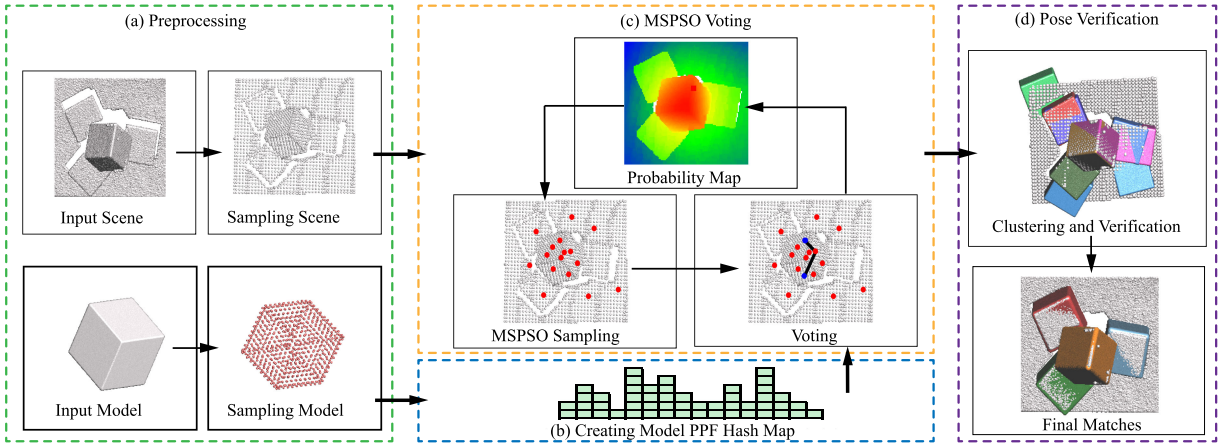


Fig. 1. Illustration of the proposed pipeline. (a) In preprocessing, the input object and scene point cloud are downsampled. (b) Model PPF hash map is generated using the point pairs of the object. (c) We progressively sample scene reference points by the MSPSO algorithm based on a probability cloud map, then perform PPF matching and Hough voting to generate pose hypotheses. (d) Improved pose clustering and verification is conducted to return final poses by filtering out duplicated and invalid poses.

traditional methods for 6-D pose estimation has also achieved competitive results [7], [10], [16], [31], [32].

C. Feature Descriptor Matching

Another kind of methods mainly used local feature descriptors [33]–[35] to establish similar point pairs between object and scene, which are used to generate the approximate 6-D poses. These methods has poor prediction performance when the object has indistinguishable local shape features, such as planar and regular surfaces.

Drost *et al.* [36] proposed a well-known 6-D pose estimation method, which uses point pair feature (PPF) to achieve global model description and local matching. The method firstly describes the 3-D object as a PPF Hash table, then samples the reference points in the scene and votes on them to generate hypothetical poses by finding the similar features in the Hash table, and finally generates the optimal pose by a clustering. Since the method was proposed, many variants [9], [37]–[41] have extended it to obtain better performance. In the PPF-based approach, the voting process is the key to the success of finding optimal poses, but this process is computationally expensive with a complexity of $O(n^2)$[9], [34] where n is the number of scene points. Previous PPF-based methods usually generate a large number of reference points by uniform or random sampling. They do not consider the voting information of the reference points, thus, reducing the efficiency of the entire voting process. Besides, invalid scene reference points have a negative impact on the robustness to noise, occlusion, and cluttered scenes, as well as increasing calculation time for pose clustering and hypotheses verification. Second, PPF-based approaches are imperfect for multiframe point clouds that contain a sequence of point clouds in a time series and are often encountered in industrial applications. The consistency information between frames is often ignored by the previous 6-D pose estimation approaches for point clouds.

In this article, we propose a new PPF-based framework for 6-D pose estimation with an emphasis on intelligent scene reference points sampling. Our observation is based on that a large number of votes indicates that the points around the reference point have a high probability on the instance's surface, while a few votes indicate low probability. Thus, the number of votes can be used to construct a probability map of the instance position in the scene. We propose to use the MSPSO algorithm [42], [43] for finding valid reference points which are located on the instance's surface. As a result, we propose the method which can combine MSPSO and probability map to iteratively sample scene reference points [see Fig. 1(c)]. In this way, we can improve the robustness of our algorithm to occlusion and complex scenes and reduce the computational burden. Besides, we further extend this framework to multiframe point clouds by considering similar object information between adjacent scene frames. It can utilize the prior knowledge in previous frames to improve the matching efficiency in real industrial applications. In summary, the main contributions of this work include the following.

1) A novel MSPSO sampling approach based on a probability map for progressively selecting valid scene reference points, which can simultaneously improve the efficiency and accuracy of 6-D pose estimation.
2) An effective pose clustering and hypothesis verification strategy to improve the robustness of 6-D pose estimation of point clouds.
3) A simple probability map-based approach for object detection from multiframe point clouds, which exploits the correlation between frames and can significantly reduce the number of invalid scene reference points, thus, speed up the matching process.

II. Methodology Overview

Our input includes an *object* $\mathcal{O}$ represented by either a surface mesh or a point cloud, and a real-scanned *scene* point cloud $\mathcal{S}$.

Each occurrence of the object in the scene is referred to one of its *instances* $\mathcal{I}$. The goal of our approach is to correctly determine the 6-D poses of all instances in the scene. We first shortly explain the main underlying ideas and concepts of the original *Drost-PPFM* method [36].

A. Pose Estimation Based on Point Pair Features

1) Point Pair Feature: It is a global feature descriptor, which describes the relative relationship between the position and direction of the two points. Given a reference point $\mathbf{p}_r$ and a target point $\mathbf{p}_t$ with normal $\mathbf{n}_r$ and $\mathbf{n}_t$, respectively, the $\mathbf{PPF}$ is a 4-D vector which is defined as

$$\mathbf{PPF}(\mathbf{p}_r, \mathbf{p}_t) = \left(||\mathbf{d}||_2, \angle(\mathbf{n}_r, \mathbf{d}), \angle(\mathbf{n}_t, \mathbf{d}), \angle(\mathbf{n}_r, \mathbf{n}_t)\right) \quad (1)$$

where $\mathbf{d} = \mathbf{p}_t - \mathbf{p}_r$, $\angle(\mathbf{a}, \mathbf{b})$ denotes the angle between vectors.

2) Drost-PPFM: It detects instances and estimates their poses via two main phases: offline global model description and online matching. In the offline phase, the object is described by a *model PPF hash table*, which is created from PPFs between each points pair. Specifically, this method calculates the PPF of each point pair and discretizes the distance and angle elements in PPF with a quantization step size of Δ_{dist} and Δ_{angle}, respectively. Then the point pairs with equal discrete feature vectors are recorded in the same hash node.

In the online phase, Drost-PPFM mainly performs feature matching and pose clustering. First, the scene point cloud is down-sampled to generate a point set $\mathcal{S}'$, of which 1/5 points are uniformly sampled as the scene reference points. To generate hypothetical poses, the scene PPFs, which are calculated by each scene reference point and all other points in $\mathcal{S}'$, are matched by the object model PPF hash map and the pose votes are cast by performing Hough voting. Finally, the hypothetical poses are clustered to improve robustness and the pose in the cluster with the highest accumulated weight is returned as the optimal pose. More details about these procedures can be found in the original paper [36].

B. Overview of Our Approach

We present a new 6-D pose estimation algorithm, called *PPFM-MSPSO*, which consists of three main components (see Fig. 1). First, we down-sample the input model and scene point cloud, then generate a model description hash map. Second, we generate the hypothetical poses by progressively sampling scene reference points based on an MSPSO algorithm and a probability map. The sampling method can exploit the voting information of each reference point to guide the search direction of MSPSO and significantly improve the accuracy of pose estimation by increasing the ratio of valid scene reference points. Finally, a pose verification operation is performed to filter out false poses due to the interference of noise and background and predict appropriate 6-D poses of the target object.

III. Method

A. Preprocessing

Generally, the input 3-D object and scene point clouds have a high density with a large number of points. The input is usually downsampled to obtain a spare set of points to speed up the matching process. However, traditional randomly or uniformly downsampling methods based on voxelization may ignore some important geometrical shape features. To solve this problem, we adopt an adaptive multiscale voxel downsampling method [9], which takes the normal vectors of the point cloud into account. Specifically, we first discretize the point cloud by creating a multiresolution grid structure. Then in each voxel cell at each level (in a fine-to-coarse order), the similar points, whose normals angle difference is less than a threshold θ, are merged. As a result, the geometrical features (e.g., edges or large curvature surface) are more preserved to sample discriminative points.

B. MSPSO Voting Module

The selection of scene PPFs has a great influence on the performance of the algorithm. With the unknown number and locations of the object instances in the scene, a large number of reference scene points need to be sampled to cover as many instances as possible, leading to a high computational cost of voting. In addition, invalid reference points generate many invalid votes that increase the amount of calculation for hypothetical poses clustering and verification, as well as reduce the matching accuracy.

Instead, we present a new voting module where the scene reference points are progressively sampled with the MSPSO algorithm to better exploit the knowledge obtained in the previous voting. A probability map is utilized to guide the MSPSO sampling. Thus, the MSPSO voting process includes three steps [see Fig. 1(c)]: the probability map initializing and updating, MSPSO sampling, and scene PPFs voting. We first initialize the probability map for scene sampling. Then, we search the optimal particle position based on the probability map for each subpopulation and select the scene point where the particle is located as a reference point for Hough voting. In the voting process, those reference points generate hypothetical poses by Hough voting [36] and update the probability map with the maximum number of votes to motivate or punish the search path of MSPSO.

We next introduce the probability map and MSPSO sampling in detail.

Probability map: The probability map P_{map} assigns a value ρ_i to each point in the down-sampled point cloud $\mathcal{S}'$, where ρ_i indicates the probability that this point is located on one of the real instances. Initially, each value ρ_i of the probability map is set to a constant. After a selected reference scene point $\mathbf{s}_r$ generates votes, the probability of $\mathbf{s}_r$ is calculated as following:

$$\rho_{\mathbf{s}_r} = \frac{V_{\mathbf{s}_r} - V_{\min}}{V_{\max} - V_{\min}} \quad (2)$$

where $V_{\mathbf{s}_r}$ is the maximum number of votes cast by $\mathbf{s}_r$, $V_{\max}$ and $V_{\min}$ are the number of votes when the probability is 1 and 0, respectively. Afterward, the probability ρ_i for other point in $\mathcal{S}'$

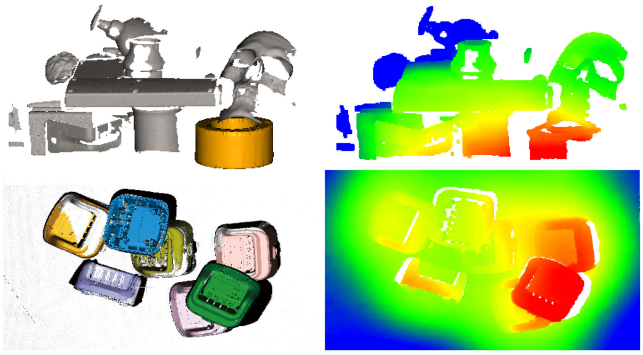


Fig. 2. Visual results of our generated pose estimation guided by the cloud probability map. Left: Final pose estimation where the detected objects are highlighted in color. Right: Probability map where the warmer color indicates a higher probability for locating the objects.

is updated according to $\rho_{\mathbf{s}_r}$

$$\rho_i = \frac{\sum_{j=1}^{N_\mathbf{s}} \omega_{j,\mathbf{s}_r} * \rho_{j,\mathbf{s}_r}}{\sum_{j=1}^{N_\mathbf{s}} \omega_{j,\mathbf{s}_r}} \tag{3}$$

where $N_\mathbf{s}$ is the number of scene reference points that have been voted. $\rho_{j,\mathbf{s}_r}$ is the probability obtained by voting with jth scene reference point. $\omega_{j,\mathbf{s}_r}$ is determined by a Gaussian kernel function, $\omega_{j,\mathbf{s}_r} = exp(-\frac{||\mathbf{p}_{j,\mathbf{s}_r}-\mathbf{p}_i||^2}{2*\sigma^2})$, where σ is the width parameter that is set to $\sigma = 0.25 d_{obj}$ (d_{obj} is the diagonal length of the object's bounding box).

However, updating ρ_i for all points is time consuming. To speed it up, we only update the neighboring points around $\mathbf{s}_r$, i.e., the points whose distance to $\mathbf{s}_r$ is less than $r_{\max} * d_{\text{obj}}$, and $r_{\max}$ is the update radius coefficient. Several examples of the resulting probability map are shown in Fig. 2.

MSPSO sampling: Different from *Drost-PPFM*, we present a novel MSPSO sampling algorithm to increase the sampling ratio of valid scene reference points and reduce the number of sampling points. We solve this problem by iteratively optimizing the position of the scene reference point in the search space with regard to the above probability map.

Here, we choose the MSPSO algorithm instead of the traditional particle swarm optimization (PSO) [44], [45], which has better global search ability than PSO and is more suitable for searching the best scene reference point on the probability map. In MSPSO, a scene reference point is regard as a particle, and each particle has a flying velocity, which is composed of three parts: the upper flying velocity, the optimal historical position of the particle, and the subpopulation

$$\begin{aligned} v_{\mathbf{s}_r}^{d,t+1} = \omega * v_{\mathbf{s}_r}^{d,t} + c_1 * r_1 * (pbest_{\mathbf{s}_r}^{d} - x_{\mathbf{s}_r}^{d,t}) \\ + c_2 * r_2 * (gbest_{\mathbf{s}}^{d} - x_{\mathbf{s}_r}^{d,t}) \end{aligned} \tag{4}$$

where $v_{\mathbf{s}_r}^{d,t}$ and $x_{\mathbf{s}_r}^{d,t}$ are, respectively, the velocity and position value of the dth dimension of the particle $\mathbf{s}_r$ in the $\mathbf{s}$ subpopulation in the tth iteration time. ω is the inertia factor between 0 and 1 (set to 0.7 by default), which controls the tradeoff between

Algorithm 1: MSPSO Voting With Probability Map P_{map}.

Input: Model PPF hash table, downsampled scene could $\mathcal{S}'$, MSPSO iteration numbers t_{max}, subpopulation size N, and scene reference point sampling rate τ.

Output: a set of hypothetical poses

1: initialize P_{map};
2: $n_s \leftarrow |\mathcal{S}'| * \tau$; $G \leftarrow int(n_s/(t_{max} * N)) + 1$;
3: **for** each subpopulation $\mathbf{s} \in \{\mathbf{s}_i | i = [1, G]\}$ **do**
4: $\mathbf{s} \leftarrow$ rejection sampling of P_{map} on $\mathcal{S}'$;
5: Initialize $pbest_{\mathbf{s}_r}$ and $gbest_{\mathbf{s}}$;
6: **for** $t \leftarrow 1$ **to** t_{max} **do**
7: **for** each particle $\mathbf{s}_r \in \mathbf{s}$ **do**
8: $\mathbf{s}_r$ votes and produces hypothetical poses;
9: update $pbest_{\mathbf{s}_r}$ and $gbest_{\mathbf{s}}$;
10: update P_{map} with (2) and (3);
11: **end for**
12: **for** each particle $\mathbf{s}_r \in \mathbf{s}$ **do**
13: **for** $k \leftarrow 1$ **to** k_u **do**
14: calculate velocity $\mathbf{v}_{\mathbf{s}_r}$ with (4);
15: calculate position $\mathbf{x}_{\mathbf{s}_r}$ with (5);
16: $\mathbf{s}_r' \leftarrow$ the nearest point of $\mathbf{x}_{\mathbf{s}_r}$;
17: $r \leftarrow rand(0, 1)$;
18: **if** $r < \rho_{\mathbf{s}_r'}$ **then**
19: $\mathbf{s}_r \leftarrow \mathbf{s}_r'$; break;
20: **end if**
21: **end for**
22: **if** $k > k_u$ **then**
23: $\mathbf{s}_r \leftarrow$ rejection sampling of P_{map} on $\mathcal{S}'$;
24: **end if**
25: **end for**
26: **end for**
27: **end for**

global and local experience. $pbest_{\mathbf{s}_r}^{d}$ is the dth dimension personal best position of particle $\mathbf{s}_r$ in $\mathbf{s}$ subpopulation found, and $gbest_{\mathbf{s}}^{d}$ is the dth dimension best position found by any particle in $\mathbf{s}$ subpopulation; c_1 is the cognitive acceleration constant, c_2 is the social acceleration constant, usually c_1=c_2=2; r_1 and r_2 are two random numbers uniformly distributed in the range [0,1). Therefore, the new position of the particle is updated by

$$x_{\mathbf{s}_r}^{d,t+1} = x_{\mathbf{s}_r}^{d,t} + v_{\mathbf{s}_r}^{d,t+1}. \tag{5}$$

MSPSO voting: The detailed MSPSO voting process is shown in Algorithm 1. Specifically, given the downsampled scene cloud $\mathcal{S}'$, we first calculate the number of subpopulation G

$$G = int(n_s/(t_{max} * N)) + 1 \tag{6}$$

where $n_s = |\mathcal{S}'| * \tau$, $t_{\max}$ is the number of iterations, N is the size of the subpopulation, and τ is scene reference point sampling rate (line 2). Then, we sample and vote on each subpopulation in turn (lines 3-27). In each subpopulation $\mathbf{s}$, we first initialize each particle $\mathbf{s}_r$ by the rejection sampling algorithm based on the probability map P_{map} (line 4). Then,

we iteratively perform $t_{\max}$ times of optimization voting, which includes three steps: $\mathbf{s}_r$ as scene reference point exercising PPFs Hough voting to generate hypothetical poses (line 8), updating P_{map} (line 10), and optimizing positions of the particles (lines 12–25).

In addition, (5) cannot be directly applied to optimize particles' positions, because the search space of traditional MSPSO is continuous, while the point cloud $\mathcal{S}'$ is a discrete point set in the 3-D space. In our algorithm, we first calculate the updated position $\mathbf{x}_{\mathbf{s}_r}$ of the particle $\mathbf{s}_r$ with (4) and (5) (lines 14–15). Then we find the nearest scene point $\mathbf{s}_r^*$ of $\mathbf{x}_{\mathbf{s}_r}$ (line 16). Finally, we judge whether $\mathbf{s}_r^*$ is the new position of the particle $\mathbf{s}_r$ (lines 17–20). In general, these steps can be iterated until the particle $\mathbf{s}_r$ is updated. To improve the global optimization performance, we stop this iteration and sample $\mathbf{s}_r$ on $\mathcal{S}'$ by the rejection sampling of P_{map} if the particle $\mathbf{s}_r$ is not updated after two iterations.

C. Pose Generation and Verification

The above MSPSO voting scheme produces a set of pose votes. Unfortunately, the candidate object poses contain a significant fraction of incorrect poses, thus, the hypothetical pose with the highest vote is not necessarily the correct match. Next, we conduct a pose verification step to search for the optimal pose in the hypothetical poses set.

1) Pose Clustering: The candidate object poses are redundant because the poses of the reference points on the same instance are similar. Grouping similar poses can improve the efficiency of pose verification. To measure the similarity of poses, we define a point triplet $\mathbf{FP}^c_{\text{obj}}$, which consists of the center point $\mathbf{c}_{\text{obj}} = (c_x, c_y, c_z)$ of the object's bounding box and two auxiliary points $\mathbf{p}^1_{\text{aux}} = (c_x - d_{obj}, c_y, c_z)$ and $\mathbf{p}^2_{\text{aux}} = (c_x, c_y - d_{\text{obj}}, c_z)$. Specifically, the similarity of the poses are calculated as following:

$$e_{i,j} = D_{\text{Chebyshev}}((\mathbf{T}_i * \mathbf{FP}^c_{\text{obj}}), (\mathbf{T}_j * \mathbf{FP}^c_{\text{obj}})). \tag{7}$$

In the clustering process, we first sort the hypothetical poses set $\mathbf{T}$ in descending order with respect to the number of votes. Then we consider each pose in $\mathbf{T}$ and judge whether they are similar to one of the subsequent poses. If it is true, we add the latter pose to the group of former poses, and the votes are accumulated. Finally, the poses in each cluster are averaged.

2) Hypotheses Verification: After pose clustering, we obtain a set of new pose hypotheses with voting scores. Due to sensor noise and background clutter, the poses with the highest voting score may not be the correct pose, therefore further verification is still required. Our verification indicator is the degree of overlap between the transformed model and the scene. Specifically, if one transformed model point $\mathbf{p}'_m$ and its closest scene point $\mathbf{p}_s$ meet the condition: $||\mathbf{p}'_m - \mathbf{p}_s||_2 < \varepsilon_D$ and $\angle(\mathbf{n}_{\mathbf{p}'_m}, \mathbf{n}_{\mathbf{p}_s}) < \varepsilon_A$, where ε_D is a distance threshold and ε_A is an angle threshold, we call these two points are overlapping. To recalculate the pose score, we first transform the object model into the scene with the pose, and find the nearest point of each model point in the scene and determine whether they are overlapping, and finally denote the pose verification score as the ratio of the number of overlapping points to the number of model points. Among all pose hypotheses, the pose with the highest verification score is the optimal matching pose.

D. Multiinstances, Multiobjects, and Multiframes Detection

To detect multiple instances in the scene, if only the top k cluster center poses are returned, this will easily lead to duplicate or missing poses. We apply a nonmaximum suppression (NMS) algorithm to filter out duplicate poses. In our implementation, we return the pose with the highest score as the first instance pose. Then we visit other poses in descending order and calculate the intersection over union (IOU) of the visited pose and the already returned instance poses. If the IOU is smaller than a threshold (e.g., 0.4), we accept the pose as a new instance pose, otherwise, we discard it. We iterate this process until k instance poses are retrieved.

To solve the multiple object detection, we first vote to generate hypothetical poses and cluster them for all objects, then merge the hypothetical poses of all objects and sort them by overlapping area size. Finally, we filter out inaccurate poses based on the aforementioned NMS algorithm.

Furthermore, in most industrial applications (such as the bin-picking task), we usually need to capture the scene and detect objects in successive frames. Such multiframe data are often overlooked in previous literature. Since the object information is similar between adjacent frames, using the prior knowledge in previous frames can significantly reduce the number of invalid scene reference points, thus, speed up the matching process. To make full use of such information, we propose a strategy for initializing and updating the probability map, which is different from the single-frame case. In the initialization phase of the probability map, we first transform the model point cloud into the current scene through

$$\mathcal{O}_c = \mathbf{T}_{p\to c} * \mathbf{T}_p * \mathcal{O} \tag{8}$$

where $\mathbf{T}_{p\to c}$ is the rigid transformation from the previous frame to the current frame. $\mathbf{T}_p$ is the instance pose estimated in the previous frame. Then, the probability $\rho_{i,m}$ of scene points that overlap with the transformed model $\mathcal{O}_c$ are assigned $\rho^o_{i,m}$ (default as 0.9), and the points without overlap are assigned $\rho^{\bar{o}}_{i,m}$ (default as 0.1). Furthermore, we set the scene reference point sampling rate τ to $1/80$.

Finally, different from the lack of prior knowledge when we assign the initial value of the probability map in a single frame application, the prior knowledge obtained in the previous frame has high credibility and is helpful for scene reference point sampling. To exploit the prior knowledge, we merge the initial value into the updating process of the probability map, which is formulated as

$$\rho_i = \frac{\rho_{i,m} + \sum_{j=1}^{N_{\mathbf{s}}} \omega_{j,\mathbf{s}_r} * \rho_{j,\mathbf{s}_r}}{1 + \sum_{j=1}^{N_{\mathbf{s}}} \omega_{j,\mathbf{s}_r}}. \tag{9}$$

E. Computational Complexity

Our approach includes three stages: preprocessing, MSPSO voting, and pose verification. The preprocessing uses the voxel downsampling algorithm, and its time complexity is $O(n)$ where n is the number of points in the input point cloud. In the MSPSO voting phase, the most complicated calculation is the PPF Hough voting. The PPF target point sampling adopts the smart sampling strategy of Hinterstoisser *et al.* [9], and the PPF reference point is sampled by our MSPSO based on the probability map. The time complexity of this stage is $O(kn_s)$, where k is usually at least one magnitude smaller than $n_{S'}$ (please refer to [9] for details), $n_{S'}$ is the number of points in the down-sampled scene point clouds, and $n_s = n_{S'} * \tau$. Our experiments have suggested that the best accuracy can be achieved when τ is set to $1/10$. Finally, the time complexity of the pose verification is $O(n_{T'} n_{\mathcal{O}'} n_{\mathcal{S}'}^{\frac{2}{3}})$, where $n_{T'}$ is the number of poses after clustering hypothetical poses, and $n_{\mathcal{O}'}$ is the number of points in the object down-sampled point clouds. The operation at this stage is mainly to find the nearest neighbors by using a KD-tree. In general, the scale of the down-sampled point cloud is around 2000 or less, and the number of hypothetical poses is about 500 or less, so the verification can be done quite efficiently.

F. Discussion

Our work has two differences from the previous studies. First, we present a new voting module where the scene reference points are progressively sampled to better exploit the knowledge obtained in the previous voting. It can increase the inlier rate, retain more instance surface information, and improve computational efficiency and robustness of the algorithm. The sampling rate of reference points is reduced from 20% of Dorst *et al.* [36] to 10%, while we can still achieve better accuracy. Note that Hinterstoisser *et al.* [9] proposed a smart sampling algorithm for sampling the second point in each point pair and achieved satisfactory results. However, we have a different purpose, where our probability map-based MSPSO is to effectively sample the scene reference points, while the smart sampling algorithm aims to optimize the sampling for the PPF target points.

Second, previous 6-D pose estimation methods based on pure point clouds have not yet considered the correlation between frames. In our method, we novelly transfer the information from the previous frame to the current frame with probability, which significantly reduces the number of invalid scene reference points, and thus, speeds up the matching process. Our experiments show that under the premise of ensuring the recognition rate, the sampling rate of the scene reference points can be reduced to 1.25% or even lower.

IV. Experimental Results

In this section, we first evaluate the proposed algorithm and verify its effectiveness by conducting ablation studies. Then we demonstrate the performance qualitatively and quantitatively through visually inspecting our results and conducting a comparison with state-of-the-art approaches.

A. Experimental Setup

Datasets: We first carry out experiments on a real-scanning dataset, DTU [46], in which there are many cylindrical and flat 3-D object models that are more challenging for pose estimation. To show our application on real industrial data, we also built a small-scale dataset, called RIPCD, which consists of 9 objects and 100 scenes. This dataset also contains multiple instances and multiple objects. In addition, although our algorithm focuses on pose estimation of unorganized 3-D point clouds, we also compare with some state-of-the-art deep learning-based methods on two RGBD datasets, including LineMOD (LM) [2] and LineMOD occlusion (LM-O)[47].

Evaluation metric: Given an estimated 6-D object pose $\hat{\mathbf{T}}$ and the ground-truth pose $\bar{\mathbf{T}}$, we use the average distance metric (ADM) to measure the L_2 distance between the model points transformed by $\hat{\mathbf{T}}$ and $\bar{\mathbf{T}}$. We define the pose error e_{ADM} as

$$e_{\text{ADM}} = \max(\text{avg}_{\mathbf{x_1}\in\mathcal{M}} \min_{\mathbf{x_2}\in\mathcal{M}} ||\bar{\mathbf{T}}\mathbf{x_1} - \hat{\mathbf{T}}\mathbf{x_2}||_2, ||\bar{\mathbf{T}}\mathbf{c}_{\text{obj}} - \hat{\mathbf{T}}\mathbf{c}_{\text{obj}}||_2) \tag{10}$$

where $\mathcal{M}$ is the template model of the object, and we consider the distance between transformed object centers $\mathbf{c}_{\text{obj}}$.

We regard an estimated pose as positive if the pose error is less than a threshold ξ_e. Then the performance for detecting each object is quantitatively measured by using recognition rate (RR) that is the ratio of the number of true positive poses compared to the number of all its ground-truth poses. We also calculate the mean recall (MR) as the mean of the per-object recognition rates to evaluate the overall performance on one dataset

$$MR = \text{avg}_{\mathbf{o}\in O} \frac{\sum_{s\in S} |P(o,s)|}{\sum_{s\in S} |G(o,s)|} \tag{11}$$

where O and S are sets of templates and test scenes. $|P(o,s)|$ is the number of correctly detected poses and $|G(o,s)|$ is the number of ground-truth poses of object o in scene s.

Competitors: We select a commercial machine vision software MVTec HALCON[1] as a competitor because it contains the optimized and significantly improved implementation of the original Drost-PPFM. We denote it Drost-PPFM*. We also compare to an open-source method Buch-17 [34], which presents a new pose voting and clustering method by integrating a local feature-based recognition pipeline. Here, we test several representative 3-D local feature descriptors, including PPF [35], spin images (SI)[33], FPFH [48], and SHOT[49].

B. Evaluation

Parameter settings: We conduct experiments with different parameter settings to evaluate their influence on recognition accuracy. In this section, we mainly analyze the eight parameters of the MSPSO sampling algorithm. The results of detailed parameter analysis are illustrated in Fig. 3, where we vary one parameter while all other parameters are fixed. We also show the

[1][Online]. Available: https://www.mvtec.com/products/halcon/

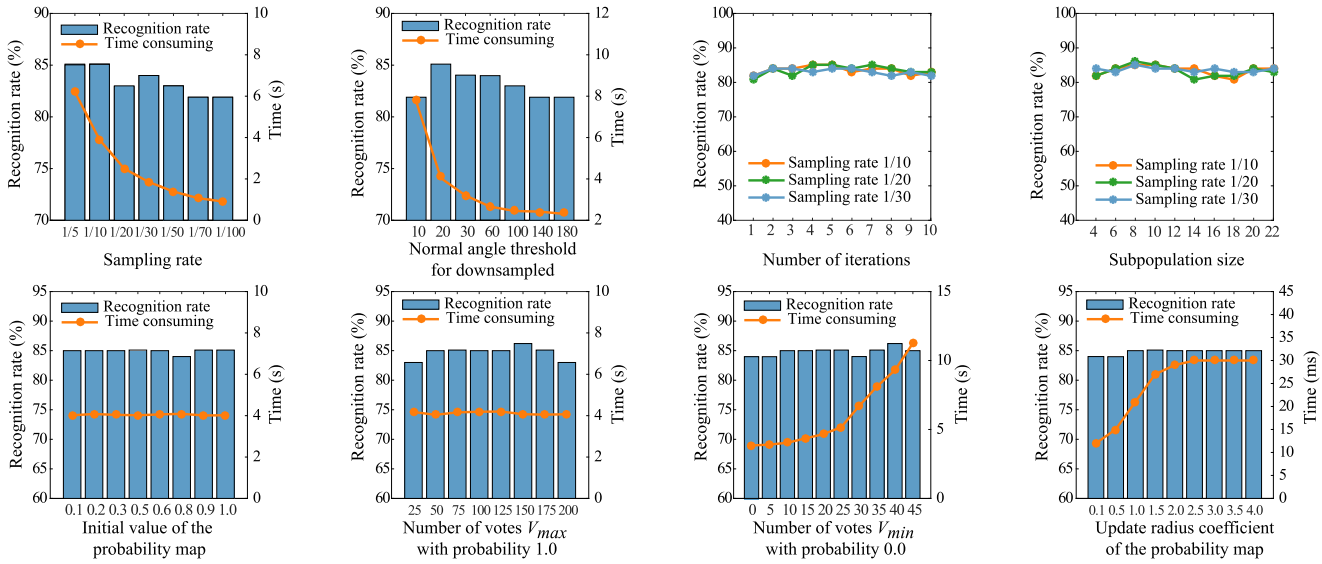


Fig. 3. Parameter analysis using a model from DTU [46]. To analyze each parameter, we set all other parameters as fixed values, which are the optimal values selected through experiments. If not specified, all parameters of our approach are set as default values: the sampling rates $\tau = 1/10$, the normal angle threshold for downsampling $\theta = 20°$, the iteration numbers $t_{\max} = 4$, the subpopulation size $N = 8$, the initial value of the probability map $\rho_i = 0.1$, the number of votes with probability 1.0 ($V_{\max} = 50$), the number of votes with probability 0.0 ($V_{\min} = 10$), and the update radius coefficient $r_{max} = 1.0$.

running times with different sampling rates τ. From the figure, we can observe that the parameter τ affects the performance of our method, where the recognition rate and running efficiency are reduced when the number of sampled scene reference points decreases. We find that when $\tau = 1/10$, the recognition accuracy and running time can be well balanced. Next, we can observe that the recognition rate has reached the best value when $r_{\max} = 1.0$, and the running time increases with the increase of the update radius coefficient in the early stage and remains stable in the later stage. Similarly, parameters θ and $V_{\min}$ affect our performance, and we can achieve the best performance when $\theta = 20°$ and $V_{\min} = 10$. In addition, our method is robust to $t_{\max}$, N, ρ_i, and $V_{\max}$.

Ablation studies: We now investigate the influence of different components of our method. We test five configurations as follows.

1) *Random sampling:* We randomly sample $|\mathcal{S}'| * \tau$ scene reference points in the down-sampled scene cloud $\mathcal{S}'$.
2) *Uniform sampling:* We uniformly sample the scene reference points, i.e., every ith ($i = 1/\tau$ in default) scene point is selected as a reference scene point.
3) *MSPSO:* The scene reference points are progressively sampled with MSPSO but without the guidance of the probability map.
4) *PSO+Probability map:* We replace the MSPSO with PSO.
5) *MSPSO+Probability map:* This is our full method.

First, Fig. 4 shows the inlier ratio of these algorithms at different sampling rates. Here, the inlier ratio represents the ratio of valid reference points on the instance's surface to the total scene reference points. We observe that using particle swarm optimization (MSPSO or PSO) increases the inlier ratio, while

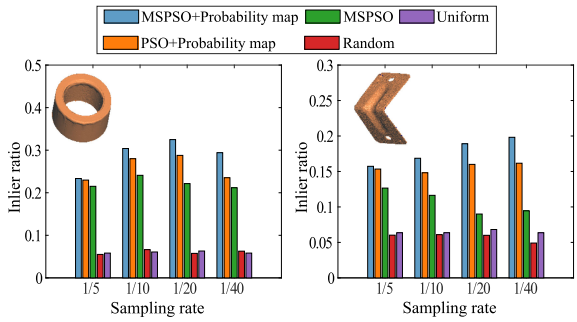


Fig. 4. Ablation studies of five sampling configurations.

MSPSO+Probability map and *PSO+Probability map* guided by the probability map could further improve the performance, which is approximately 3 to 5 times that of random and uniform sampling. In comparison, our MSPSO is more effective than PSO.

Further, we analyze the recognition rate of these methods at different sampling rates. The experimental results are shown in Table I. We see that we obtain the best performance with *MSPSO+Probability map*, while the performance of conventional random and uniform sampling is the worst.

Multi-instance and multiobject detection: Fig. 5 verifies the feasibility of our method for solving multi-instances and multiobjects detection and pose estimation problem. In Fig. 5, the input object point cloud is sampled from CAD models, while the input scene point cloud is captured by a structured-light scanner. If a scene consists of more than one object, we perform the steps of MSPSO sampling, PPF voting, pose clustering, and verification in parallel for all objects (this way we can also

TABLE I
RESULTS OF ABLATION EXPERIMENTS USING TWO MODELS FROM DTU DATASET [46]. THE BEST RESULT OF EACH MEASUREMENT IS MARKED IN **BOLD**.

Methods	DTU-13-Tape							DTU-28-Angle_bar						
Sampling Rate	1/5	1/10	1/20	1/40	1/50	1/100	Mean	1/5	1/10	1/20	1/40	1/50	1/100	Mean
MSPSO + Probability map	**85.1**	**85.1**	**83.0**	**85.1**	83.0	**81.9**	**83.9**	**58.0**	**55.7**	**53.4**	**52.3**	**51.1**	**35.2**	**51.0**
PSO + Probability map	83.0	83.0	81.9	81.9	**84.0**	78.7	82.1	52.3	54.5	52.3	47.7	44.3	33.0	47.4
MSPSO	83.0	84.0	**83.0**	84.0	**84.0**	78.7	82.8	56.8	48.9	51.1	50.0	44.3	31.8	47.2
Random	84.0	81.9	81.9	79.7	78.7	78.7	80.8	50.0	48.9	48.9	46.6	43.2	30.7	44.7
Uniform	81.9	83.0	79.7	80.8	77.7	75.5	79.8	51.1	50.0	47.7	45.5	44.3.2	31.8	45.2

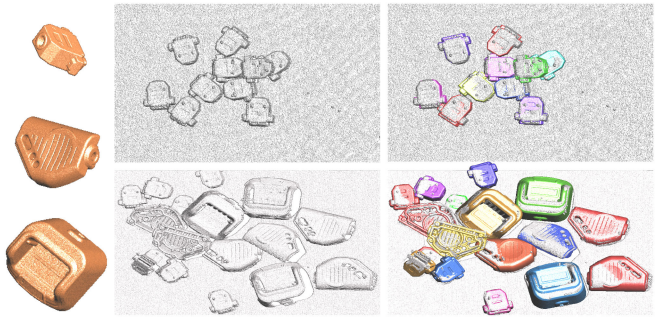


Fig. 5. Given one or more objects and a 3-D point cloud scene, our method detects all instances in the scene and estimates the 6-D pose for each instance.

TABLE II
AVERAGE RUNNING TIME(S) OF GRASPING 20 OBJECTS IN THE ROBOTIC ARM GRASPING SYSTEM. *MF* IS THE ABBREVIATION FOR OUR MULTIFRAME POINT CLOUD DETECTION ALGORITHM

Methods				
Drost-PPFM*	2.8	2.8	2.9	3.5
wtihout MF	5.4	21.6	19.2	26.4
wtih MF	2.1	2.8	2.2	3.9
wtih MF and parallel	1.1	1.5	1.3	1.9

distinguish different objects). Then in the final step of returning appropriate poses, we put the pose hypotheses of all objects together and rank them according to the overlapping information of each pose in the scene. From Fig. 5 we see that our method detects all of the instances for each kind of object and returns their correct 6-D poses.

Multiframe scenes: Finally, to demonstrate that we can speed up the matching process using multiframe data, we conduct tests on the robotic-arm grasping system. For comparison, we also report the running time without multiframe data as well as the time of Drost-PPFM*. The experimental results are shown in Table II. In a single-threaded environment, our method can obtain similar efficiency to Drost-PPFM*. Furthermore, we achieve the best performance through simple multicore parallel processing using the OpenMP library.

C. Comparisons

Comparison on DTU dataset: Table III shows the quantitative comparison of the performance score and timing on DTU, where the recognition rate of each object and each method is reported. The DTU dataset is quite challenging because of its high occlusion and clutter. We further avoid testing the simple feature-rich objects and select flat, cylindrical, and thin-edge objects without many local distinguishable features. The results show that our algorithm outperforms other competitors for most of the test objects. In terms of running time, the commercial software HALCON (Drost-PPFM*) is the fastest because it takes advantage of the hardware and each step is also fully optimized. Comparing to Buch-17, our method is reasonably faster, but each step can still be further accelerated on GPU to meet the needs of industrial applications.

Comparison on real industrial scenes: Next, we conduct a comparison on our proposed RIPCD dataset which is collected from real industrial scenes. Fig. 6 shows a qualitative comparison using several complex scenes for a robotic gripping task. The grasping objects involve common housings, polyhedrons, and threaded industrial devices. Similarly, we see that Drost-PPFM* and our approach achieve good performance on these scenes, but ours is still slightly better. Table IV reports the recognition rate on all of 9 objects in 100 scenes. This quantitative comparison verifies that our algorithm achieves the best-performing results on this dataset.

The surface geometric elements of industrial objects are simple. The local features of the point cloud are less discriminative. It is difficult to accurately estimate the pose for methods based on local features. The key to the success of Drost-PPFM* and our method is the excellent global description performance of PPF. Our method optimizes the algorithm for sampled scene reference points and hypothetical poses verification, which is more robust than Drost-PPFM*.

Comparison on LM and LM-O: Even though our method is not specially designed for RGB-D data, we compare to a series of state-of-the-art deep learning-based methods, including depth-only, RGB-only, and RGB-D methods. Table V shows the comparison results. Our method, which does not use multiframe information, obtains the best performance among the methods using only depth information. It outperforms CloudAEE+ICP [18] by 3.9% and 13.0% on LM and LM-O, respectively. Moreover, we achieve the same performance as PVN3D [28] on LM, which is one of state-of-the-art works using both RGB and depth information.

The objects in LM-O [47] are heavily occluded, making pose estimation more difficult. As expected, we observe a significant performance drop for each method. However, our algorithm still outperforms other competitors.

TABLE III
QUANTITATIVE COMPARISON ON DTU DATASET. THE POSE ERROR THRESHOLD IS SET TO 0.1

Methods	RR_2	RR_4	RR_{10}	RR_{12}	RR_{13}	RR_{14}	RR_{15}	RR_{16}	RR_{17}	RR_{20}	RR_{28}	RR_{31}	RR_{37}	RR_{41}	MR	Time(s)
Buch-17-PPF	67.1	71.2	62.4	59.4	11.7	48.7	**73.3**	14.7	85.0	15.3	19.3	**76.8**	62.4	29.6	55.7	1.71
Buch-17-SHOT	46.4	51.1	21.2	42.2	15.9	32.4	36.3	14.8	50.2	20.2	40.9	44.5	36.0	42.0	36.8	3.34
Buch-17-SI	64.3	72.3	46.5	48.4	3.2	39.9	62.3	21.3	75.4	18.0	45.5	61.9	61.4	43.2	51.4	1.49
Buch-17-FPFH	35.0	51.1	36.3	35.9	14.9	45.6	56.8	16.3	71.7	10.4	52.3	45.2	41.1	43.2	43.1	22.1
Drost-PPFM*	67.1	69.0	57.9	68.0	77.7	63.5	62.3	42.6	82.8	55.8	29.5	54.8	**66.0**	86.4	65.8	0.40
Ours	**70.0**	**69.9**	**68.6**	**71.1**	**85.1**	**71.4**	60.9	**50.8**	**86.6**	**71.4**	**58.0**	67.1	59.4	**98.1**	**72.4**	1.53

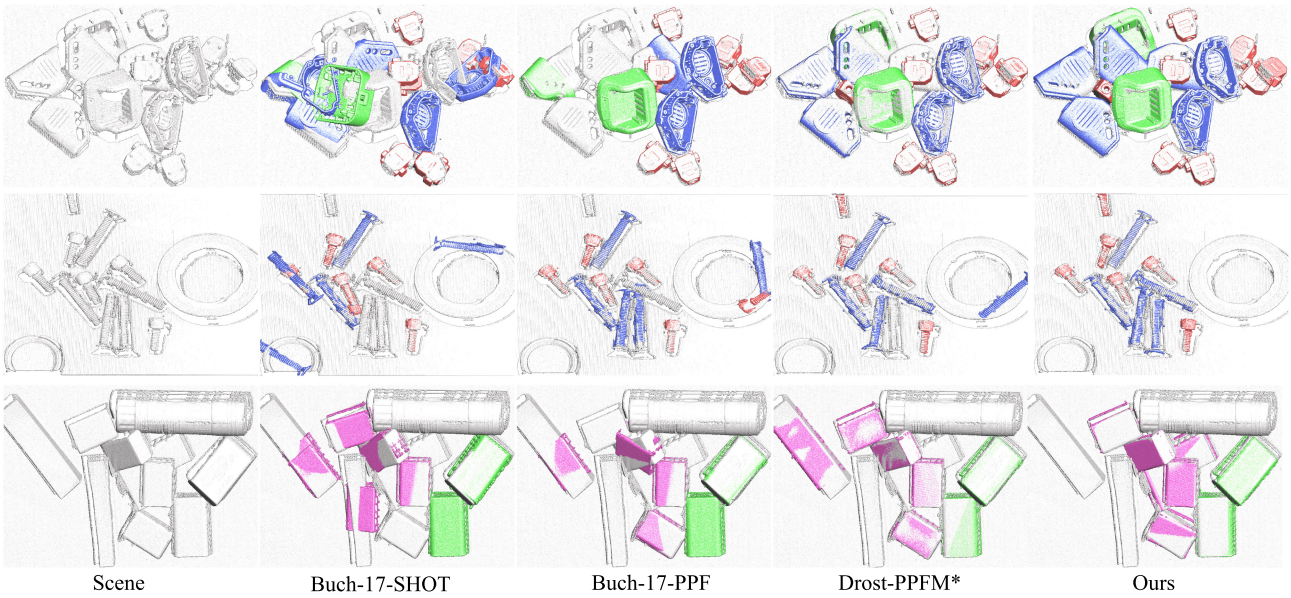


Fig. 6. Qualitative comparison using five scenes from our proposed RIPCD dataset.

TABLE IV
QUANTITATIVE COMPARISON ON RIPCD DATASET. THE POSE ERROR THRESHOLD IS SET TO 0.1

Methods	RR_{case1}	RR_{case2}	$RR_{connector}$	RR_{cube}	$RR_{eraser-big}$	$RR_{eraser-small}$	$RR_{nameplate}$	$RR_{screw-long}$	$RR_{screw-short}$	MR	Time(s)
Buch-17-PPF	24.2	56.1	**87.4**	66.8	35.0	24.5	**100.0**	20.0	76.1	65.2	5.36
Buch-17-SHOT	8.1	40.4	18.0	53.1	12.5	21.3	73.3	24.0	**94.4**	31.8	10.40
Buch-17-SI	0.0	31.6	62.6	66.4	57.5	12.8	**100.0**	20.0	93.0	52.4	6.65
Buch-17-FPFH	4.0	19.3	23.3	38.9	42.5	40.4	86.7	20.0	67.6	30.7	39.83
Drost-PPFM*	90.9	**100.0**	79.9	68.7	82.5	84.0	**100.0**	28.0	**94.4**	80.3	3.25
Ours	**97.0**	**100.0**	83.7	**76.8**	**95.0**	**86.2**	**100.0**	**40.0**	**94.4**	**85.1**	4.13

TABLE V
QUANTITATIVE EVALUATION OF 6-D POSE ON ADD(S) [50] METRIC ON THE LINEMOD [2] AND LINEMOD OCCLUSION DATASET [47]. THE BEST RESULT OF EACH MEASUREMENT IS MARKED IN **BOLD**. THE NAMES OF SYMMETRIC OBJECTS ARE ALSO IN **BOLD**

(a) LineMOD

Modality	RGB			RGBD				D		
Method	PoseCNN+ DeepIM [20], [21]	PVNet	CDPN [22]	SSD-6D +ICP [25]	PointFusion [26]	DF [27]	PVN3D [28]	CloudPose +ICP [17]	CloudAEE +ICP [18]	Ours
ape	77.0	43.6	64.4	65.0	70.4	92.3	97.3	58.3	92.5	**97.5**
bvise	97.5	99.9	97.8	80.0	80.7	93.2	99.7	65.6	91.8	**100**
cam	93.5	86.9	91.7	78.0	60.8	94.4	99.6	43.0	88.9	**99.7**
can	96.5	95.5	95.9	86.0	61.1	93.1	99.5	84.7	96.4	**99.7**
cat	82.1	79.3	83.8	70.0	79.1	96.5	**99.8**	84.6	97.5	99.7
driller	95.0	96.4	96.2	73.0	47.3	87.0	99.3	83.3	99.0	**99.8**
duck	77.7	52.6	66.8	66.0	63.0	92.3	**98.2**	43.2	92.7	98.0
e.box	97.1	99.2	99.7	**100**	99.9	99.8	99.8	99.5	99.8	**100**
glue	99.4	95.7	99.6	**100**	99.3	**100**	**100**	98.8	99.0	**100**
holep	52.8	82.0	85.8	49.0	71.8	92.1	**99.9**	72.1	93.7	98.7
iron	98.3	98.9	97.9	78.0	83.2	97	**99.7**	70.3	95.9	99.4
lamp	97.5	99.3	97.9	73.0	62.3	95.3	**99.8**	93.2	96.6	99.3
phone	87.7	92.4	90.8	79.0	78.8	92.8	99.5	81.0	97.4	**99.8**
average	88.6	86.3	89.9	79.0	73.7	94.3	**99.4**	75.2	95.5	**99.4**

(b) LineMOD Occlusion

Modality	Method	average
RGB	PoseCNN [20]	24.9
	PVNet [10]	40.8
RGBD	PoseCNN+ICP [20]	78.0
	PVN3D [28]	63.2
D	CloudPose+ICP [17]	44.2
	CloudAEE [18]	57.1
	CloudAEE+ICP [18]	66.1
	Ours	**79.1**

V. Conclusion

We presented an improved approach based on Drost-PPFM for 6-D pose estimation of disordered objects in 3-D point cloud scenes. We achieved major improvements in terms of efficiency and robustness by improving the sampling strategy for scene reference points. A new MSPSO algorithm based on the probability map could effectively improve the sampling of scene reference points. At the same time, the method of initializing the probability map with multiframe data was also proposed to improve the efficiency of object pose estimation. We demonstrated our advantages by comparing it to the SOTA methods on several datasets.

Future work will be to explore more efficient object detection and pose estimation methods. A possible direction is to design an end-to-end deep neural network based on the middle-level structural analysis or high-level semantic information. For example, instead of using point correspondences, we could extract middle-level geometric primitives (such as plane, cylinder, sphere, cone) from objects and scenes to build feature correspondences. Besides, using object detection or segmentation maybe also improve the estimation performance. In addition, we will further improve our algorithm to adapt it to structured RGB-D scenes.

References

[1] S. Hinterstoisser *et al.*, "Gradient response maps for real-time detection of textureless objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 876–888, May 2012.

[2] S. Hinterstoisser *et al.*, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 858–865.

[3] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas, "Detection and fine 3D pose estimation of texture-less objects in RGB-D images," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 4421–4428.

[4] A. Tejani, R. Kouskouridas, A. Doumanoglou, D. Tang, and T.-K. Kim, "Latent-class hough forests for 6 DoF object pose estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 119–132, Jan. 2018.

[5] Y. Konishi, K. Hattori, and M. Hashimoto, "Real-time 6D object pose estimation on CPU," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2019, pp. 3451–3458.

[6] Z. He, Z. Jiang, X. Zhao, S. Zhang, and C. Wu, "Sparse template-based 6-D pose estimation of metal parts using a monocular camera," *IEEE Trans. Ind. Electron.*, vol. 67, no. 1, pp. 390–401, Jan. 2020.

[7] S. Daftry *et al.*, "Machine vision based sample-tube localization for mars sample return," in *Proc. IEEE Aerosp. Conf.*, 2021, pp. 1–12.

[8] T. Hodan *et al.*, "Bop: Benchmark for 6D object pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 19–34.

[9] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige, "Going further with point pair features," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 834–848.

[10] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2019, pp. 4561–4570.

[11] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, "An augmented linear mixing model to address spectral variability for hyperspectral unmixing," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1923–1938, Apr. 2019.

[12] D. Hong, L. Gao, J. Yao, B. Zhang, A. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 7, pp. 5966–5978, Jul. 2021.

[13] D. Hong, N. Yokoya, J. Chanussot, J. Xu, and X. X. Zhu, "Joint and progressive subspace analysis (JPSA) with spatial-spectral manifold alignment for semisupervised hyperspectral dimensionality reduction," *IEEE Trans. Cybern.*, vol. 51, no. 7, pp. 3602–3615, Jul. 2021.

[14] D. Hong *et al.*, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 5, pp. 4340–4354, May 2021.

[15] D. Hong *et al.*, "Interpretable hyperspectral artificial intelligence: When nonconvex modeling meets hyperspectral remote sensing," *IEEE Geosci. Remote Sens. Mag.*, vol. 9, no. 2, pp. 52–87, Jun. 2021.

[16] F. Hagelskjær and A. G. Buch, "Pointvotenet: Accurate object detection and 6 DoF pose estimation in point clouds," in *Proc. IEEE Int. Conf. Image Process.*, 2020, pp. 2641–2645.

[17] G. Gao, M. Lauri, Y. Wang, X. Hu, J. Zhang, and S. Frintrop, "6D object pose regression via supervised learning on point clouds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 3643–3649.

[18] G. Gao, M. Lauri, X. Hu, J. Zhang, and S. Frintrop, "CloudAAE: Learning 6D object pose regression with on-line data synthesis on point clouds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 11081–11087.

[19] Y. Shi, J. Huang, X. Xu, Y. Zhang, and K. Xu, "Stablepose: Learning 6D object poses from geometrically stable patches," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2021, pp. 15 222–15231.

[20] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," *Robotics: Science and Systems (RSS)*, Pittsburgh, Pennsylvania, pp. 1–10, Jun. 2018, doi: 10.15607/RSS.2018.XIV.019.

[21] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "DeepIM: Deep iterative matching for 6D pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 683–698.

[22] Z. Li, G. Wang, and X. Ji, "CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 7678–7687.

[23] C. Wu, L. Chen, Z. He, and J. Jiang, "Pseudo-siamese graph matching network for textureless objects' 6D pose estimation," *IEEE Trans. Ind. Electron.*, pp. 1–1, 2021, doi: 10.1109/TIE.2021.3070501.

[24] Z. Yang, X. Yu, and Y. Yang, "DSC-PoseNet: Learning 6DoF object pose estimation via dual-scale consistency," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2021, pp. 3907–3916.

[25] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1521–1529.

[26] D. Xu, D. Anguelov, and A. Jain, "Pointfusion: Deep sensor fusion for 3D bounding box estimation," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2018, pp. 244–253.

[27] C. Wang *et al.*, "Densefusion: 6D object pose estimation by iterative dense fusion," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2019, pp. 3343–3352.

[28] 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," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2020, pp. 11 632–11 641.

[29] Y. He, H. Huang, H. Fan, Q. Chen, and J. Sun, "FFB6D: A full flow bidirectional fusion network for 6D pose estimation," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2021, pp. 3003–3013.

[30] W. Chen, X. Jia, H. J. Chang, J. Duan, L. Shen, and A. Leonardis, "FS-Net: Fast shape-based network for category-level 6D object pose estimation with decoupled rotation mechanism," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2021, pp. 1581–1590.

[31] C. Zhuang, Z. Wang, H. Zhao, and H. Ding, "Semantic part segmentation method based 3D object pose estimation with RGB-D images for bin-picking," *Robot. Comput.- Integr. Manuf.*, vol. 68, 2021, Art. no. 102086.

[32] R. König and B. Drost, "A hybrid approach for 6DoF pose estimation," in *Proc. Eur. Conf. Comput. Vis. Workshops*. Cham, Switzerland: Springer, 2020, pp. 700–706.

[33] 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.

[34] A. G. Buch, L. Kiforenko, and D. Kraft, "Rotational subgroup voting and pose clustering for robust 3D object recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 4137–4145.

[35] A. Buch and D. Kraft, "Local point pair feature histogram for accurate 3D matching," in *Proc. Brit. Mach. Vis. Conf.*, 2018, pp. 1–12.

[36] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, 2010, pp. 998–1005.

[37] C. Choi, Y. Taguchi, O. Tuzel, M. Liu, and S. Ramalingam, "Voting-based pose estimation for robotic assembly using a 3D sensor," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 1724–1731.

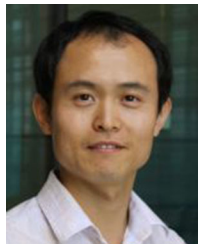
[38] B. Drost and S. Ilic, "3D object detection and localization using multimodal point pair features," in *Proc. Int. Conf. 3D Imag., Model., Process., Visual. Transmiss.*, 2012, pp. 9–16.
[39] T. Birdal and S. Ilic, "Point pair features based object detection and pose estimation revisited," in *Proc. Int. Conf. 3D Vis.*, 2015, pp. 527–535.
[40] J. Vidal, C.-Y. Lin, and R. Martí, "6D pose estimation using an improved method based on point pair features," in *Proc. Int. Conf. Control, Autom. Robot.*, 2018, pp. 405–409.
[41] 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, 2018, Art. no. 2678.
[42] I. Montri and S. Siriporn, "A multi-subpopulation particle swarm optimization: A hybrid intelligent computing for function optimization," in *Proc. Int. Conf. Natural Comput.*, 2007, pp. 679–684.
[43] L. Bi, W. Cao, W. Hu, and M. Wu, "Intelligent tuning of microwave cavity filters using granular multi-swarm particle swarm optimization," *IEEE Trans. Ind. Electron.*, vol. 68, no. 12, pp. 12901–12911, Dec. 2021.
[44] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. Int. Symp. Micro Mach. Human Sci.*, 1995, pp. 39–43.
[45] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, 1995, pp. 1942–1948.
[46] T. Sølund, A. G. Buch, N. Krüger, and H. Aanæs, "A large-scale 3D object recognition dataset," in *Proc. Int. Conf. 3D Vis.*, 2016, pp. 73–82.
[47] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6D object pose estimation using 3D object coordinates," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 536–551.
[48] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 3212–3217.
[49] S. Salti, F. Tombari, and L. Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Comput. Vis. Image Understanding*, vol. 125, pp. 251–264, 2014.
[50] S. Hinterstoisser *et al.*, "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes," in *Proc. Asian Conf. Comput. Vis.*, 2012, pp. 548–562.

Xuejun Xing received the B.S. degree from the China University of Petroleum, Dongying, China, in 2010. He is currently working toward the M.S. degree in computer engineering with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China.

His research interests include 6-D pose estimation, 3-D reconstruction, and machine vision.

Jianwei Guo received the bachelor's degree in software engineering from Shandong University, Jinan, China, in 2011 and the Ph.D. degree in computer science from the Chinese Academy of Sciences (CASIA), Beijing, China, in 2016.

He is an Associate Professor with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, CASIA. His research interests include 3-D vision, computer graphics, and image processing.



Liangliang Nan received the Ph.D. degree in mechanical and electronic engineering from the Graduate University of Chinese Academy of Sciences, Beijing, China, in 2009.

He is currently an Assistant Professor with the Delft University of Technology (TU Delft), Delft, Netherlands. Prior to joining TU Delft, he was a Research Scientist with KAUST, Thuwal, Saudi Arabia during 2013–2018, and an Associate Professor with SIAT, Beijing, China, during 2009–2013. His research interests include computer graphics, computer vision, and 3-D geoinformation.

Qingyi Gu (Member, IEEE) received the M.E. and Ph.D. degree in engineering from Hiroshima University, Higashihiroshima, Japan, in 2010, and 2013 respectively.

He is currently a Professor with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include high-speed image processing, real-time 3-D reconstruction, and applications in industry and biomedicine.

Xiaopeng Zhang (Member, IEEE) received the Ph.D. degree in computer science from the Institute of Software, Chinese Academic of Sciences, Beijing, China, in 1999.

He is a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His main research interests include image processing, computer graphics, and computer vision.

Dong-Ming Yan received the bachelor's and master's degrees in computer science and technology from Tsinghua University, Beijing, China, in 2002 and 2005, respectively, and the Ph.D. degree in computer science from Hong Kong University, Pok Fu Lam, Hong Kong, in 2010.

He is a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include computer graphics, computer vision, and pattern recognition.