Efficient Center Voting for Object Detection and 6D Pose Estimation in 3D Point Cloud

Jianwei Guo, Xuejun Xing, Weize Quan, Dong-Ming Yan, Qingyi Gu, Yang Liu, Xiaopeng Zhang

Abstract—We present a novel and efficient approach to estimate 6D object poses of known objects in complex scenes represented by point clouds. Our approach is based on the well-known *point pair feature* (PPF) matching, which utilizes self-similar point pairs to compute potential matches and thereby cast votes for the object pose by a voting scheme. The main contribution of this paper is to present an improved PPF-based recognition framework, especially a new center voting strategy based on the relative geometric relationship between the object center and point pair features. Using this geometric relationship, we first generate votes to object centers resulting in vote clusters near real object centers. Then we group and aggregate these votes to generate a set of pose hypotheses. Finally, a pose verification operator is performed to filter out false positives and predict appropriate 6D poses of the target object. Our approach is also suitable to solve the multi-instance and multi-object detection tasks. Extensive experiments on a variety of challenging benchmark datasets demonstrate that the proposed algorithm is discriminative and robust towards similar-looking distractors, sensor noise, and geometrically simple shapes. The advantage of our work is further verified by comparing to the state-of-the-art approaches.

Index Terms—6D pose estimation, 3D object recognition, Point pair features, 3D point cloud

1 Introduction

Object recognition and pose estimation in point cloud have been active in computer vision and robotics. The goal of a 6D pose estimator is to predict a rigid object pose, which is represented by a 4×4 matrix $\mathbf{T} = [\mathbf{R}, \mathbf{t}; \mathbf{0}, 1]$, consisting of a 3×3 rotation matrix $\mathbf{R}$ and a 3×1 translation vector $\mathbf{t}$. The pose aligns the object to a scene by transforming a 3D point $\mathbf{p}_o$ in the object coordinate system to a 3D point $\mathbf{p}_c$ in the camera coordinate system, *i.e.*, $\mathbf{p}_c = \mathbf{T}\mathbf{p}_o$. Finding accurate poses of the objects is a crucial prerequisite for various downstream applications, including robot bin-picking [1], autonomous driving [2], augmented reality [3], [4], indoor scene reconstruction [5], to name a few.

In the past years, a large number of literatures focus on detecting 3D objects and determining their poses in a 3D scene, which is represented in the form of either point clouds or RGB-D images. Early works [6], [7] rely on matching 3D local point descriptors to build a number of correspondences between the object and the scene, then the correspondences are used to generate 6D pose candidates which are further refined by using an *Iterative Closest Point* (ICP) algorithm. This kind of approaches achieves low recognition performance when detecting objects with uniform shapes, *i.e.*, objects with planar or repetitive structures. Recently, deep learning-based object pose recovery methods are proposed for RGB [8], [9] or RGB-D images [10], [11]. However, deep learned 6D pose estimators should be re-trained for each new detection environment, thus they cannot easily be generalized well to unseen object classes which are outside the training distribution. Besides, the improvements inspired by these works on RGB/RGB-D images are not directly tailored to 3D point cloud data.

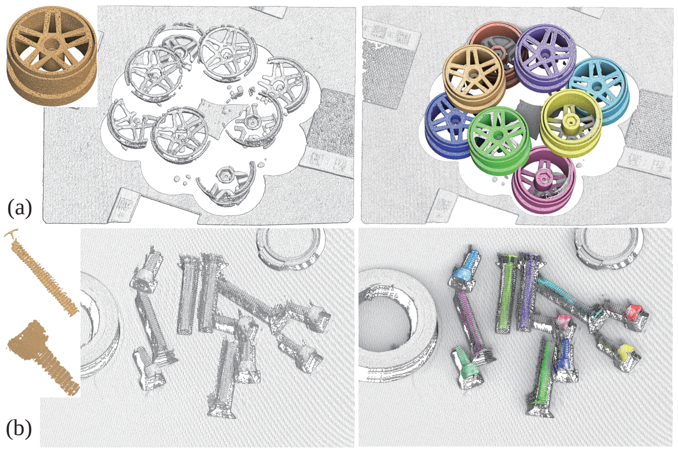

Fig. 1. Single (a) and multiple target (b) matching and 6D pose estimation in point clouds using our center voting scheme. Given one or more templates and a point cloud of a 3D scene, our method detects all instances of the templates in the scene, and predicts the 6D pose for each instance.

Drost et al. [12] propose a new recognition framework in a 'model globally and match locally' manner based on oriented point pair features (PPF). It operates purely on point clouds and outperforms other feature-based state-of-the-art approaches [13]. Since introduced, many variants [14], [15], [16], [17], [18], [19], [20] have been proposed to obtain better and faster recognition performance. These PPF matching methods have shown very successful object detection in a vast number of industrial and robotic applications. Al-

- *J. Guo, W. Quan, D.-M. Yan, Q. Gu and X. Zhang 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.*
- *X. Xing is with School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China.*
- *Y. Liu is with Suzhou CASIA All Phase Intelligence Technology Co.ltd.*

though being well studied, PPF-based approaches still have some issues that limit its performance: (1) it is sensitive to sensor noise, heavy occlusion and background clutter; (2) it has problems when recognizing rotationally symmetric objects without distinctive texture, which are often encountered in the industrial environments; (3) it is inefficient in terms of computation time because it relies on searching a large set of paired feature correspondences.

In this paper, we present a new object detection and 6D pose estimation method based on point pair features. We improve the whole pipeline of [12] and make significant adaptations in each crucial component to make PPF more discriminative and robust towards noise, occlusions and geometrically simple shapes. Our algorithm is based on an intuitive observation that the geometric relationship between the model PPF and object center is simple and clear. Thus we introduce a novel center-targeted voting scheme, where by voting we essentially generate new points near objects' centers in the scene, which can be aggregated through a clustering module to generate pose candidates. This introduces a simpler approach to compute transformation matrix from the matched point pairs. Then a hypotheses verification approach in context of the center voting module is proposed to find correct poses.

Previous PPF-based methods generate a pose vote for each scene reference point, then clustering is performed on the generated poses, which are easily affected by the scene noises. Unlike this, our approach first groups the voted object centers, and then generates one pose by averaging each group. In this way, we can improve the robustness against noise as well as reduce the computational burden to some extent. Furthermore, the object center is invariant under pose ambiguity, thus our approach has the ability to detect geometrically simple shapes. In summary, the main contributions of this work are as follows:

- An improved PPF-based recognition pipeline that is *intuitive, simple, and effective* to detect objects in point clouds and recover their 6D poses.
- A new center voting module based on the relative geometric relationship between the object center and point pair features. It provides a simpler but more accurate approach to compute potential poses.
- A new pose clustering and hypotheses verification approach that takes into account our center voting strategy. It supports multi-instance and multi-object detection.

2 Related work

In this section, we present an overview of the key techniques towards object pose estimation in 3D point clouds in past years. The relevant work for matching rigid objects from a single RGB-D input image are also briefly revisited. For more comprehensive discussions, we refer the reader to the survey papers [21], [22], [23].

Pose estimation from point clouds. Early works on 3D object recognition and pose estimation are based on local point descriptors to match feature points, such as local representation of differential surface properties [24], point signatures [25], spin images [26]. Following this idea, in [6], a local 3D shape context descriptor was used for matching segmented point cloud models. Taati and Greenspan [27] formulated the selection of a good local shape descriptor for 3D object recognition as an optimization problem with a number of variable-dimensional descriptors. Their descriptors have high precision and generalization. Buch *et al.* [7] thoroughly analyzed and evaluated several aspects of existing local shape descriptors on multiple datasets. Then, they introduced a RANSAC-based method to fuse several features to improve matching accuracy and achieved better generalization properties, but suffered from a cubic complexity in the number of feature correspondences. In a follow-up work, Buch *et al.* [28] proposed a more robust pose estimation method via rotational subgroup voting and pose clustering. The work of [13] presented a detailed comparison with the most popular local features. However, as mentioned earlier, these approaches performed unsatisfactorily when detecting objects with planar or constant-curvature surfaces, because there will be many different points which yield the same local feature.

Drost *et al.* [12] proposed one of the most promising algorithms by using point pair features to ease the process of matching 3D models to 3D scenes. We will give a general review of this approach in Sec. 3. Many researchers developed extensions to improve recognition accuracy and reduce computational complexity of this framework. Not only using surface information, Choi *et al.* [17] extracted additional edge or boundary information to construct point pair features in order to match industrial objects with planar primitives. Drost *et al.* [15] combined the stable information from two modalities, *i.e.,* the intensity and depth data, to propose a multimodal, scale- and rotation-invariant feature. Besides, an effective geometric edge extractor was designed to facilitate this process. Birdal and Ilic [19] improved the pipeline of [12] via combining several revised techniques. Specifically, they coupled the object detection with a coarse-to-fine segmentation, a weighted Hough voting and an interpolated recovery of pose parameters were applied during matching, and an occlusion-aware ranking was used for the testing and sorting of all the generated hypotheses. To further reduce the impact of clutter and sensor noise, Hinterstoisser *et al.* [20] proposed a novel sampling and voting schemes, together with minor modifications to the pre- and post-processing steps, to achieve competitive results. Vidal *et al.* [29] proposed a 6D pose estimation approach based on point pair features voting, along with a novel preprocessing step and an improved clustering step. In addition, they introduced view-dependent re-scoring process are proposed. Vidal *et al.* [30] further enhanced the work of [20], where a novel subsampling step with normal clustering and neighbor pairs filtering was introduced, then a faster kd-tree neighbor search is proposed for run-time. Two filtering postprocessing steps were also applied to discard special ambiguous cases. However, this approach combined various heuristics and it was too complex to implement. To accelerate the template matching approach based on point tuple features, Vock *et al.* [31] introduced two key improvements: during the scoring phase, they designed a voxel based scoring method with an early exit strategy; for the generation of transformation hypotheses, they introduced a novel sampling strategy which considers stable, salient points and exploits the locality of possible template occurrences. Although these methods yield competitive results, there is still some room for improvement, *e.g.,* sensitivity to noise and difficulties in the detection of geometrically simple objects.

Pose estimation from RGB/RGB-D images. Recovering 6D pose from 2.5D data is different from 3D point cloud. In the literature, several template matching approaches were proposed, where the best-known method is LINEMOD [32] for real-time 3D object instance detection. Subsequently, this method was generalized by [33] to incorporate new quantized cues: they proposed an efficient representation of templates that captures the multiple modalities, *i.e.,* image gradients and quantized surface normals, for better performance. Hodaň *et al.* [34] proposed another method for the detection and accurate 3D localization of multiple texture-less objects. They addressed the excessive computational complexity of sliding window approaches by fast filtering and voting procedure based on hashing. The main drawback of these methods is that they are not robust to distortions along object borders caused by clutter and occlusions.

For learning-based systems, random forest is firstly used for predicting both 3D object coordinates and object instance probabilities, such methods include [35], [36], [37]. Recently, Wohlhart and Lepetit [8] learned the descriptors using the convolutional neural network by enforcing simple similarity constraint on poses of the same object and dissimilarity constraint on different objects. This work uses a scalable nearest neighbor search to evaluation and delivers impressive results in terms of object retrieval and pose estimation. Kehl *et al.* [10] proposed a local-patch-based deep learning method to solve object detection and pose estimation. They employed a convolutional auto-encoder model pre-trained on a large collection of random local patches to regress the patch descriptor, matched them against codebooks of descriptor of synthetic patches and finally casted 6D votes using a threshold. Sundermeyer *et al.* [38] implicitly learned 3D orientation via an augmented auto-encoder, which is trained to encode 3D model views in a self-supervised way. Wang *et al.* [11] presented a new framework, called DenseFusion, in which a heterogeneous architecture is used to process the RGB image and depth individually, then a dense fusion network is applied to extract pixel-wise dense feature embedding. In addition, an iterative pose refinement procedure is integrated to further improve the pose estimation while achieving near real-time inference. Gao *et al.* [39] presented a deep learning system that regresses 6D object poses from depth information. They first perform semantic segmentation then use two separate networks for rotation and translation regression. There is a benchmark for 6D object pose estimation (BOP) [22], [40] to continuously report the state of the arts in estimating the 6D pose of rigid objects from RGB or RGB-D images. In the latest report[1], a variety of recent deep-learning-based methods have been evaluated, including CosyPose [41], Pix2Pose [42], CDPN [43], Hybrid-DL-PointPairs [44], leaping from 2D to 6D [45], EPOS [46], PointVoteNet [47], SingleMultiPathEncoder [48]. It shows that the methods based on deep neural networks achieve good performance on 2.5D datasets. However, the improvements inspired by these methods on RGB/RGB-D images are not directly tailored to 3D point cloud.

1. The website for latest datasets and state-of-the-art results: https://bop.felk.cvut.cz/challenges/bop-challenge-2020/

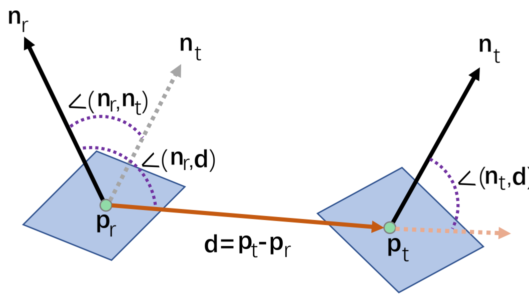

Fig. 2. Illustration of the point pair feature (Eq. 1) of two oriented points $(\mathbf{p}_r, \mathbf{p}_t)$.

3 Background and Overview

The input to our framework consists of a 3D object of interest that we call a *template*, and a *scene* represented by a point cloud that captures a partial 3D view of the real world. Each template occurrence in the scene denotes one of its *instance*, which may be a result of 3D rotation and translation of the (partial) template. Our goal is to recognize the template in the scene and determine its 6D pose with respect to the sensor. The output of our method is therefore a list of recognized instances along with their 6D poses in the scene.

Our approach is based on the basic framework of original PPF matching method [12], which is referred to *Drost-PPFM* in this paper. To make the paper self-contained, in this section, we first give a general review of the Drost-PPFM approach, then present our key ideas.

3.1 Point Pair Features based Object Detection

Point Pair Feature encodes the relative position and orientation of two oriented points. As depicted in Fig. 2, given a reference point $\mathbf{p}_r$ and a target point $\mathbf{p}_t$ with normals $\mathbf{n}_r$ and $\mathbf{n}_t$ respectively, the point pair feature is an asymmetric 4-dimensional vector, which is formally defined as:

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

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

Drost-PPFM recognizes instances and estimates their poses via two main phases: off-line global template description and on-line template matching. First, at the off-line stage, the template is usually down-sampled to a sparse set of uniformly distributed model points. Then the template description is created by computing $\mathbf{PPF}$s for all permutations of model point pairs. The resulting $\mathbf{PPF}$s are discretized and organized in a hash table where hash keys are the discretized feature vectors and the values encode the feature's pose with respect to the template. In this context, a feature's pose is achieved by storing the index of the first model point $\mathbf{p}_r$ and an angle α that can be derived from a standardized set of transformations. Actually, the stored hash table is our desired template description which will be used during the matching stage.

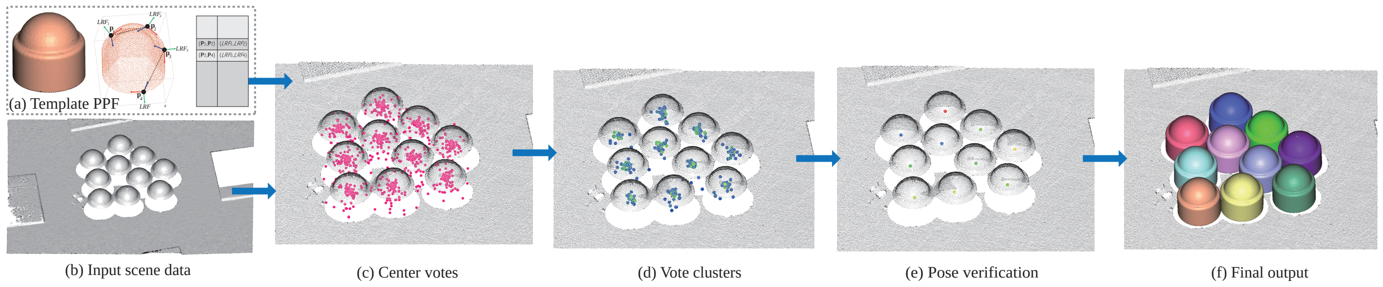

Fig. 3. Overview: (a) a global model description based on PPFs and local reference frames; (b) the input scene point cloud; (c) center votes by matching scene PPFs in the hash table; (d) top 100 clusters after pose clustering, where each cluster is color-encoded by the number of votes (warmer color indicates higher weight); (e) hypotheses verification; (f) the final estimated 6D poses shown by overlaying the aligned template.

At run-time, the uniform subsampling procedure is also applied to the scene point cloud to obtain a set of scene points, then every i-th ($i = 5$ in default) scene point is used as reference scene point. To detect each instance of the template in the scene, the scene **PPF**s paired by every reference point with all other scene points are computed and matched to the template **PPF**s by using previously built hash table. If similar features on the template are retrieved for each reference scene point, it suggests a set of possible template poses, *i.e.,* each pose is represented by a model reference point and an angle. For the voting scheme, the pose suggestions are accumulated in a Generalized *Hough Transform* manner. The maxima weighted by the votes number are extracted in the Hough space, to form the raw pose hypotheses, which are subsequently clustered. The poses in the cluster with the highest accumulated weight are averaged to return the optimal object pose. In case multiple object instances should be retrieved, the top k clusters are considered.

3.2 Our Key Improvements

Although our modeling and matching framework follows *Drost-PPFM*, we make significant adaptations in each key step to make **PPF**s more robust towards noise, occlusions and geometrically simple shapes.

As shown in Fig. 3, our algorithm consists of three core components corresponding to our main contributions. First, in the off-line phase, our global template description is represented by the template center and a hash table based on **PPF**s. However, instead of discretizing the features' poses to the bin's value, we only need to store the local reference frame (LRF) computed at each model point. Then in the matching phase, we present a novel center-targeted voting scheme. When a scene **PPF** is matched to a template **PPF**, we could build two pairs of local reference frames between the corresponding reference and target points, respectively. Two center votes are then generated by transforming the template center according to each pair of local reference frames. As a result, the votes appear near instance centers and in turn can be aggregated through a clustering module to generate pose candidates. Finally, a pose verification operator based on the concept of *Intersection over Union* (IOU) is performed to return the poses with the maximal scores.

4 Methodology

This section gives a complete and improved algorithm for 6D object pose estimation, where we describe our novel ideas and all extensions based on original *Drost-PPFM*.

4.1 Global Model Description

The input template, $\mathcal{M}$, is considered to be either a mesh data or a point cloud. We denote the template diameter, d_{obj}, as the diagonal length of its bounding box, which is relevant to most of the parameters used in our approach. The template center, $\mathbf{c}_{obj} = (x_c, y_c, z_c)$, is computed as the center point of its bounding box. Besides, we also compute two auxiliary points $\mathbf{p}^1_{aux} = (x_c - d_{obj}, y_c, z_c)$ and $\mathbf{p}^2_{aux} = (x_c, y_c - d_{obj}, z_c)$. Then with the template center, we form a point triplet $\mathcal{G}_{obj} = (\mathbf{c}_{obj}, \mathbf{p}^1_{aux}, \mathbf{p}^2_{aux})$, which will be used for pose computation in the later center voting module.

To speed up the matching computation and avoid too many non-discriminative point pair features, the template is usually re-sampled and represented by a sparse set of oriented model points, $\mathcal{P} = \{\mathbf{p}_i, \mathbf{n}_i\}^n_{i=1}$, where a normal is associated with each point coordinate. Different from *Drost-PPFM* where the template is uniformly subsampled, we adopt an adaptive sampling approach similar to [30] by taking the normal information into account. To do that, we first discretize the template by building multi-resolution voxel-grid structures with respect to the object diameter, where the sampling rate for the coarsest resolution is τ_d. Then for each voxel cell at each resolution (in a fine-to-coarse order), we merge the similar points where the angle between their normals is smaller than a threshold θ ($\theta = 20°$ in default). As a result, the feature parts (*e.g.,* sharp edges or curved surfaces) of the input template are more sufficiently sampled than planar parts, thus yielding a set of discriminative point pairs.

Next, we compute the **PPF**s between every pair of model points in $\mathcal{P}$ by calculating the feature vector of Eq. 1. The resulting **PPF**s are discretized with a distance and angle quantization step size (Δ_{dist} and Δ_{angle}), then used as indices of a hash table (more detailed explanations can be referenced in original *Drost-PPFM*). However, instead of discretizing the features' poses into the pointed table bins, we only need to store the local reference frames computed at

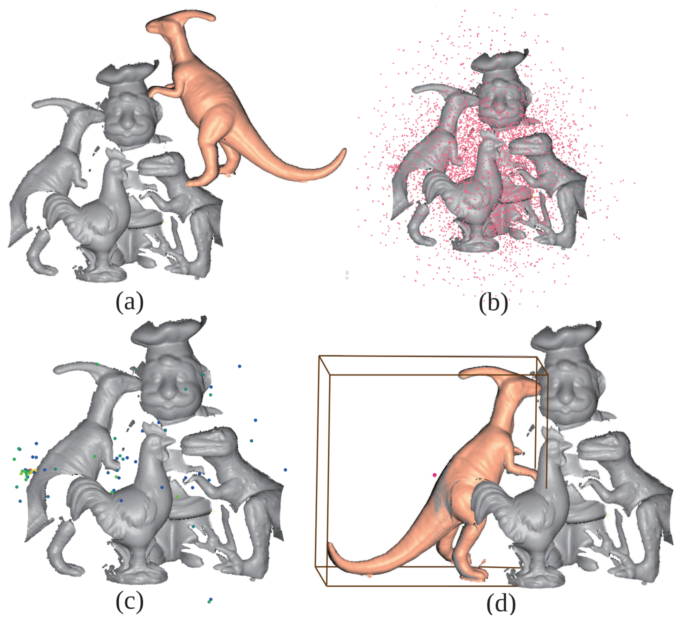

Fig. 4. An example for visualizing the center voting process: (a) the input template and scene point clouds; (b) transformed template centers; (c) top 100 clusters after pose clustering; (d) after hypotheses verification, the final estimated 6D pose is shown by overlaying the aligned template.

corresponding reference point $\mathbf{p}_r$ and target point $\mathbf{p}_t$. Their local reference frames are defined as:

$$\mathcal{O}_{\mathbf{p}_r} = [\mathbf{n}_r \times \frac{\mathbf{d}}{||\mathbf{d}||}, \mathbf{n}_r \times (\mathbf{n}_r \times \frac{\mathbf{d}}{||\mathbf{d}||}), \mathbf{n}_r],$$
$$\mathcal{O}_{\mathbf{p}_t} = [\mathbf{n}_t \times \frac{\mathbf{d}}{||\mathbf{d}||}, \mathbf{n}_t \times (\mathbf{n}_t \times \frac{\mathbf{d}}{||\mathbf{d}||}), \mathbf{n}_t], \tag{2}$$

where operator $\times$ represents the vector cross product.

4.2 Center Voting Module

Given the input scene, we subsamples it using above adaptive sampling approach to obtain a subset of scene points, denoted as $\mathcal{S}$. However, due to the objects are usually placed on a table or a platform in industrial environments, the captured scene point cloud often contains relatively large background planes. Therefore, before downsampling, we first apply an efficient RANSAC approach to extract planar primitives from the point cloud, then remove the large planes whose sizes are much bigger than the template diameter. This step could greatly reduce the number of 3D points, thus speeding up the whole matching process.

During run-time voting, a reference scene point $\mathbf{s}_r$ is paired with every other scene point $\mathbf{s}_t$ and the corresponding scene $\mathbf{PPF}$ and local reference frames ($\mathcal{O}_{\mathbf{s}_r}$ and $\mathcal{O}_{\mathbf{s}_t}$) are computed. Then we find template $\mathbf{PPF}$s for all possible model point pairs $(\mathbf{p}_r, \mathbf{p}_t)$ that generate the same quantized feature as $(\mathbf{s}_r, \mathbf{s}_t)$ by searching previously built hash table in a constant time. For each matched scene–model pair correspondence, the transformation between reference points are calculated as following:

$$\mathbf{T}_{\mathbf{p}_r \to \mathbf{s}_r} = \begin{bmatrix} \mathbf{R}_r & \mathbf{t}_r \\ 0 & 1 \end{bmatrix},$$
$$\mathbf{R}_r = \mathcal{O}_{\mathbf{s}_r}^{\mathsf{T}} \cdot \mathcal{O}_{\mathbf{p}_r}, \tag{3}$$
$$\mathbf{t}_r = \mathbf{s}_r - \mathbf{R}_r \mathbf{p}_r.$$

Here the rotation matrix $\mathbf{R}_r$ aligns the local reference frame of the template with the scene, and $\mathbf{t}_r$ translates model point $\mathbf{p}_r$ to the scene $\mathbf{s}_r$. Therefore, $\mathbf{T}_{\mathbf{p}_r \to \mathbf{s}_r}$ gives us a candidate pose. Similarly, we can obtain another transformation matrix $\mathbf{T}_{\mathbf{p}_t \to \mathbf{s}_t}$ that maps $\mathbf{p}_t$ to $\mathbf{s}_t$.

To determine whether the pair $(\mathbf{p}_r, \mathbf{p}_t)$ is a possible position of $(\mathbf{s}_r, \mathbf{s}_t)$ on the template surface, we transform the template center $\mathbf{c}_{obj}$ into the scene by using $\mathbf{T}_{\mathbf{p}_r \to \mathbf{s}_r}$ and $\mathbf{T}_{\mathbf{p}_t \to \mathbf{s}_t}$ respectively to obtain $\mathbf{c}_{obj}^r$ and $\mathbf{c}_{obj}^t$; in other words, a vote is cast for the template center. However, if the distance between $\mathbf{c}_{obj}^r$ and $\mathbf{c}_{obj}^t$ is larger than a distance threshold ζ (we set $\zeta = 0.1 d_{obj}$), the vote is discarded directly. Otherwise, we accept this vote and compute transformed triplet $\mathcal{G}_{obj}^r$ by $\mathbf{T}_{\mathbf{p}_r \to \mathbf{s}_r}$ as an attribute of the current vote. After processing all reference scene points using above center voting module, we generate multiple pose votes that appear near instance centers. Fig. 4 (b) shows an illustration of the transformed template centers.

4.3 Pose Clustering and Hypotheses Verification

Due to the noisy data and different sampling rate of the scene and template point cloud, our center voting scheme may generate isolated or incorrect votes. To remove incorrect votes and increase the matching accuracy, we propose a new clustering approach that takes into account our center voting strategy. In particular, we build a uniform voxel-grid with cell size ι_d to voxelize the points in all transformed triplets $\mathcal{G}_{obj}^r$. Therefore, each point in each triplet is mapped to an index of the grid cell. Then in each cell containing transformed template centers, we consider each pair of triplets, *e.g.*, $\mathcal{G}_{obj}^{r1}$ and $\mathcal{G}_{obj}^{r2}$ w.r.t. transformed template centers $\mathbf{c}_{obj}^{r1}$ and $\mathbf{c}_{obj}^{r2}$, respectively. If the cell indices of the three points in $\mathcal{G}_{obj}^{r1}$ equal to corresponding indices in $\mathcal{G}_{obj}^{r2}$, we group $\mathcal{G}_{obj}^{r1}$ and $\mathcal{G}_{obj}^{r2}$ into one cluster. This clustering approach enforces that similar triplets should be transformed by similar translations and rotations. Note that in our implementation, to minimize the impact of the binning problem, we also perform smoothing over neighboring cells to further help group nearby correct triplets together. After all cells are processed, we obtain multiple clusters which are ranked in decreasing order of the number of triplets (or votes), see Fig. 4 (c).

We average the transformed triplets contained in each cluster, and generate a list of pose hypotheses associated with scores by computing the transformation between the original triplet and each averaged triplet. However, we observe that the pose with the maximum score may not be optimal in case of sensor noise and background clutter. To verify each hypothesis pose, we transform the template according to the pose and the score of this pose is re-calculated by estimating the visibility of the template in the scene. For each template point $\mathbf{p}_i$, we find k nearest scene points and if there is a scene point $\mathbf{s}_j$ that satisfies $||\mathbf{p}_i - \mathbf{s}_j|| < \varepsilon_D$ and $\angle(\mathbf{n}_{\mathbf{p}_i}, \mathbf{n}_{\mathbf{s}_j}) < \varepsilon_A$ where $\varepsilon_D = 0.02 d_{obj}$ is a distance threshold and $\varepsilon_A = 25^\circ$ is an angle threshold, then we regard that $\mathbf{p}_i$ is visible in the scene and $\mathbf{s}_j$ is a fitted scene point for current pose. Finally, all of the poses are re-ordered by using the number of visible points, and we return the pose with the maximum score as the optimal one that best fits the scene points (Fig. 4 (d)).

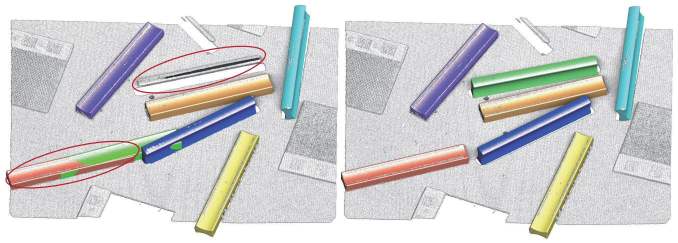
Fig. 5. Multi-instance detection. Left: there may be duplicate and missing poses after pose hypotheses verification. Right: our pose filtering based on NMS improves the recognition accuracy.

4.4 Multi-instance and Multi-object Detection

In case of detecting multiple instances of the object, an intuitive approach is to retrieve the top k poses from above re-scored pose hypotheses. However, this approach can easily lead to duplicate or missing poses, see Fig. 5 (left). In our algorithm, we perform a pose filtering step based on non-maximum suppression (NMS) which is widely used in many computer vision applications. From the list of pose hypotheses, we first return the pose with the maximum score, and then visit each other pose in decreasing order. If the Intersection over Union between the fitted scene points of current visited pose and last returned pose is larger than 0.4, we filter out current visited pose; otherwise, we accept and return it. We iteratively perform this step in the remaining poses until we retrieve k poses.

To solve multiple objects recognition task, we perform the steps of center voting, pose clustering and hypotheses verification in parallel for all 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 visibility of each object in the scene. The same pose filtering approach based on NMS is applied to reject poses that do not correspond to any object. According to our test results in next section, we found this approach could achieve better results than detecting the objects one by one, *i.e.*, segmenting out the scene data containing the instances of one object then proceeding to the next object.

5 Experimental Results

In this section, we first demonstrate the effectiveness of the proposed algorithm by evaluating different parameter settings. Then after describing the used evaluation metrics, we provide a complete comparison with state-of-the-art approaches in terms of pose accuracy and recognition rate. Their performance is thoroughly evaluated using both qualitative and quantitative results on a large number of well-known datasets. Our algorithm is implemented in C++. All results presented in this paper are obtained on a desktop computer equipped with an Intel i7-7700k processor clocked at 4.2GHz Ghz, 16 GB of RAM.

5.1 Experimental Setup

Datasets. For our performance evaluation and comparisons, we carry out experiments on the public-domain datasets, which are selected based on different criteria, *e.g.*, quality of the data, variety of objects, complexity of scenes and acquisition technique. In particular, we evaluate the algorithms on three 3D point cloud data which are most common used for 3D object recognition. The UWA [49] contains 4 complete object models and 50 view based scenes, which are all laser scanned and represented as high resolution meshes. The objects are highly occluded in each scene and the number of instances to recognize is 188 in total. Toshiba CAD [50] provides a challenging and realistic 3D dataset with a vision-based geometry capture system. The dataset consists of 12 shape classes which are fabricated from CAD models, with and without rotational symmetries. The DTU [51] is a large-scale dataset consisting of 45 objects and 3204 scenes captured by a structured light scanner, where each scene contains 10 objects. The objects belong to three different types: geometric complex objects, cylindrical and flat 3D object models. We focus on the last two types because they are more challenging for pose estimation.

Evaluation metric. To determinate the pose accuracy, several pose error functions are proposed [52] to compute the error of an estimated 6D object pose $\hat{\mathbf{T}}$ w.r.t the ground-truth pose $\bar{\mathbf{T}}$. A widely used pose error function is *average distance metric* (ADM) which measures the L_2 distance between the model points transformed by the estimated pose and ground-truth pose respectively. In [52], two alternatives (e_{ADD} and e_{ADI}) of ADM are defined for objects without and with symmetry properties:

$$\begin{aligned} e_{ADD} &= \operatorname*{avg}_{\mathbf{x}\in\mathcal{M}} ||\bar{\mathbf{T}}\mathbf{x} - \hat{\mathbf{T}}\mathbf{x}||_2, \\ e_{ADI} &= \operatorname*{avg}_{\mathbf{x_1}\in\mathcal{M}} \min_{\mathbf{x_2}\in\mathcal{M}} ||\bar{\mathbf{T}}\mathbf{x_1} - \hat{\mathbf{T}}\mathbf{x_2}||_2. \end{aligned} \tag{4}$$

However, since e_{ADI} yields relatively small errors , values of e_{ADD} and e_{ADI} should not be directly compared [52]. To solve this problem, we define a new e_{ADI} as:

$$e^{'}_{ADI} = \max(\operatorname*{avg}_{\mathbf{x_1}\in\mathcal{M}} \min_{\mathbf{x_2}\in\mathcal{M}} ||\bar{\mathbf{T}}\mathbf{x_1} - \hat{\mathbf{T}}\mathbf{x_2}||_2, ||\bar{\mathbf{T}}\mathbf{c}_{obj} - \hat{\mathbf{T}}\mathbf{c}_{obj}||_2), \tag{5}$$

where we also consider the distance between transformed object centers $\mathbf{c}_{obj}$.

After defining the pose error function, we accept a pose estimate 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 true positive poses compared to all retrieved 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:

$$MR = \operatorname*{avg}_{\mathbf{o}\in O} \frac{\sum_{s\in S}|P(o,s)|}{\sum_{s\in S}|G(o,s)|}, \tag{6}$$

where O and S are the sets of all 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.

Description of competing algorithms. We thoroughly compare our method against most competitive methods. On 3D point cloud data, we select a commercial machine vision software MVTec HALCON[2] as a competitor because it contains the optimized and improved implementation of the original Drost-PPFM. We denote it Drost-PPFM*.

2. https://www.mvtec.com/products/halcon/

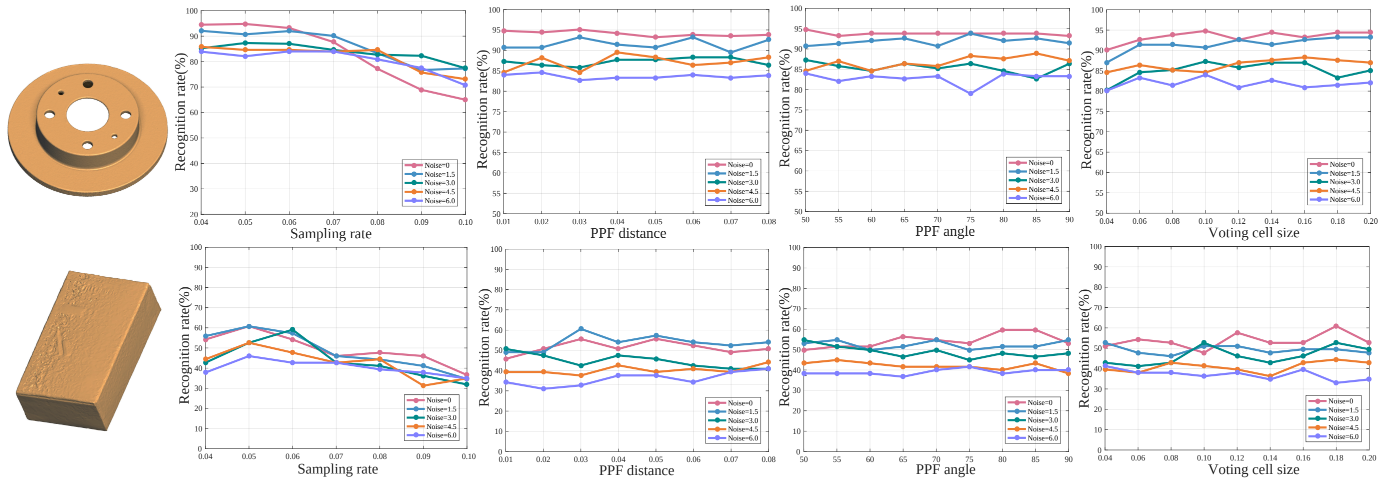

Fig. 6. Parameter analysis using two models from DTU dataset [51]. For each parameter setting, we also show the performance of our approach reacting with data at different noise levels.

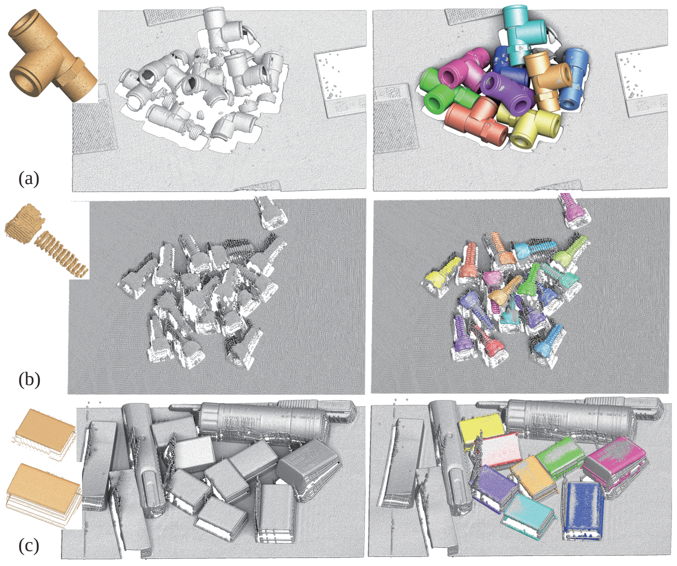

Fig. 7. Recognition results for multi-instances and multi-objects. The number of instances is 9 (a), 17 (b) and 8 (c), respectively.

In addition, we compare to an open-source method Buch-17 [28], which presents a new pose voting and clustering method by integrating a local feature-based recognition pipeline. Here we test several representative 3D local feature descriptors, including PPF, spin images (SI) [26], fast point feature histograms (FPFH) [53] and signature of histogram of orientations (SHOT) [54].

5.2 Evaluation

Parameter settings. We first run different detection tests with varied parameters to evaluate their influence on the recognition rate. While the majority of parameters can be set to default, we mainly analyze the following four parameters: the sampling rate τ_d for downsampling template and scene point clouds, the quantization step size of distance Δ_{dist} and angle Δ_{angle} for computing quantized PPFs, and the cell size ι_d of the voxel-grid for clustering center votes. To be independent from the template size, the values of τ_d, Δ_{dist} and ι_d are relative to the template diameter d_{obj}. Fig. 6 shows the ablation studies for those four parameters, *i.e.*, we vary one parameter while fixing all other parameters. We can observe that our algorithm is robust to the values of Δ_{dist}, Δ_{angle} and ι_d. The parameter τ_d affects our performance, where we see an obvious performance drop when it is larger than 0.06. We find that setting $\tau_d = 0.05$ achieves the best performance for most of the models.

Resistance against noise. The robustness of our method is also demonstrated in Fig. 6. To generate noisy scene point clouds, we randomly add Gaussian noise to the point coordinates with different values of standard deviation. As shown in the figures, our algorithm has a good resilience against noise. The performance is slightly reduced as the level of noise increases, but we still perform well on noisy data.

Multi-instance and multi-object detection. Figs. 1 and 7 verify the use of our method for solving multi-instance and multi-object recognition problems. In Fig. 1 (a) and Fig. 7 (a) we use a full object model as the template, while in other examples the template is selected as a partial part of the object. For both cases, our method detects all of the instances and returns their correct 6D poses.

Algorithm inspection. To better understand our algorithm, we analyze the behavior of our center voting strategy. Taking the scene in Fig. 3 as an example, given the clustering voxel-grid with cell size ι_d, if one center vote is located in a cell that contains a ground-truth center, this center vote is judged as an inlier. The ratio of inliers in Fig. 3 (c) is termed as r_{inlier}. However, the inlier center vote may be cast by reference point pairs that are not on the object instances. Therefore, we count how many valid inliers are cast by the point pairs that are located on the instances, and the ratio of valid inliers among all inliers is termed as r_{inlier}^{valid}. The ratio of inliers after pose clustering (Fig. 3 (d)) and verification (Fig. 3 (e)) are r_{inlier}^{c} and r_{inlier}^{v}, respectively. Note that we cannot count the number of valid inliers in Fig. 3 (d) and (e) because the clustering process has grouped the valid and invalid inliers together.

TABLE 1
Statistic analysis of our center voting strategy and triplets verification.

Scene	ι_d	$r_{inlier}(\%)$	$r^{cluster}_{inlier}(\%)$	$r^{verif}_{inlier}(\%)$	$r^{valid}_{inlier}(\%)$	$r^{obj}_{inlier}(\%)$	$Dist$	$r_{trip}(\%)$
Fig. 3	0.20	50.55	84	100	95.66	59.89	7.8	×
	0.18	40.44	68	100	96.69	46.83	7.6	×
	0.16	39.23	78	100	97.18	45.06	7.4	×
	0.14	36.65	70	100	97.40	41.87	7.2	×
	0.12	33.15	70	100	99.19	37.99	7.0	×
	0.10	30.16	38	100	99.29	34.10	4.3	×
	0.08	25.66	35	100	99.50	28.94	6.3	×
	0.06	17.22	21	90	96.73	19.16	6.9	×
	0.04	15.38	10.29	80	100	15.38	6.5	×
Fig. 4	0.20	1.06	2	100	78.07	16.99	25.2	8.39
	0.18	1.63	4	100	57.07	25.09	24.5	8.97
	0.16	1.80	2	100	54.95	27.29	24.4	8.82
	0.14	1.44	2	100	58.31	20.19	23.6	9.97
	0.12	0.92	4	100	84.86	12.55	22.6	10.05
	0.10	1.31	10	100	95.83	17.25	21.5	14.85
	0.08	1.06	8	100	93.47	13.31	21.1	12.45
	0.06	1.12	16	100	89.74	13.40	20.1	15.46
	0.04	0.81	8	100	100	8.75	21.1	11.25

TABLE 2
Quantitative comparison on UWA dataset. The best result of each measurement is marked in **bold** font. ξ_e is the pose error threshold, RR is per-object recognition rates, and MR represents the overall performance on this dataset.

Methods	ξ_e	RR_{cheff}	$RR_{chicken}$	RR_{para}	RR_{Trex}	MR
Buch-17-PPF	$0.1d_{obj}$	**100**	**100**	97.8	97.8	98.95
	$0.2d_{obj}$	**100**	**100**	97.8	97.8	98.95
	$0.3d_{obj}$	**100**	**100**	97.8	97.8	98.95
	$Time(s)$	4.36	2.53	3.41	3.21	3.39
Buch-17-SHOT	$0.1d_{obj}$	96	85.4	75.6	84.4	85.63
	$0.2d_{obj}$	96	85.4	75.6	84.4	85.63
	$0.3d_{obj}$	96	85.4	77.8	84.4	86.16
	$Time(s)$	4.61	2.95	5.63	5.37	4.61
Buch-17-SI	$0.1d_{obj}$	**100**	**100**	97.8	**100**	99.47
	$0.2d_{obj}$	**100**	**100**	97.8	**100**	99.47
	$0.3d_{obj}$	**100**	**100**	97.8	**100**	99.47
	$Time(s)$	3.26	2.56	3.85	2.91	3.14
Buch-17-FPFH	$0.1d_{obj}$	**100**	97.9	95.6	97.8	97.88
	$0.2d_{obj}$	**100**	97.9	95.6	97.8	97.88
	$0.3d_{obj}$	**100**	97.9	95.6	97.8	97.88
	$Time(s)$	12.85	10.32	23.9	22.9	17.25
Drost-PPFM*	$0.1d_{obj}$	98	93.8	91.1	97.8	95.23
	$0.2d_{obj}$	98	93.8	91.1	97.8	95.23
	$0.3d_{obj}$	98	93.8	91.1	97.8	95.23
	$Time(s)$	**0.98**	**0.53**	**0.31**	**0.41**	**0.57**
Ours	$0.1d_{obj}$	**100**	**100**	**100**	**100**	**100**
	$0.2d_{obj}$	**100**	**100**	**100**	**100**	**100**
	$0.3d_{obj}$	**100**	**100**	**100**	**100**	**100**
	$Time(s)$	3.65	3.65	3.65	3.65	3.65

Next, to analyze the spread of the center votes, we compute two indicators from Fig. 3 (c): the first one is r^{obj}_{inlier}, which shows how many point pairs that are on the instances vote for the correct center; the second one is the average distance $Dist$ from the valid inliers to the corresponding ground-truth centers, and this $Dist$ reveals the approximate deviation of the valid votes from the correct centers.

These numerical statistics are reported in Table 1. In order to avoid interference between different instances, we also report the analysis on Fig. 4 in which there is only one instance but with complex background. From the analysis, we see that the ratio of inliers in Fig. 3 is much greater than those of Fig. 4. The reason is that the background of Fig. 3 is relatively simple, and there is no interference from other objects. The r^{obj}_{inlier} and deviation $Dist$ of the valid votes also shows the same phenomenon. Besides, from deviation $Dist$ we can observe that our voting module really generates votes near the object centers.

Finally, we compute another indicator (r_{trip}) to inspect how many triplets that are from object instances are really located in the same bin containing the ground-truth triplet. The results are also reported in Table 1. Note we can only analyze r_{trip} in Fig. 4 because the ground-truth triplet is not unique for the symmetric object in Fig. 3.

TABLE 3
Quantitative comparison on Toshiba CAD dataset. The best result of each measurement is marked in **bold** font.

Methods	ξ_e	RR_1	RR_2	RR_3	RR_4	RR_5	RR_6	RR_7	RR_8	RR_9	RR_{10}	MR
Buch-17-PPF	$0.1d_{obj}$	5	5	5	90	45	20	0	50	0	15	23.50
	$0.2d_{obj}$	5	25	10	90	55	40	0	50	5	25	30.50
	$0.3d_{obj}$	15	50	10	100	55	50	0	50	12	40	38.20
	$Time(s)$	10.90	4.18	34.50	9.10	3.62	3.84	6.87	4.62	6.83	7.43	9.19
Buch-17-SHOT	$0.1d_{obj}$	0	15	0	10	50	20	0	20	5	25	14.50
	$0.2d_{obj}$	5	55	5	45	55	20	0	20	10	30	24.50
	$0.3d_{obj}$	5	75	5	45	65	20	0	25	45	90	37.50
	$Time(s)$	15.45	2.50	21.00	15.50	7.38	7.40	13.35	14.78	4.41	4.80	10.66
Buch-17-SI	$0.1d_{obj}$	20	25	50	95	95	60	0	60	0	40	44.50
	$0.2d_{obj}$	20	50	50	95	95	70	0	60	0	40	48.00
	$0.3d_{obj}$	30	55	50	95	95	70	0	65	20	70	55.00
	$Time(s)$	10.70	1.69	12.10	4.92	3.60	6.02	6.00	4.46	3.72	3.84	5.70
Buch-17-FPFH	$0.1d_{obj}$	5	25	5	85	25	15	0	35	0	25	22.00
	$0.2d_{obj}$	10	40	5	95	25	20	0	35	5	30	26.50
	$0.3d_{obj}$	10	50	10	95	30	35	0	35	25	45	33.50
	$Time(s)$	35.00	14.32	58.67	30.52	20.61	36.33	30.23	45.83	42.00	36.33	34.98
Drost-PPFM*	$0.1d_{obj}$	**95**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	45	95	93.50
	$0.2d_{obj}$	**95**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	75	95	96.50
	$0.3d_{obj}$	**95**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	95	**100**	99.00
	$Time(s)$	**0.58**	**0.51**	**3.2**	2.16	**0.23**	1	**1.58**	1.6	**1.53**	**1.25**	**1.36**
Ours	$0.1d_{obj}$	**95**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**80**	**100**	**97.50**
	$0.2d_{obj}$	**95**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**90**	**100**	**98.50**
	$0.3d_{obj}$	**95**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**99.50**
	$Time(s)$	9.44	2.45	4.71	**1.4**	3.33	**0.81**	3.07	**0.46**	9.14	5.25	4.01

5.3 Comparisons

Comparison on 3D point cloud datasets. Tables 2, 3 and 4 show an extensive comparison of performance scores and timings on datasets of UWA, Toshiba CAD and DTU, respectively. Recognition rate per object and per dataset are reported in each table. Furthermore, for fair comparison, we set three levels (10%, 20%, 30% w.r.t the object diameter d_{obj}) of the pose error threshold ξ_e to evaluate each algorithm sufficiently.

The quantitative comparisons verify that our algorithm achieves best-performing results on these datasets. Specifically, since UWA dataset contains 3D models that possesses rich shape variations, all of the test algorithms obtain good matching accuracy for this dataset, where we achieve 100% recognition rates for all objects even with a high degree of occlusion. On Toshiba CAD dataset which is with limited shape features, methods based on the framework of Buch-17 [28] perform worst because they rely on finding local feature correspondences thus are very little descriptive for planar and constant curvature surfaces. On the other hand, Drost-PPFM* and our approach still obtain good performance scores on this dataset, and ours is slightly better. DTU dataset is more challenging because it has distinct levels of complexity with high occlusion and clutter. We avoid testing feature-rich objects and select repetitive, flat, cylindrical and thin-edge objects without many local features. As expected, we observe a significant performance drop for each method. However, our algorithm still outperforms other competitors for most of the test objects. The main reason of our improvement over other PPF-based methods is that we conducted pose clustering on the transformed object centers instead of the pose matrix, which is sensitive to the values of rotation and translation elements. Fig. 8 shows qualitative comparison results on the DTU dataset.

Finally, in terms of running time, HALCON/Drost-PPFM*, as a commercial software, takes advantage of hardware and it is the fastest. Our method is considerably faster comparing to Buch-17. Moreover, each step of our method can be parallelized thus in future we could implement it on GPU for further acceleration.

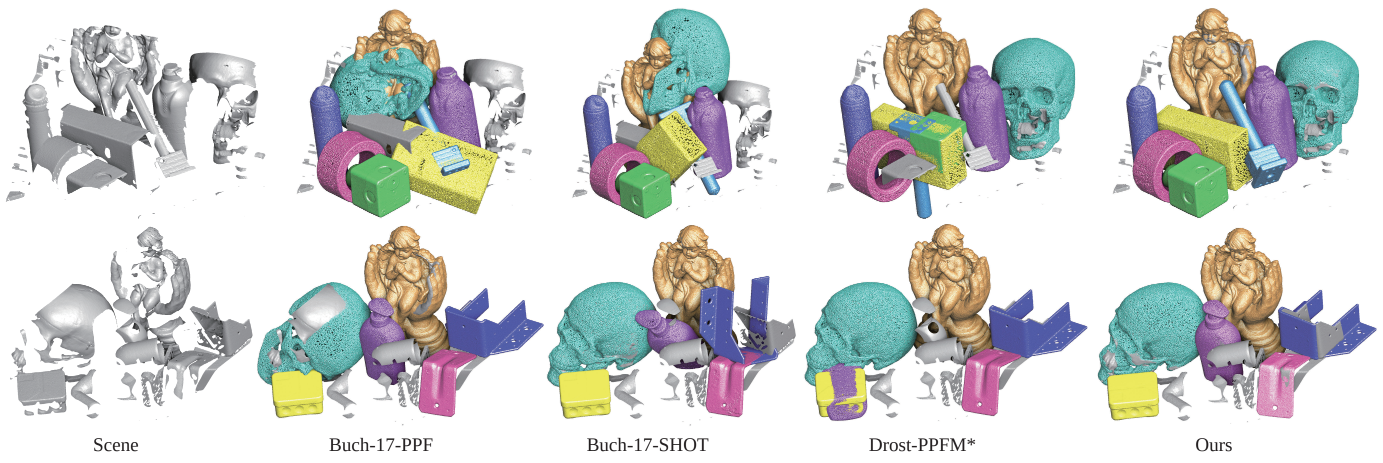

Fig. 8. Qualitative comparison results on two scenes from DTU dataset.

TABLE 4
Quantitative comparison on DTU dataset. The best result of each measurement is marked in **bold** font.

Methods	ξ_e	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}	RR_{48}	RR_{50}	MR
Buch-17-PPF	$0.1d_{obj}$	67.1	71.2	62.4	59.4	11.7	48.7	**73.3**	14.7	85	15.3	19.3	**76.8**	62.4	29.6	23.1	7.1	55.26
	$0.2d_{obj}$	**75**	72.3	76.3	64.1	15.9	49.1	**74**	42.6	**89.7**	24.5	20.5	**78**	63.4	33	46.1	7.1	60.41
	$0.3d_{obj}$	**77**	73.4	82.8	72.7	20.2	49.1	**74**	62.3	**93.1**	27.8	21.6	**78.7**	65.5	38.9	46.1	7.1	63.82
	$Time(s)$	2.76	3.44	1.15	1.2	1.36	1.45	1.52	1.68	1.15	1.07	1.46	1.41	**1.36**	3.52	1.04	0.52	1.70
Buch-17-SHOT	$0.1d_{obj}$	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	42	15.4	0	36.45
	$0.2d_{obj}$	55.7	54.9	39.2	44.5	15.9	38.2	37.7	29.5	61.1	26.2	42	51.6	36.5	45.1	15.4	14.3	43.11
	$0.3d_{obj}$	60.7	54.9	52.3	48.4	15.9	39.5	39	52.5	70.4	30.1	45.5	52.3	41.1	47.5	30.8	14.3	48.16
	$Time(s)$	5.71	7.36	1.45	2.86	1.67	3.02	1.65	2.46	2.08	4.88	3.48	3.31	2.89	4.49	1.43	1.39	3.32
Buch-17-SI	$0.1d_{obj}$	64.3	**72.3**	46.5	48.4	3.2	39.9	62.3	21.3	75.4	18	45.5	61.9	61.4	43.2	0	0	50.82
	$0.2d_{obj}$	69.3	**76.1**	61.2	57	6.4	40.4	64.4	42.6	85.7	25.7	47.7	66.5	63.5	46.3	7.7	0	57.07
	$0.3d_{obj}$	75	**78.9**	73.8	63.3	7.4	48.8	65.1	59	91.6	28.4	47.7	67.1	66	50	15.4	0	62.16
	$Time(s)$	2.36	2.96	0.91	1.14	1.28	1.23	1.13	0.67	1.69	1.01	0.81	**1.33**	1.77	1.72	1.03	0.58	1.49
Buch-17-FPFH	$0.1d_{obj}$	35	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	0	0	42.60
	$0.2d_{obj}$	47.1	54.3	52.2	44.5	18.1	48.7	63	36.1	79.1	25.1	53.4	54.8	43.7	51.2	0	0	50.60
	$0.3d_{obj}$	52.9	56.5	66.1	55.5	19.1	48.7	63	50.8	85.4	30.6	55.7	56.8	46.2	56.7	30.8	0	55.83
	$Time(s)$	34.2	49.1	8.47	13.9	8.76	13.9	12.64	9.54	8.4	19.8	13.6	21.5	19.87	77.6	8.79	8.76	21.90
Drost-PPFM*	$0.1d_{obj}$	67.1	69	57.9	**68**	77.7	63.5	62.3	42.6	82.8	**55.8**	29.5	54.8	66	86.4	15.4	0	65.08
	$0.2d_{obj}$	69.3	69	64.1	**70.3**	77.7	64.5	62.3	49.2	85.6	**63.4**	29.5	54.8	66.5	86.4	23.1	0	67.31
	$0.3d_{obj}$	70.7	69	66.1	**70.3**	77.7	64.9	63	50.8	87.9	**66.7**	31.8	54.8	69.5	88.2	46.2	0	68.88
	$Time(s)$	**0.19**	**0.28**	**0.06**	**0.08**	**0.02**	**0.04**	**0.05**	**0.04**	**0.024**	**0.03**	**0.47**	1.66	2.46	**0.03**	**0.04**	**0.03**	**0.39**
Ours	$0.1d_{obj}$	**67.8**	61.7	**64.9**	58.6	**81.9**	**68**	63	**55.7**	**85.4**	54.6	**68.2**	69	**68**	**95.7**	**46.2**	**21.4**	**69.29**
	$0.2d_{obj}$	72.9	63.4	**74.7**	60.2	**83**	**68**	64.4	**60.7**	88.2	59.5	**69.3**	69	**68.5**	**96.3**	**46.2**	**21.4**	**71.88**
	$0.3d_{obj}$	75	64.5	**76.7**	60.2	**83**	**68**	65.1	**63.9**	90.3	61.2	**69.3**	69.7	**70.6**	**96.3**	**53.8**	**21.4**	**73.24**
	$Time(s)$	11	17.32	1.36	11.8	10.1	30.5	15.25	6.79	0.85	10.5	6.51	8.4	12.6	0.956	8.85	0.11	10.15

TABLE 5
Comparison of average recall scores on several RGB-D datasets. The performances of other methods are reported in BOP challenge 2020.

Methods	CNN	Image	T-LESS	IC-BIN	LM-O	ITODD
CosyPose-Synt+Real-ICP [41]	Yes	RGB-D	0.701	**0.647**	**0.714**	0.313
Hybrid-DL-PointPairs [44]	Yes	RGB-D	0.655	0.430	0.631	0.483
CosyPose-Synt+Real [41]	Yes	RGB	**0.728**	0.583	0.633	0.216
Pix2Pose-BOP20-w/ICP-ICCV19 [42]	Yes	RGB-D	0.512	0.390	0.588	0.351
CosyPose-PBR-1View [41]	Yes	RGB	0.640	0.583	0.633	0.216
Vidal-Sensors18. [30]	No	D	0.538	0.393	0.582	0.435
CDPNv2BOP20-RGB-ICP [43]	Yes	RGB-D	0.464	0.450	0.630	0.186
Drost-CVPR10 [12]	No	D	0.444	0.388	0.527	0.316
Félix&Neves-ICRA17-IET19 [55], [56]	Yes	RGB-D	0.212	0.323	0.394	0.069
Sundermeyer-IJCV19+ICP [57]	Yes	RGB-D	0.487	0.281	0.237	0.158
Zhigang-CDPN-ICCV19 [43]	Yes	RGB	0.124	0.257	0.374	0.070
Sundermeyer-IJCV19 [57]	Yes	RGB	0.304	0.217	0.146	0.101
Pix2Pose-BOP-ICCV19 [42]	Yes	RGB	0.275	0.215	0.077	0.032
DPOD (synthetic) [58]	Yes	RGB	0.081	0.130	0.169	0.000
Ours	No	D	0.341	0.294	0.182	**0.519**

Comparison on RGB-D datasets. In addition, although our algorithm is not tailored to 2.5D recognition, we also provide a comparison on recent RGB-D datasets. To this end, we evaluate our method using the evaluation protocol proposed in BOP challenge and compare various advanced approaches therein. In this challenge, the task needed to solve is 6D localization of a varying number of instances of a varying number of objects in a single RGB-D image. Note the data captured by consumer RGB-D cameras usually has a low and heavily inhomogeneous point density, and the used datasets in BOP challenge include challenging test cases (household and industry-relevant objects) with a high amount of clutter and occlusion. Due to this, our hypotheses verification based on 3D IOU usually generates many false positive poses which greatly reduce the performance. For such RGB-D datasets, we implemented a view-dependent re-scoring measure to filter out incorrect poses by checking model-scene data consistency and silhouette matching [20], [30]. The rest of our pipeline is kept unchanged and we strictly follow the evaluation protocol described in BOP challenge.

The comparisons against 14 previous approaches are reported in Table 5. We observe that most of top-performing methods are based on deep neural networks and make full use of RGB image channels to achieve good performance. Note that even for one algorithm, its performance can be different by integrating different ingredients, such as depth-

TABLE 6
Ablation study of the pose clustering baseline based on a simple angle comparison, where $PCBA^a$ represents the baseline with angle threshold a, RR is per-object recognition rate using the DTU model ($\#16$ or $\#41$ in Fig. 6) and with different number of scene reference points.

Methods	ξ_e	$RR^{16}_{2.5\%}$	$RR^{16}_{5\%}$	$RR^{16}_{10\%}$	$RR^{16}_{20\%}$	$RR^{41}_{2.5\%}$	$RR^{41}_{5\%}$	$RR^{41}_{10\%}$	$RR^{41}_{20\%}$
$PCBA^{2.5°}$	$0.1d_{obj}$	33.7	**39.3**	37.7	44.3	88.9	89.5	90.7	89.5
	$0.2d_{obj}$	**44.2**	47.8	44.3	45.9	90.7	90.1	92.6	90.7
	$0.3d_{obj}$	**54.1**	**54.1**	49.2	50.8	91.4	90.7	92.6	90.7
$PCBA^{5°}$	$0.1d_{obj}$	29.5	29.5	34.4	42.6	87.6	86.4	89.5	90.7
	$0.2d_{obj}$	39.3	45.9	47.5	49.2	88.9	87.6	91.4	92
	$0.3d_{obj}$	44.3	52.5	**57.4**	57.4	89.5	88.9	91.4	92.6
$PCBA^{10°}$	$0.1d_{obj}$	31.1	27.9	34.4	37.7	88.3	88.3	91.4	91.4
	$0.2d_{obj}$	39.3	42.6	49.2	47.5	90.1	90.7	93.2	92
	$0.3d_{obj}$	45.9	47.5	**57.4**	57.4	90.7	92	93.2	92
$PCBA^{15°}$	$0.1d_{obj}$	31.1	32.8	32.3	41	87.2	90.1	91.4	90.1
	$0.2d_{obj}$	42.6	39.3	47.5	52.5	89.2	91.4	92	91.4
	$0.3d_{obj}$	45.4	49.2	50.8	55.7	90.3	92.6	92.6	92
Ours	$0.1d_{obj}$	**36.1**	42.6	**45.4**	**55.7**	**92.6**	**93.8**	**94.4**	**95.7**
	$0.2d_{obj}$	42.6	**49.2**	**54.1**	**60.7**	**93.2**	**93.8**	**94.4**	**96.3**
	$0.3d_{obj}$	52.5	52.5	**57.4**	**63.9**	**93.4**	**94.6**	**94.2**	**96.8**

based ICP refinement and data augmentation. Although we do not yet perform perfectly on all those 2.5D datasets, we achieve results comparable to some recent deep-learning-based methods in terms of average recall by only using the depth information. We would like to further increase our accuracy by considering RGB information in future work.

5.4 Ablation studies

Pose clustering. In our approach, we use the triplets voting and identification for pose clustering. A more intuitive baseline is to directly compare the rotation of poses using rotation angle and cluster poses by setting a distance threshold for the angle. We implement such a pose clustering baseline based on a simple angle comparison, which is abbreviated as *PCBA*. We compute the rotation angle using the method described in [59], [60]:

$$2cos|\alpha| = trace(R_1^{-1}R_2), \tag{7}$$

where the absolute orientation error $|\alpha|$ measures the minimum rotation angle required to align the rotations (R_1 and R_2) of two poses. However, it is very slow to perform all pairwise pose comparison (it takes 4-5 minutes for matching one scene). Therefore, we speed it up by using our voxelization, *i.e.*, we only compare the poses located in the same grid cell. Table 6 shows the comparison results using the DTU models shown in Fig. 6, where we define four pose accuracy intervals by varying the angle thresholds: $2.5°, 5°, 10°, 15°$. We also select different number of scene points as references for fair and thorough comparison. From the table, we can see that the baseline method has a good effect, but our method still achieves better accuracy in general.

Pose hypotheses verification. To evaluate the effect of our pose hypotheses verification, we perform an ablation study by using the algorithm of *Iterative Closest Point* (ICP). After we cluster poses in Sec. 4.3, we transform the template into the scene according to the poses, and perform ICP to refine the poses. Then in order to compute the score of each pose, we implement two variants: (1) using the number of triplet votes in each cluster as the score, which is indicated as *ICP+voting*; (2) using the ICP distance between the refined template and scene as the score (a smaller distance means a better pose), which is indicated as *ICP+distance*. As observed from the quantitative comparison in Table 7, our pose hypotheses verification based on the template visibility is clearly better than these two variants.

TABLE 7
Ablation study of two variants of pose hypotheses verification based on ICP+Scoring, where we also test them on the DTU model ($\#16$ or $\#41$) used in Fig. 6.

Methods	ξ_e	$RR^{16}_{2.5\%}$	$RR^{16}_{5\%}$	$RR^{16}_{10\%}$	$RR^{16}_{20\%}$	$RR^{41}_{2.5\%}$	$RR^{41}_{5\%}$	$RR^{41}_{10\%}$	$RR^{41}_{20\%}$
ICP+voting	$0.1d_{obj}$	14.8	23	16.4	32.8	62.9	59.3	76.5	75.3
	$0.2d_{obj}$	26.2	36.1	37.7	47.5	77.2	79.6	81.5	82.1
	$0.3d_{obj}$	37.7	44.3	54.1	57.4	80.9	81.5	84	82.7
ICP+distance	$0.1d_{obj}$	1.6	3.3	4.91	6.56	58	56.2	56.2	58
	$0.2d_{obj}$	8.2	9.8	13.1	14.8	59.9	57.4	59.3	60.5
	$0.3d_{obj}$	16.4	9.8	14.8	21.3	65.4	61.1	61.7	64.2
Ours	$0.1d_{obj}$	**36.1**	**42.6**	**45.4**	**55.7**	**92.6**	**93.8**	**94.4**	**95.7**
	$0.2d_{obj}$	**42.6**	**49.2**	**54.1**	**60.7**	**93.2**	**93.8**	**94.4**	**96.3**
	$0.3d_{obj}$	**52.5**	**52.5**	**57.4**	**63.9**	**93.4**	**94.6**	**94.2**	**96.8**

5.5 Limitations

An inherent shortcoming of the PPF based approaches is that it relies on the surface normals. In our method we use a traditional method to estimate the point normals, thus our matching performance is affected by the accuracy of normal estimation to a large extent, especially for the RGB-D datasets. We would like to integrate more robust approach for estimating surface normals from noisy point sets (*e.g.*, PCPNet [61]). Another limitation is that we cannot handle object scaling because the PPF relys on the Euclidean distance between the point pair.

6 Conclusion and Future Work

We have presented a new approach for 6D pose estimation of free-form objects in 3D point cloud. We achieve major improvements by modifying the whole pipeline of original point pair feature matching. A new center voting scheme associated with pose clustering and hypotheses verification operations enables us to find correct poses. We demonstrate the advantages of our approach by comparing to the state-of-the-art methods on various challenging benchmark datasets.

Future work would be to explore new point pair feature that does not rely on estimated surface normals. A possible direction may be investigating deep neural networks to automatically learn such features. Besides, we are also working on further improving our performance on RGB-D datasets by enhancing depth quality. Combining semantic image information (*e.g.*, object detection or segmentation based on deep neural networks) and 3D point pair features would also be an intriguing problem to explore.

References

[1] C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka, "Rigid 3d geometry matching for grasping of known objects in cluttered scenes," *The International Journal of Robotics Research*, vol. 31, no. 4, pp. 538–553, 2012.
[2] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1907–1915.
[3] E. Marchand, H. Uchiyama, and F. Spindler, "Pose estimation for augmented reality: a hands-on survey," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 22, no. 12, pp. 2633–2651, 2015.
[4] E. Marder-Eppstein, "Project tango," in *ACM SIGGRAPH 2016 Real-Time Live!*, 2016, pp. 25–25.

[5] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. X. Chang, and M. Nießner, "Scan2cad: Learning cad model alignment in rgb-d scans," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2614–2623.

[6] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," in *European Conference on Computer Vision (ECCV)*, 2004, pp. 224–237.

[7] A. G. Buch, H. G. Petersen, and N. Krüger, "Local shape feature fusion for improved matching, pose estimation and 3d object recognition," *SpringerPlus*, vol. 5, no. 1, p. 297, 2016.

[8] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3109–3118.

[9] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2938–2946.

[10] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, "Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 205–220.

[11] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3343–3352.

[12] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3d object recognition," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 998–1005.

[13] L. Kiforenko, B. Drost, F. Tombari, N. Krüger, and A. G. Buch, "A performance evaluation of point pair features," *Computer Vision and Image Understanding*, vol. 166, pp. 66–80, 2018.

[14] E. Kim and G. Medioni, "3d object recognition in range images using visibility context," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 3800–3807.

[15] B. Drost and S. Ilic, "3d object detection and localization using multimodal point pair features," in *International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, 2012, pp. 9–16.

[16] C. Choi and H. I. Christensen, "3d pose estimation of daily objects using an rgb-d camera," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 3342–3349.

[17] C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam, "Voting-based pose estimation for robotic assembly using a 3d sensor," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 1724–1731.

[18] O. Tuzel, M.-Y. Liu, Y. Taguchi, and A. Raghunathan, "Learning to rank 3d features," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 520–535.

[19] T. Birdal and S. Ilic, "Point pair features based object detection and pose estimation revisited," in *International Conference on 3D Vision (3DV)*, 2015, pp. 527–535.

[20] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige, "Going further with point pair features," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 834–848.

[21] A. S. Mian, M. Bennamoun, and R. A. Owens, "Automatic correspondence for 3d modeling: an extensive review," *International Journal of Shape Modeling*, vol. 11, no. 02, pp. 253–291, 2005.

[22] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas, "Bop challenge 2020 on 6d object localization," in *European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 577–594.

[23] C. Sahin, G. Garcia-Hernando, J. Sock, and T.-K. Kim, "A review on object pose recovery: from 3d bounding box detectors to full 6d pose estimators," *arXiv preprint arXiv:2001.10609*, 2020.

[24] F. Stein and G. Medioni, "Structural indexing: Efficient 3-d object recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 2, pp. 125–145, 1992.

[25] C. S. Chua and R. Jarvis, "Point signatures: A new representation for 3d object recognition," *Int. Journal of Computer Vision*, vol. 25, no. 1, pp. 63–85, 1997.

[26] 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, 1999.

[27] B. Taati and M. Greenspan, "Local shape descriptor selection for object recognition in range data," *Computer Vision and Image Understanding*, vol. 115, no. 5, pp. 681–694, 2011.

[28] A. G. Buch, L. Kiforenko, and D. Kraft, "Rotational subgroup voting and pose clustering for robust 3d object recognition," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4137–4145.

[29] J. Vidal, C.-Y. Lin, and R. Martí, "6d pose estimation using an improved method based on point pair features," in *international conference on control, automation and robotics (ICCAR)*, 2018, pp. 405–409.

[30] 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, 2018.

[31] R. Vock, A. Dieckmann, S. Ochmann, and R. Klein, "Fast template matching and pose estimation in 3d point clouds," *Computers & Graphics*, vol. 79, pp. 36–45, 2019.

[32] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 876–888, 2011.

[33] 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 *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 858–865.

[34] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4421–4428.

[35] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6d object pose estimation using 3d object coordinates," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 536–551.

[36] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim, "Latent-class hough forests for 3d object detection and pose estimation," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 462–477.

[37] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold *et al.*, "Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3364–3372.

[38] M. Sundermeyer, Z.-C. Marton, M. Durner, and R. Triebel, "Augmented autoencoders: Implicit 3d orientation learning for 6d object detection," *Int. Journal of Computer Vision*, pp. 1–16, 2019.

[39] G. Gao, M. Lauri, Y. Wang, X. Hu, J. Zhang, and S. Frintrop, "6d object pose regression via supervised learning on point clouds," *arXiv preprint arXiv:2001.08942*, 2020.

[40] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis *et al.*, "Bop: Benchmark for 6d object pose estimation," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 19–34.

[41] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 574–591.

[42] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 7668–7677.

[43] Z. Li, G. Wang, and X. Ji, "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation," in *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 7678–7687.

[44] R. König and B. Drost, "A hybrid approach for 6dof pose estimation," in *European Conference on Computer Vision Workshops*. Springer, 2020, pp. 700–706.

[45] J. Liu, Z. Zou, X. Ye, X. Tan, E. Ding, F. Xu, and X. Yu, "Leaping from 2d detection to efficient 6dof object pose estimation," in *European Conference on Computer Vision Workshops*. Springer, 2020, pp. 707–714.

[46] T. Hodan, D. Barath, and J. Matas, "Epos: estimating 6d pose of objects with symmetries," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 703–11 712.

[47] F. Hagelskjær and A. G. Buch, "Pointvotenet: Accurate object detection and 6 dof pose estimation in point clouds," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 2641–2645.

[48] M. Sundermeyer, M. Durner, E. Y. Puang, Z.-C. Marton, N. Vaskevicius, K. O. Arras, and R. Triebel, "Multi-path learning for object pose estimation across domains," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 13 916–13 925.

[49] 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, 2006.

[50] M.-T. Pham, O. J. Woodford, F. Perbet, A. Maki, B. Stenger, and R. Cipolla, "A new distance for scale-invariant 3d shape recognition and registration," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 145–152.

[51] T. Sølund, A. G. Buch, N. Krüger, and H. Aanæs, "A large-scale 3d object recognition dataset," in *International Conference on 3D Vision (3DV)*, 2016, pp. 73–82.

[52] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6d object pose estimation," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 606–619.

[53] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *IEEE international conference on robotics and automation*, 2009, pp. 3212–3217.

[54] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 356–369.

[55] C. Raposo and J. P. Barreto, "Using 2 point+ normal sets for fast registration of point clouds with small overlap," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5652–5658.

[56] P. Rodrigues, M. Antunes, C. Raposo, P. Marques, F. Fonseca, and J. P. Barreto, "Deep segmentation leverages geometric pose estimation in computer-aided total knee arthroplasty," *Healthcare Technology Letters*, vol. 6, no. 6, pp. 226–230, 2019.

[57] M. Sundermeyer, Z.-C. Marton, M. Durner, and R. Triebel, "Augmented autoencoders: Implicit 3d orientation learning for 6d object detection," *Int. Journal of Computer Vision*, vol. 128, no. 3, pp. 714–729, 2020.

[58] S. Zakharov, I. Shugurov, and S. Ilic, "Dpod: Dense 6d pose object detector in rgb images," *arXiv preprint arXiv:1902.11020*, 2019.

[59] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *International journal of computer vision*, vol. 103, no. 3, pp. 267–305, 2013.

[60] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic *et al.*, "Benchmarking 6dof outdoor visual localization in changing conditions," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8601–8610.

[61] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra, "Pcpnet learning local shape properties from raw point clouds," in *Computer Graphics Forum*, vol. 37, no. 2, 2018, pp. 75–85.