PPR-Net++: Accurate 6-D Pose Estimation in Stacked Scenarios

Long Zeng, *Member, IEEE*, Wei Jie Lv, Zhi Kai Dong, and Yong Jin Liu, *Senior Member, IEEE*

***Abstract*— Most supervised learning-based pose estimation methods for stacked scenes are trained on massive synthetic datasets. In most cases, the challenge is that the learned network on the training dataset is no longer optimal on the testing dataset. To address this problem, we propose a pose regression network PPR-Net++. It transforms each scene point into a point in the centroid space, followed by a clustering process and a voting process. In the training phase, a mapping function between the network's critical parameter (i.e., the bandwidth of the clustering algorithm) and the compactness of the centroid distributions is obtained. This function is used to adapt the bandwidth between centroid distributions of two different domains. In addition, to further improve the pose estimation accuracy, the network also predicts the confidence of each point, based on its visibility and pose error. Only the points with high confidence have the right to vote for the final object pose. In experiments, our method is trained on the IPA synthetic dataset and compared with the state-of-the-art algorithm. When tested with the public synthetic Siléane dataset, our method is better in all eight objects, where five of them are improved by more than 5% in average precision (AP). On IPA real dataset, our method outperforms a large margin by 20%. This lays a solid foundation for robot grasping in industrial scenarios.**

***Note to Practitioners*—Our work is motivated by industrial product assembly based on robot grasping. The industrial parts are usually manufactured by numerical machines and piled in bins. Our method can estimate the poses of visible parts accurately. A pose of a part includes its centroid and spatial orientations. Combined with a depth camera, this algorithm allows an industrial robot to understand complex stacked scenes. We improve the pose estimation accuracy in order to assemble parts with robot grasping, without an additional pose adjuster. Our network can learn from a synthetic dataset and apply it to real-world data, without a significant accuracy drop. The synthetic dataset can be obtained easily by computer simulation programs, so the training data are sufficient. Experiments demonstrate that our method outperforms the state-of-the-art pose estimation approaches.**

***Index Terms*— 6-D pose estimation, bin-picking, point-wise regression, robot grasping, stacked scenario.**

Manuscript received September 27, 2020; revised July 9, 2021; accepted August 24, 2021. This article was recommended for publication by Associate Editor K. Harada and Editor D. O. Popa upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 61972220 and Grant 61725204, in part by the University Stability Support Program of Shenzhen under Grant WDZC20200821140447001, and in part by the Grant from Science and Technology Department of Jiangsu Province, China. *(Long Zeng and Wei Jie Lv are co-first authors.) (Corresponding author: Yong Jin Liu.)*

Long Zeng is with the Department of Advanced Manufacturing, Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China (e-mail: zenglong@sz.tsinghua.edu.cn).

Wei Jie Lv is with the Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China (e-mail: lwj19@mails.tsinghua.edu.cn).

Zhi Kai Dong is with Rockchip, Fuzhou 350003, China (e-mail: zhikai.dong@rock-chips.com).

Yong Jin Liu is with BNRist, MOE Key Laboratory of Pervasive Computing, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: liuyongjin@tsinghua.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TASE.2021.3108800.

Digital Object Identifier 10.1109/TASE.2021.3108800

I. Introduction

ACCURATE 6-D object pose estimation (OPE) is a crucial prerequisite for vision-guided robot grasping from stacked scenarios. A stacked scene is a pile of objects which are randomly dropped in a bin. It is universal in real industrial applications, such as part assembly or part loading for computerized numerical control (CNC). If the pose estimation is accurate enough, a product can be assembled directly with robot grasping, i.e., without pose adjuster [1]. The OPE task is challenging due to parts' variety and heavy occlusion in a stacked scene. Thus, an accurate 6-D OPE method for real stacked scenes is needed.

Recent 6-D OPE algorithms are dominated by convolutional neural networks (CNNs)-based approaches. The recent OPE methods optimize the instance segmentation and pose estimation jointly, such as [2]–[4]. They are faster and more accurate compared with multistage estimation methods [5]. Our group previously proposed a point-wise regression network PPR-Net [2] for parts in stacked scenarios. Similar to VoxelNet [6], PPR-Net assumes that if points belong to the same part instance, their predicted centroids will be close to each other in the centroid space.[1] Thus, it first maps cluttered scene points to points in the centroid space, followed by a clustering process and a voting process. PPR-Net outperformed the state-of-the-art approaches at that time by a large margin of 15%–41% on the benchmark Siléane dataset. However, two major problems are found based on our extensive experiments (detailed in Section III). First, the hyperparameter, i.e., the bandwidth of the mean-shift clustering [7], from the training dataset is not optimal anymore for the testing dataset, since they have different noise distributions. The selection of bandwidth is critical, which affects the clustering results seriously. This reality gap is often denoted as Sim-to-Real (simulation-to-real) problem. As Sim-to-Real is universal in practical scenarios, it is desirable if there is a method that can automatically transfer the optimal bandwidth (i.e., the bandwidth corresponding to highest pose accuracy) from synthetic dataset to the

[1] A 6-D pose is described by 3-D translations, i.e., centroid, and 3-D rotations, so the centroid space (3-D) belongs to the pose space (6-D). The clustering and the visualization given in this article are performed only on centroid space.

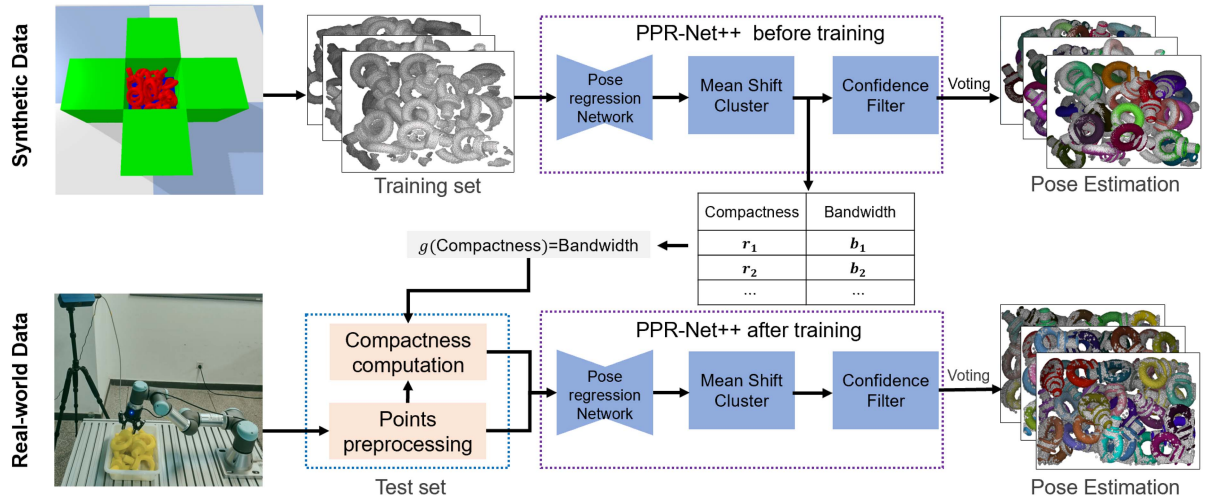


Fig. 1. PPR-Net++ overview: pose regression network is trained with synthetic data, followed by a mean-shift clustering [7] and voting process for pose estimation. In the training phase, the grid-searched optimal bandwidth and compactness of centroid distributions are recorded, where the mapping function between them is learned and used to compute optimal bandwidth for real testing data.

real dataset. Second, the final pose prediction is the average of point-wise predictions in a cluster. In fact, some points' prediction are more accurate than others. The predicted pose would be improved if unreliable points are screened out.

This article proposes a Sim-to-Real learning network for accurate 6-D pose estimation. As shown in Fig. 1, it is trained on synthetic data and adapted to real data. It consists of a point-wise pose regression network, a clustering process, and a voting process. It is extended from PPR-Net [2] with two import improvements. First, the optimal bandwidth is transferred from synthetic data to real data by a mapping function. We find that the optimal bandwidth is a function of the compactness of the centroid distributions, although they are affected by many factors, such as a part's shape, size, and the environment's noise patterns. We believe that this mapping function is common to centroid distributions of all kinds of stacked scenarios, whether synthetic or real dataset. Thus, we first design a simple metric of second-order statistic to describe the compactness of a centroid distribution. Then, the mapping function between the compactness value and bandwidth is learned in the training phase. This makes the transfer of optimal bandwidths from synthetic domain to real domain possible. It saves our manual parameter adjustment work significantly, too. Second, the PPR-Net is extended with a confidence learning module to predict the confidence of each point in the point cloud. PPR-Net screened out less reliable points with visibility only, whereas our confidence module also considers the pose error. Only the points with high confidence are selected as voted points, which improves the prediction accuracy of the 6-D pose of objects. This improved PPR-Net method is denoted as PPR-Net++.

In our experiments, PPR-Net++ is trained on the IPA synthetic dataset, whereas the Siléane dataset [8] and IPA real dataset [9] are selected as the test sets. Compared with PPR-Net, the experimental results show that the average precisions (APs) [8] of our methods are better in all eight objects of the Siléane dataset, and the prediction for Bunny and Pepper object is improved by more than 10%. Compared with the state-of-the-art algorithm OP-Net with Lori_1 and PP [3], our method is better in all ten objects, too. In particular, the APs of five in ten objects are improved more than 5%. Compared with OP-Net with Lori_2 and PP [3], the APs of seven in ten objects are significantly better, one is worse, and the other two objects are basically the same. Therefore, compared with our previous work PPR-Net [2] and the state-of-the-art method OP-Net [3], the performance of our new method exceeds a large margin on both Siléane dataset and IPA real dataset.

In summary, we substantially improve our previous work PPR-Net [2] and the major contributions are as follows.

1) We design an accurate 6-D pose regression network PPR-Net++ with a bandwidth computation and a confidence learning module.
2) We propose an effective optimal bandwidth adaptation method for PPR-Net++, allowing the transfer of optimal bandwidth parameter from synthetic data to real data.

II. Related Work

We first review the most related work on 6-D OPE methods and then the Sim-to-Real methods.

A. 6-D Object Pose Estimation

6-D OPE methods in the literature can be roughly classified into three categories: template-based, feature-based, and learning-based methods. In template-based methods [10], a rigid template and a similarity score are two necessary components. Templates are usually obtained by rendering the corresponding 3-D model in different viewpoints, and then are used to scan different location points to compute similarity scores for each location. The best match is obtained by comparing these similarity scores [11]. However, the similarity computation deteriorates when objects have severe occlusions.

The feature-based methods can handle the severe occlusion problem well. They extract local features on the 3-D models and save them in an efficient data structure in prior, such as VFH [12] and LINEMOD [13]. When necessary, the local features from points of interest in the scene are computed by using the same computational procedure and matched to pre-computed features of those 3-D models. Once the correspondences are established, from which 6-D poses can be recovered [14]. The seminal work on a local descriptor is point pair feature (PPF) [15]. Many variant versions of PPF have been proposed, like [16]. However, their performance drops significantly under complex scenarios with similar-looking objects or instances of the same object exhibit similar features.

Recently, research studies in 6-D OPE have been dominated by deep CNN approaches. The instance-level segmentation of point cloud is needed before pose estimation, since a stacked scene usually contains multiple instances of objects. Deep learning has shown remarkable performance on segmentation tasks such as Mask R-CNN [5] and its extension [17] on depth images for robot grasping. Recent learning progress on point cloud, e.g., PointNet [18], PointNet++ [19] and PoinSIFT [20], allows us to extract task-specific features directly from 3-D sensor scenarios data. With this progress, Wang *et al.* [21] proposed SGPN which is an instance segmentation architecture. It generates group proposals by similarities between point pairs in embedded feature space.

There is a trend to consider instance segmentation and pose estimation simultaneously. SSD-6-D [22] extended classical image detector SSD by regressing object poses for estimated object bounding boxes. Xiang *et al.* [23] presented a pixel-wised network with three prediction branches. The semantic branch predicts the type of each pixel. The translation branch outputs translation of each pixel for voting in Hough space, and the rotation branch directly regresses quaternion for each instance. He *et al.* [4] applied point-wise regression to simultaneously achieve semantic segmentation, instance segmentation, and keypoint prediction from the RGB-D features, and then estimated pose by least-squares fitting based on 3-D models. Our previous work PPR-Net [2] extended the idea of Hough voting from approaches [24] to point clouds using a point-wise regression learning framework. It estimated 6-D pose with point cloud containing only camera coordinate information and outperformed the previous methods by a large margin in the public benchmark Siléane dataset for bin-picking scenarios at that time.

However, there is a reality gap between the simulated data and physical data, i.e., synthetic data and real data, due to visual and dynamic differences [25]. That is, the learned network on synthetic training data is not optimal anymore on the testing data. This problem bothered us a lot in our experiments, details can be found in Section III.

B. Sim-to-Real Methods

In 6-D OPE methods, it requires significant human effort to collect and annotate a large number of real samples for training. Instead, massive simulation data generated by physic simulation engine Bullet and V-REP [26] is an alternative solution, e.g., IPA datasets [9]. Commonly used physics simulation engines are V-REP / CoppeliaSim, PyRep, MoJuCo, Blender, Gazebo, and so on [27]. Previous work has explored several strategies to bridge this gap.

One strategy is to understand the noise pattern in the real depth data and mimic this pattern in synthetic depth data or to make the simulator closely match the physical conditions by using realistic rendering techniques [28]. This technique can be easily integrated with other techniques.

Another strategy is domain randomization [29]. For example, when a deep CNN is trained, it is common to augment training data to prevent over-fitting on a small dataset, e.g., 3-D object recognition [30]. Kleeberger and Huber [3] proposed an end-to-end object pose network that was trained completely on the simulated dataset. They randomize various aspects of the simulation, such as the pose and size of distractor objects, adding noise, blurring, elastic transformation, dropout, and so on. They obtained the state-of-the-art performance on the Siléane dataset and the real-world dataset.

The third important strategy is domain adaptation. Tzeng *et al.* [31] proposed to learn a correspondence between domains that allows the real images to be mapped into a space that is understandable by the model. Bousmalis *et al.* [32] proposed a generative adversarial network (GAN) that learned a transformation in the pixel space from one domain to the other, in an unsupervised manner. James *et al.* [33] translated simulated images to a canonical simulation version which were then used for policy training, and then the trained system can be used to translate real-world images to canonical images to achieve a sim-to-real transfer.

In this article, we map the point cloud with heavy occlusion into an understandable centroid space, where the centroid distributions of different parts or domains have a simple relation to the bandwidth parameter of the mean-shift algorithm. This function is explored by experiments and adapts our network from synthetic training data to real testing data.

III. Problem Identification

This section revisits PPR-Net and analyzes its limitations to identify the research problem of this article.

A. Revisit of PPR-Net

PPR-Net [2] is an efficient point-wise 6-D pose regression network. It converts the complex 6-D OPE problem into a clustering and voting problem in centroid space, instead of a complex and unstable instance segmentation process in the point cloud with severe occlusion. The basic idea of PPR-Net is based on a simple observation: if points belong to the same part instance, their predicted centroids will be close to each other in the centroid space.

As shown in Fig. 2, the proposed regression network consists of two stages. In the first stage, the points are mapped to the centroid space with three branches, which share the common backbone module A. This backbone usually is a deep CNN $f(\{x_1, \ldots, x_{N_p}\})$ that consumes unordered point cloud directly, where $x_i \in R^{3+C}$, $3 + C$ is the number of coordinate

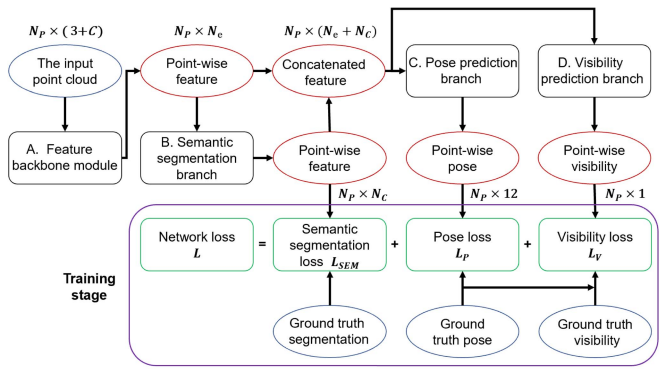


Fig. 2. PPR-Net components and their loss functions.

dimensions and additional point properties. It extracts global feature vectors of size $N_p \times N_e$ for an input point cloud $N_p \times (3+C)$, N_p is the number of original points, and N_e is the dimensionality of the global feature vector.

The semantic branch B recognizes object categories by a shared full connected layer. The input of this branch is $N_p \times N_e$ feature vectors and its output size is $N_p \times N_C$, where N_C is the predefined total number of object categories in the scene. This point-wise semantic information is helpful for other learning tasks. Thus, it is concatenated to the point feature into a new feature vector $N_e + N_c$, which is the input of pose prediction branch C. Branch C outputs a point-wise vector $N_p \times 12$, which is the 6-D pose of an object described by a 12-D vector (transformed by 3-D translation and 3-D Euler angle). It is noted that normalization is not required for rotation estimation since it is estimated as an Euler angle. Meanwhile, the concatenated feature is also inputted to the visibility prediction branch D, which outputs a visibility value of how much the object can be saw. It is useful to determine whether the estimated pose is reliable for robot grasping.

B. Bandwidth Selection Problem

The second stage is a clustering algorithm performed in the centroid space. The centroid space is first converted into a density field by mean-shift clustering algorithm [7]. It supposes that each point is a source with highest density and is propagated outside described by a kernel function $K(\cdot)$, e.g., normal kernel. Thus, the density of a specific point x is

$$f(x) = \frac{C_d}{N \cdot h^d} \sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right) \tag{1}$$

where h is the bandwidth and $[(x - x_i)/h]$ is the normalized distance between point x_i and x; C_d is normalization constant; N is the number of points in the centroid space; d is the dimensionality of the point in centroid space.

From (1), we know that the density of a point is the summation of density effects of all other points, and the influence range of each point is controlled by the bandwidth parameter h. The center of a cluster is a local maximum in the density field and can be obtained by a density gradient method. This parameter is critical to PPR-Net and it is adjusted manually. After extensive experiments, three major problems are identified as detailed below.

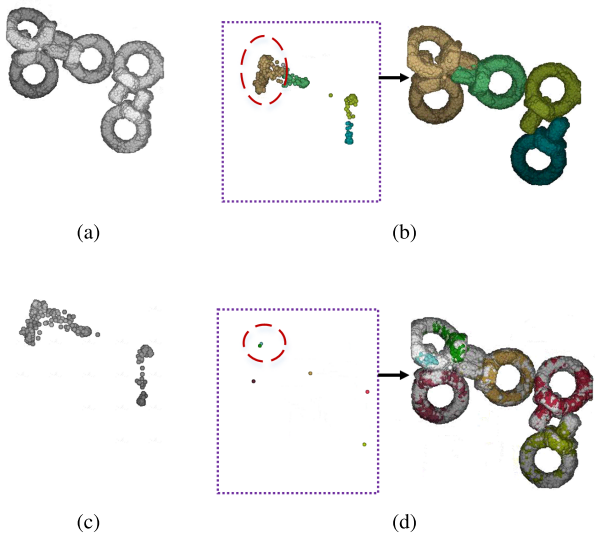


Fig. 3. Effects of bandwidth parameter: the dashed circle in (b) are classified as one centroid cluster, i.e., under-segmentation, whereas the dashed circle shown in (d), many centroids are overlapped and has two different centroid clusters, i.e., over-segmentation. (a) Point cloud. (b) Under-segmentation with large bandwidth. (c) Predicted centroids. (d) Over-segmentation with small bandwidth.

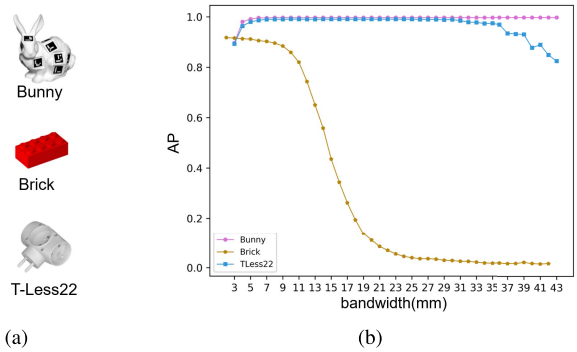


Fig. 4. Optimal bandwidth is affected by part geometry. (a) Objects. (b) AP curves.

First, the bandwidth selection will affect the number of clusters. Bandwidth is the only parameter that needs to be adjusted in the clustering stage [refer to (1)] and controls the influence range of each point. Take the five stacked ring screws shown in Fig. 3(a) for example, their centroids are visualized in Fig. 3(c). If the bandwidth is too large, the clusters will be under-segmented, i.e., the number of clusters is less than the real object number. As the dashed circle in Fig. 3(b) shown, the centroids are classified as one cluster. If the bandwidth is too small, the results will be over-segmented. For example, the dashed circle shown in Fig. 3(d) has two different centroid clusters.

Second, the optimal bandwidth is affected by part geometry. Fig. 4(a) shows three objects, i.e., Bunny, Brick, and T-Less22. They have different geometric shapes and Fig. 4(b) shows their corresponding AP curves, i.e., value pairs of bandwidth and corresponding APs. The optimal bandwidth can be obtained from an AP curve (see Section V-C). From these curves,

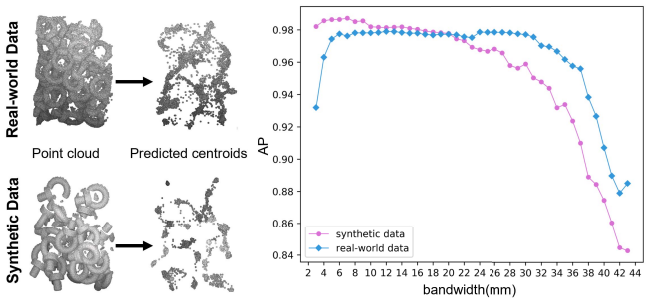


Fig. 5. Optimal bandwidths are different for synthetic and real data. (a) Comparison of prediction. (b) AP curves.

we know the optimal bandwidths for different parts are quite different, too. That is, the optimal bandwidth for one part is not optimal anymore for another part in most cases.

Third, the optimal bandwidth also depends on the noise patterns in centroid distributions. In general, real-world data have more noise than synthetic data. As shown in Fig. 5(a), when both point clouds are mapped to centroid space, the difference between their centroid distributions is significant. Fig. 5(b) shows their AP curves. From Fig. 5, we know that the optimal bandwidth learned from synthetic training data cannot be applied to real training data directly, in most cases.

C. Accuracy Loss From Improper Voting Rule

In PPR-Net, the predicted pose of the jth cluster P^j is equal to the pose average of all its points, i.e.,

$$P^j = \frac{1}{N_j}\sum_{i=1}^{N_j} p_i \tag{2}$$

where N_j is the number of points in the jth cluster.

This formula means that all points in the cluster votes to its object's final pose equally. However, we argue that if more accurate points have higher voting rights, the pose accuracy will be better. The error of ith point contains a translational error L_{t_i} and a rotational error L_{R_i} [8], which are computed by

$$\begin{aligned} L_{t_i} &= \left\| t_i^{\text{pred}} - t^{gt} \right\| \\ L_{R_i} &= \min_{r_1 \in R\left(R_i^{\text{pred}}\right), r_2 \in R(R^{gt}),} \|r_2 - r_1\|. \end{aligned} \tag{3}$$

Fig. 6(a) shows the Bunny object and its predicted centroids, and Fig. 6(b) shows its translational and rotational error statistics. From these statistics, we know that most points are near the mean error. It is intuitive that if points with large errors are selected to vote, their pose errors will deteriorate the part's final pose. Thus, it is reasonable to filter out those points with large errors.

PPR-Net has a filter mechanism based on the point's visibility. However, based on our observation, a point with high visibility does not necessarily mean its predicted pose has higher accuracy. Therefore, to solve this problem, we improve PPR-Net network to assign a point-wise confidence value, considering both point's visibility and pose error, which is detailed in Section IV.

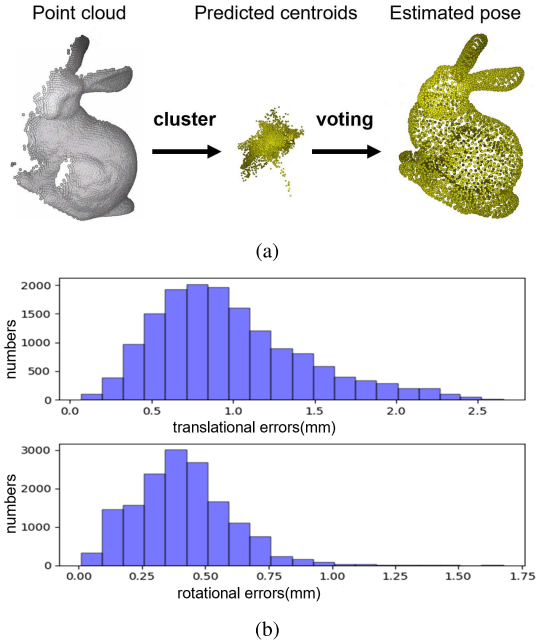


Fig. 6. Translational and rotational errors. (a) Centroids. (b) Error statistics.

IV. PPR-Net++: Accurate Pose Estimation Network

To solve the aforementioned problems, a new point-wise pose regression architecture, denoted as PPR-Net++, is designed. As shown in Fig. 7, different to PPR-Net, it also contains pose and visibility prediction branches. Similarly, it consumes unordered point clouds with N_p points. The global point-wise feature vectors $F^e = \{f_i^e\}_{i=1}^{N_p}$ of size $N_p \times N_e$ are extracted by the backbone network. Then, the pose branch and visibility prediction branch consume F_e with shared MLPs to obtain the predicted point-wise centroid offset $of_i^{\text{pred}} = [\delta x_i, \delta y_i, \delta z_i]$, rotation in Euler angles $\text{angle}_i^{\text{pred}} = [\alpha_i, \beta_i, \gamma_i]$, visibility $v_i^{\text{pred}} \in [0, 1]$.

Compared with PPR-Net, PPR-Net++ has two additional modules, i.e., bandwidth computation and confidence learning modules, which are detailed in Sections IV-A and IV-B, respectively.

A. Bandwidth Computation Module

From Section III-B, we know that the optimal bandwidth is affected by many factors, such as bandwidth configuration, part geometry, and noise patterns. They are coupled together in an unknown and complex way, so we treat them together as a black box and consider the relations between varying parameters and optimal bandwidth as a function. Specifically, our strategy for bandwidth selection is to use small bandwidth for centroid distribution with high compactness to prevent under-segmentation, and large bandwidth for data distribution with small compactness to prevent over-segmentation.

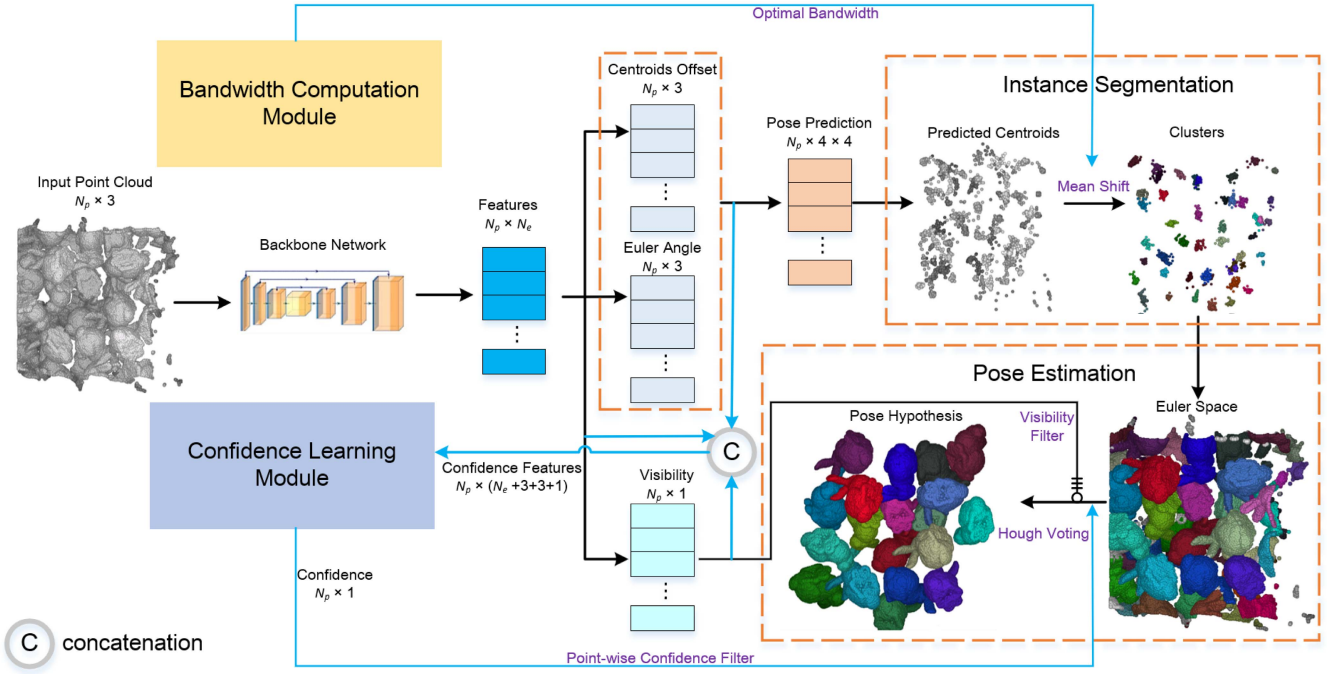


Fig. 7. Architecture of PPR-Net++, where the blue arrows are to illustrate the improvements of PPR-Net++ based on PPR-Net, including bandwidth computation module and confidence learning module.

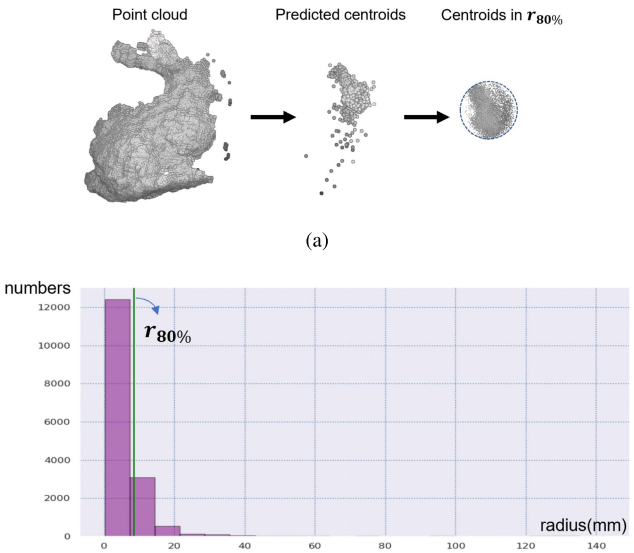


Fig. 8. Geometric meaning of $r_{80\%}$. (a) $r_{80\%}$ of the Bunny. (b) Error distribution chart.

Compactness, σ, describes how tight the points are distributed around its ideal center. It is similar to the standard deviation of the normal distribution, which describes the average deviation from its mass center. Thus, compactness is also a second-order statistic variable. Its geometric meaning can be explained by the error distribution chart, as shown in Fig. 8(b), which is the error distribution chart of the Bunny object in Fig. 8(a). In this chart, the horizontal axis is the deviation error, i.e., radius distance from points to their geometric center. The vertical axis is the number of points with this error value. Thus, it is reasonable to approximate the compactness with the radius of a sphere surrounding all centroids. The smaller the radius is, the greater the compactness of centroid distribution is.

We found that the optimal bandwidth is a function of the compactness of the centroid distribution, i.e.,

$$h_{\text{optimal}} = g(\sigma) \tag{4}$$

where h_{optimal} is the optimal bandwidth, σ is the compactness, and $g(\cdot)$ is the mapping function.

From Fig. 8(b), we know most points have their deviation errors less than 10 mm, only small proportion of points have errors greater than 10 mm. Thus, in order to eliminate the interference of outliers, we usually take the radius of the surrounding sphere to include 80% points, denoted as $r_{80\%}$. In a stacked scene, the $r_{80\%}$ is the average value of all clusters in this scene. Fig. 8(b) shows the $r_{80\%}$ on the error distribution chart of the Bunny object. Based on the previous analysis, the larger the $r_{80\%}$ is, the smaller the compactness of centroid distribution is and the larger the bandwidth is.

We hope to learn a specific mapping function during the training phase. In the testing phase, once the compactness of centroid distribution is obtained, we can compute the optimal bandwidth via this mapping function. The specific mapping function $g(\cdot)$ is obtained experimentally in Section V-C.

B. Confidence Learning Module

The confidence value represents the accuracy confidence of the point's predicted object pose. We argue that if the points

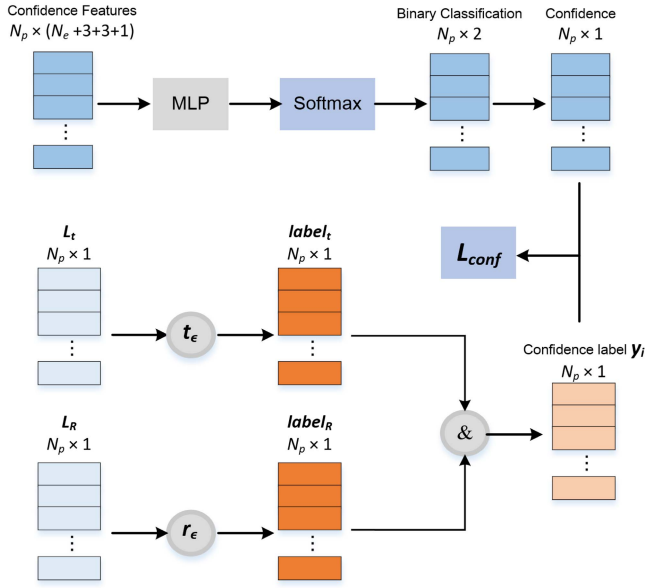


Fig. 9. Confidence learning module architecture.

of a cluster are selectively allocated voting rights, the pose accuracy can be further improved. Thus, a confidence learning module is designed to predict the point-wise confidence from point-wise feature and prediction error.

The confidence learning module's architecture is detailed in Fig. 9, consisting of MLPs and a softmax layer. Its input confidence feature vector is a concatenation of four elements as $\{[f_i^e, of_i^{\text{pred}}, \text{angle}_i^{\text{pred}}, v_i^{\text{pred}}]\}_{i=1}^{N_p}$. Its output is a binary classification vector $p_i = [p_{i\_\text{cls}_0}, p_{i\_\text{cls}_1}]$ for each point, where $p_{i\_\text{cls}_0} \in [0,1]$ and $p_{i\_\text{cls}_1} \in [0,1]$ are the binary classification probability. Then, the $p_{i\_\text{cls}_1}$ is taken as point-wise confidence, i.e., $\text{conf}_i^{\text{pred}} = p_{i\_\text{cls}_1} \in [0,1]$. Finally, the cross-entropy loss function is adopted to compute the module error

$$L_{\text{conf}} = -\frac{1}{N_p} \cdot \sum_{i=1}^{N_p} \left[y_i \cdot \lg \text{conf}_i^{\text{pred}} + (1-y_i) \cdot \lg \left(1 - \text{conf}_i^{\text{pred}}\right) \right] \tag{5}$$

where $\{y_i\}_{i=1}^{N_p}$ is the point-wise label of the binary classification and $\{\text{conf}_i^{\text{pred}}\}_{i=1}^{N_p}$ is the probability of the positive sample cls_1 in the binary classification.

In (5), the binary classification labels $\{y_i\}_{i=1}^{N_p}$, are computed on-fly during the training phase, by bitwise operation of the translation and rotation confidence labels, i.e., $\{\text{label}_{t_i}\}_{i=1}^{N_p}$ and $\{\text{label}_{R_i}\}_{i=1}^{N_p}$, respectively,

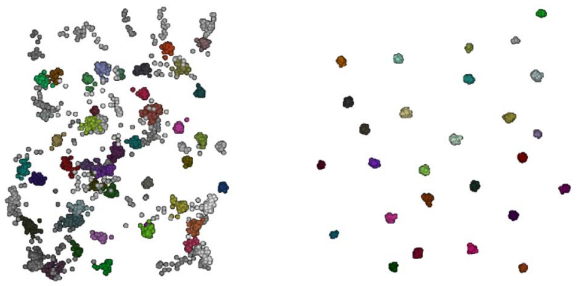
$$\{y_i\}_{i=1}^{N_p} = \{\text{label}_{t_i}\}_{i=1}^{N_p} \ \& \ \{\text{label}_{R_i}\}_{i=1}^{N_p} \tag{6}$$

where label_{t_i} and label_{R_i} are computed by

$$\text{label}_{t_i} = \begin{cases} 1, & \text{if } L_{t_i} < t_\epsilon, \\ 0, & \text{if } L_{t_i} \geq t_\epsilon, \end{cases} \qquad \text{label}_{R_i} = \begin{cases} 1, & \text{if } L_{R_i} < r_\epsilon \\ 0, & \text{if } L_{R_i} \geq r_\epsilon \end{cases}$$

where L_{t_i}, L_{R_i}, t_ϵ, and r_ϵ are translational error, rotational error, translational error threshold, and rotational error threshold.

From the cross-entropy loss function in (5), we know the probability value of cls_1 is trained to learn $y_i = 1$, which represents both the translational error and rotational error are small. Therefore, it is reasonable to treat the point-wise confidence as a criterion to screen out points. The higher confidence of a point is, the smaller the translational error and rotational error of the regression pose of the point is, so the confidence learning module can eliminate lots of unreliable points. Thus, it can further improve the accuracy and stability of the OPE. Fig. 10 is the centroid distribution of the Bunny object before and after screening points with less confidence value.



(a) (b)

Fig. 10. Centroids distribution of the Bunny object and confidence screening. (a) Without confidence screening. (b) With confidence screening.

TABLE I

IPA BIN-PICKING DATASET AND SILÉANE DATASET

Dataset	Object	# Drop limit	# Train cycles	# Test cycles	# Scenes
IPA synthetic dataset	Bunny	80	500	10	41,310
	C.Stick	60	500	10	31,110
	Pepper	90	500	10	46,410
	Brick	150	500	10	77,010
	Gear	60	500	10	31,110
	T-Less 20	99	500	10	51,000
	T-Less 22	100	500	10	51,510
	T-Less 29	79	500	10	40,800
IPA real dataset	Gear shaft	30	500	10	15,810
	Ring screw	35	500	10	18,360
	Gear shaft (real)	22	0	10	230
	Ring screw (real)	28	0	10	290
Siléane synthetic dataset	Bunny	80	0	4	324
	Candlestick	60	0	2	122
	Pepper	90	0	2	182
	Brick	150	0	2	302
	Gear	60	0	2	122
	T-Less 20	99	0	2	200
	T-Less 22	100	0	2	202
	T-Less 29	79	0	2	160

V. EXPERIMENTS

In this section, the benchmarked datasets and results are given in Sections V-A and V-B. Then, the optimal bandwidth computation and architecture are analyzed in Sections V-C and V-D, respectively.

A. Datasets and Evaluation Metrics

To evaluate the performance of the PPR-Net++, the public Siléane dataset [8] and Fraunhofer IPA bin-picking dataset [9] are evaluated. Their contents and formats are summarized in Table I. The Siléane dataset contains eight objects: Brick, Bunny, Candlestick (C.Stick), Gear, Pepper, T-Less20, T-Less22, and T-Less29. Each object only has 2–4 test cycles,

TABLE II
POSE ESTIMATION RESULTS ON SILÉANE DATASET, EVALUATION METRIC IS $10\% \times D$

Object	Bunny	C.Stick	Pepper	Brick	Gear	T-Less20	T-Less22	T-Less29	Ring screw	Gear shaft
PPF [15, 8]	0.29	0.16	0.06	0.08	0.62	0.20	0.08	0.19	-	-
PPF PP [15, 8]	0.37	0.22	0.12	0.13	0.63	0.23	0.12	0.23	-	-
LINEMOD+ [35, 8]	0.39	0.38	0.04	0.31	0.44	0.25	0.19	0.20	-	-
LINEMOD+ PP [35, 8]	0.45	0.49	0.03	0.39	0.50	0.31	0.21	0.26	-	-
Sock et al. [36]	0.74	0.64	0.43	-	-	-	-	-	-	-
OP-Net with $Lori_1$ and PP [3]	0.94	0.97	**0.98**	0.42	0.84	0.88	0.86	0.58	0.93	0.99
OP-Net with $Lori_2$ and PP [3]	0.76	0.96	0.93	**0.80**	0.60	0.58	0.55	0.39	0.75	**1.00**
PPR-Net [2]	0.82	0.91	0.80	-	-	0.81	-	-	-	-
PPR-Net with ICP [2]	0.89	0.95	0.84	-	-	0.85	-	-	-	-
PPR-Net++	**0.99**	**0.98**	**0.98**	0.47	**1.00**	**0.93**	**0.92**	**0.94**	**0.98**	0.99

with no training cycles. A cycle means the object model is dropped to a bin randomly one by one, starting from one to its drop limit. Each drop configuration is a scene. Taking the Bunny object in the test cycles of Siléane dataset, for example, its drop limit is 80, thus it has 80 scenes with stacked objects (from 1 to 80) and a 1 background scene with an empty bin. Thus, there are 81 scenes for one cycle and four test cycles, so the total number of scenes of the Bunny object is $324 = 81 \times 4$. The test cycles of the Siléane dataset have unknown noise patterns to mimic real-world point clouds. The Fraunhofer IPA synthetic dataset extends the Siléane dataset with plenty of cycles for training, where both training and testing datasets are simulated data. In IPA real dataset, two industrial parts, i.e., ring screw and gear shaft, are added. The training set is simulation data, while the test set is real-world data, with real labels. Fraunhofer IPA bin-picking dataset is the first public industrial dataset for instance segmentation and 6-D pose estimation. Different from the Siléane dataset, the training set in the IPA dataset has no noise data.

The evaluation metric proposed by Brégier *et al.* [8] is adopted, which defines the distance as the minimum Euclidean distance between the equivalent pose sets

$$\text{Dist}(R(P_1), R(P_2)) = \min_{p_1 \in R(P_1), p_2 \in R(P_2)} \|p_2 - p_1\| \qquad (7)$$

where $R(P)$ is an equivalent set of poses. Each pose is vectorized into up to 12-D vector.

This formula considers the object's symmetry and is easy to compute. Generally, when the minimum distance between the predicted pose and the ground truth pose is less than 10% of the diameter of the object's minimum bounding sphere, i.e., $10\% \times D$ denoted as distance threshold, the result of pose prediction is considered to be correct. The smaller the distance threshold is, the stricter the evaluation standard is, the more difficult the object's 6-D pose recognition is, and the higher recognition accuracy is required. In our experiments, three distance thresholds, i.e., $10\% \times D$, $5\% \times D$, and $2\% \times D$, are used.

B. Pose Estimation and Comparison

We train our network PPR-Net++ on the IPA synthetic dataset, where a random Gaussian noise between -2 and 2 mm is added to each point. Then, it is tested on both synthetic Siléane dataset and IPA real dataset. Their optimal bandwidths are computed via the $g_{S2R}(r_{80\%}^{\text{single}})$ function obtained experimentally in Section V-C since their noise patterns are different to IPA synthetic dataset ($_{S2R}$ represents Sim-to-Real). The pose accuracy is evaluated under the $10\% \times D$ distance threshold, same with previous methods. Table II lists the performance of our method against the state-of-the-art OPE methods. Obviously, compared with learning methods, these traditional non-learning methods have inferior performance on the Siléane dataset, such as PPF [8], [15], PPF PP [8], [15], LINEMOD+ [8], [34], and LINEMOD+ PP [8], [34].

On the synthetic Siléane dataset, our new method is better in all eight objects than that of PPR-Net. More specifically, the APs of the Bunny and Pepper objects are improved by about 10%. Other objects have certain improvements at a different level. For Sim-to-Real experiments, we trained on IPA synthetic dataset and tested on both Siléane dataset and IPA real dataset, with ten objects. Our results are compared with the state-of-the-art algorithm OP-Net [3], which has two versions. Compared with OP-Net with Lori_1 and PP, our method is better in all ten objects. In particular, the APs of seven in ten objects are improved by at least 5%. Compared with OP-Net with Lori_2 and PP, the APs of seven in ten objects are significantly better, one is worse, and the other two objects are basically the same.

On the IPA real dataset, there are only two labeled objects. PPR-Net++ and OP-Net have approximately the same performance on the Gear shaft object since its geometry is simple. For the complex ring screw, the AP of our method is improved by 5%, compared with OP-Net with Lori_1 and PP. When compared with OP-Net with Lori_2 and PP, this improvement is 23%, which is quite a large margin.

The qualitative comparison of PPR-Net and PPR-Net++ is shown in Fig. 11 with the same scene. PPR-Net uses an empirical bandwidth for the clustering algorithm and thus results in under-segmentation, which further leads to instance segmentation error and 6-D pose estimation error, as the dotted red circle line shown. However, PPR-Net++ uses the optimal bandwidth computed from the mapping function for clustering and the confidence to filter the unreliable point predictions. It overcomes previous under-segmentation problem automatically.

Therefore, compared with PPR-Net and the state-of-the-art method OP-Net, the performance of our new method surpasses

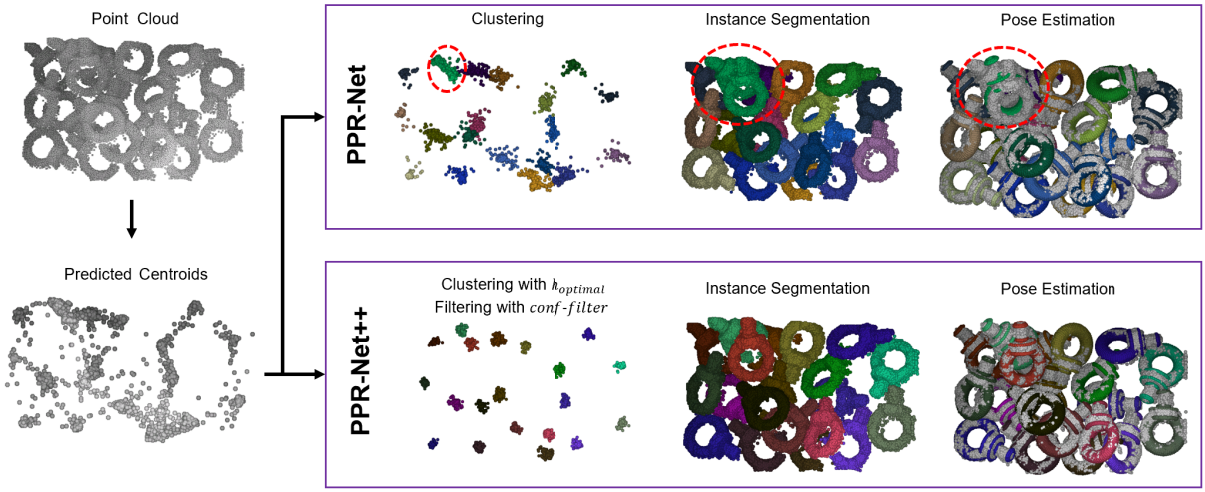


Fig. 11. Qualitative comparison of PPR-Net and PPR-Net++, where PPR-Net uses empirical bandwidth (35 mm) for clustering, while PPR-Net++ uses optimal bandwidth (17 mm) calculated by our mapping function for clustering and the confidence to filter the unreliable point predictions.

a large margin on both synthetic Siléane dataset and IPA real dataset.

C. Analysis on Optimal Bandwidth Computation

The Sim-to-Real problem between a train and a testing dataset actually considers whether they have the same noise distribution. Its content can be generalized. Sim-to-Sim has similar noise patterns for source and target datasets. This usually happens when point clouds are generated from the same virtual camera configurations. Sim-to-Real has different noise patterns for source and target datasets. The IPA synthetic training dataset and IPA real testing dataset in Table I belong to Sim-to-Real type, since their noise patterns are quite different.

The procedure to compute the mapping function for both types is similar. That is, a specific mapping function is learned by fitting a set of discrete points $(r_{80\%}, h_{\text{search}})$ with a template function, such as polynomial or exponential function. h_{search} is an optimal bandwidth obtained manually from three $10\% \times D$, $5\% \times D$, and $2\% \times D$ AP curves, as the Brick object shown in Fig. 12. We first choose a small range near the optimal bandwidth for each AP curve. Then, the global optimal bandwidth is a value selected from the intersection ranges of the three AP curves.

The computation of the compactness $r_{80\%}$ is different for the above two types. There is no way to directly compute the compactness of a testing scene's centroid distribution since we do not know which points belong to which instances in advance, different from a training scene where we have the point-wise labels. For the Sim-to-Sim type, since the training dataset has the same distribution as the testing dataset, it is reasonable to directly use the learned $r_{80\%}$ in the training phase to the testing phase. It is denoted as $r_{80\%}^{\text{stacked}}$ since the compactness is the average of all instances in a stacked scene. For the Sim-to-Real type, we cannot apply the learned $r_{80\%}$ in the training phase to the testing phase, since the training dataset has the different distribution with the testing dataset. Alternatively, we found we can easily compute the $r_{80\%}$ of the scene with only one single object in a cycle, denoted as $r_{80\%}^{\text{single}}$.

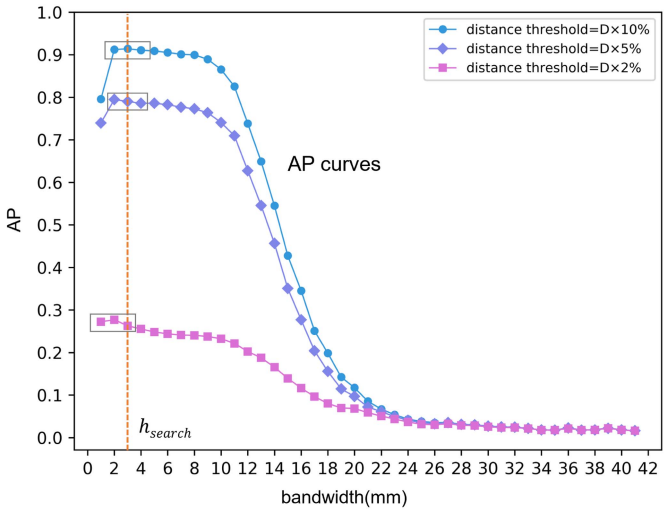


Fig. 12. Optimal bandwidth obtained from 10%, 5%, and 2% AP curves.

Fig. 13 shows the eight discrete points $(r_{80\%}^{\text{stacked}}, h_{\text{search}})$ experimented on the eight objects of IPA synthetic dataset. From this figure, we guess there is a linear relationship between the optimal bandwidth and $r_{80\%}^{\text{stacked}}$. Based on the principle of Occam razor, in order to ensure the generalization ability of the mapping function, we use the linear function to fit the given points and obtain

$$h_{\text{optimal}} = g_{S2S}\left(r_{80\%}^{\text{stacked}}\right) \approx 1.17 \times r_{80\%}^{\text{stacked}} + 0.57. \quad (8)$$

Similar to the Sim-to-Sim type, we can obtain eight discrete points $(r_{80\%}^{\text{single}}, h_{\text{search}})$ experimented on the eight objects of IPA synthetic dataset. The linear function to fit the given points is

$$h_{\text{optimal}} = g_{S2R}\left(r_{80\%}^{\text{single}}\right) \approx 1.45 \times r_{80\%}^{\text{single}} + 1.35. \quad (9)$$

D. Analysis on Architecture Design

Compared with PPR-Net, the bandwidth computation module and confidence learning module are newly added

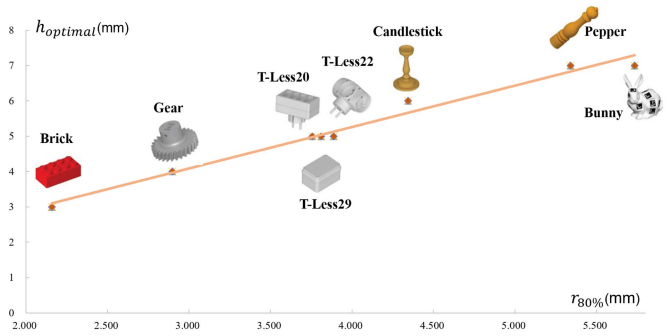


Fig. 13. Mapping function fit from eight discrete points ($r_{80\%}^{\text{stacked}}$, h_{search}).

TABLE III

EFFECTS OF BANDWIDTH COMPUTATION AND CONFIDENCE MODULES EXPERIMENTED ON IPA SYNTHETIC DATASET (OURS AND W/O MEANS PPR-NET++ AND WITHOUT, RESPECTIVELY)

Distance threshold	Method	Brick	Bunny	C.stick	Pepper
$10\% \times D$	PPR-Net	0.908	**0.997**	0.966	**0.996**
	Ours w/o $h_{optimal}$	0.927	**0.997**	0.966	0.995
	Ours w/o $conf$	**0.930**	**0.997**	**0.996**	0.992
	Ours	0.921	0.996	**0.996**	0.993
$2\% \times D$	PPR-Net	0.249	0.931	0.953	0.962
	Ours w/o $h_{optimal}$	**0.290**	0.940	0.958	**0.974**
	Ours w/o $conf$	0.254	0.941	0.985	0.972
	Ours	0.267	**0.943**	**0.986**	0.969

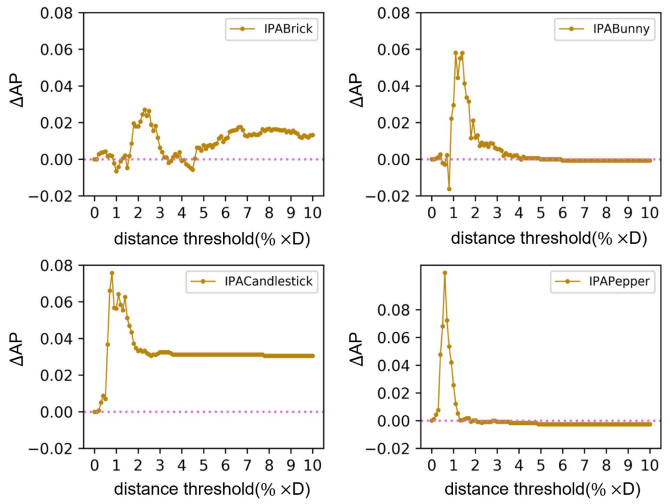


Fig. 14. AP improvements with respect to distance threshold compared with PPR-Net.

modules. To test their effects, one cycle of the four objects, i.e., Brick, Bunny, Candlestick, Pepper from the Fraunhofer IPA bin-picking synthetic dataset, are selected for evaluation. The optimal bandwidths are computed via the $g_{S2S}(r_{80\%}^{\text{stacked}})$, since the centroid distributions of training and testing dataset are the same. The compared methods include PPR-Net and PPR-Net++, while PPR-Net++ without confidence and PPR-Net++ w/o h_{optimal} are the PPR-Net++ without confidence and bandwidth computation module, respectively. The bandwidth of PPR-Net++ w/o h_{optimal} is the same as that of PPR-Net.

Their results are summarized in Table III. From this table, we know that their APs are very close to each other, under a

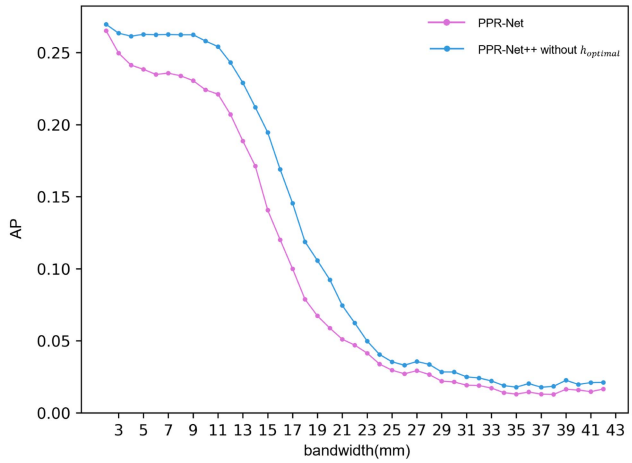


Fig. 15. AP curves of the bunny with PPR-Net and PPR-Net++ without h_{optimal}.

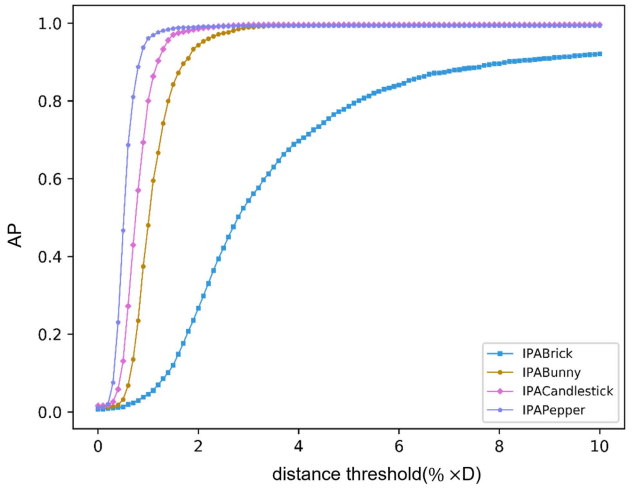


Fig. 16. AP-distance threshold curves.

larger threshold, e.g., $10\% \times D$. However, when the evaluation metric is stricter, e.g., $2\% \times D$, the bandwidth computation module, and confidence learning module have an obvious improvement on APs.

Fig. 14 shows the AP improvement under different distance thresholds. The vertical axis is the AP improvement values of PPR-Net++ over PPR-Net. From this figure, we know that there are about 3%–9% AP improvements when we apply a stricter evaluation metrics.

From Table III, we also know that the improvements of PPR-Net++ may not be greater than the cumulative effect of the two modules used alone. Their effects are coupled together. The bandwidth computation module can effectively improve the clustering results by filtering the outliers, i.e., the points far away from the center of the cluster. These outliers usually have a high probability that they also have higher prediction errors, which will lead these outliers to have lower confidence values. Therefore, they are coupled together. However, the combined effect of the two modules is generally better than that only a single module is used.

In Fig. 15, we plot the trend of AP with bandwidth, for the Brick object under the distance threshold of $2\% \times D$. It shows that the confidence learning module can help to maintain

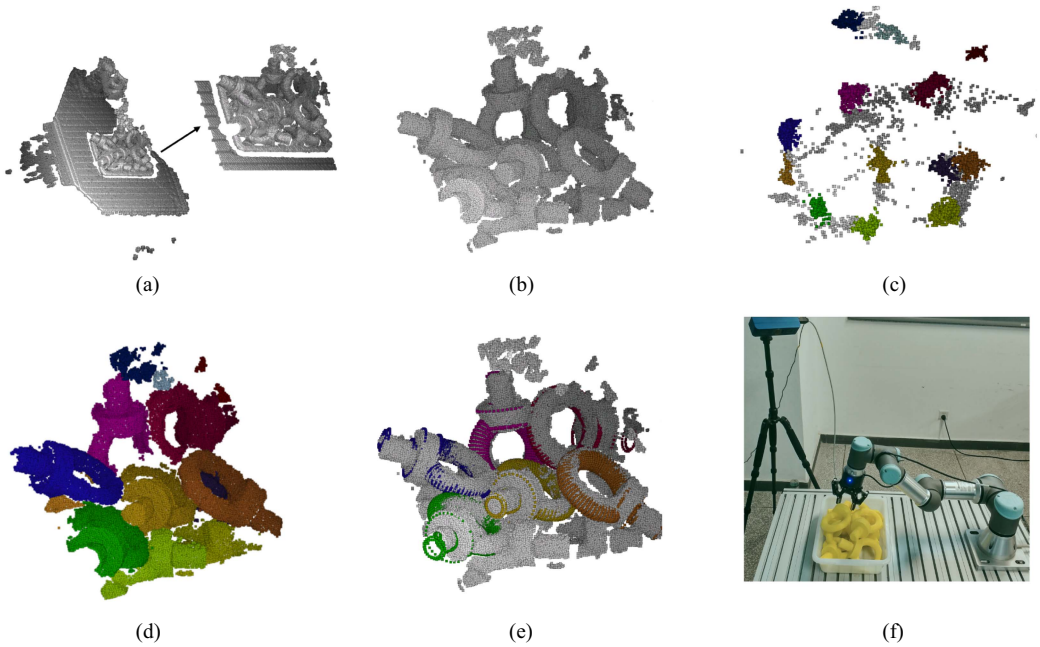


Fig. 17. Experiment on real-world robot bin-picking. (a) Scene point clouds. (b) Processed point clouds. (c) Predicted centroids. (d) Instance segmentation. (e) Pose estimation. (f) Grasp experiments.

high accuracy in a certain range, reduce the dependence on bandwidth, and make the pose prediction more stable. Therefore, the comprehensive use of the bandwidth computing module and the confidence learning module can not only reduce the workload of manual parameter adjustment but also improve the accuracy and stability.

In Fig. 16, we further plot the trend of AP with distance threshold, for Brick, Bunny, Candlestick, and Pepper in the IPA synthetic dataset. And we know that with even more stringent evaluation metrics, such as $2\% \times D$, the performance of our method is still satisfied for most objects. However, there is a threshold below which accuracy drops sharply, for each part.

E. Real-World Experiments on Robot Bin-Picking

The PPR-Net++ was integrated into our robot grasping pipeline. To demonstrate its effectiveness in the real-world scenarios, we printed the ring screw object and put them in a box randomly. In the experiment, the point clouds were captured by a fixed Ensenso N35 range camera. The goal was to correctly pick all object instances and place them in the target positions. For this experiment, we trained PPR-Net++ on the IPA ring screw synthetic dataset and tested with 3000 annotated synthetic scenes with noise (200 cycles, 1–15 instances in a scene). These testing data were generated by setting the virtual camera the same as our real Ensenso N35 camera configuration. We adopted the same evaluation method as Section V-A to calculate AP for the ring screw. The quantitative evaluation showed that PPR-Net++ achieves AP of 0.99 for ring screw on the test set.

Our grasping pipeline is shown in Fig. 17. We locate a predefined box in the point clouds to get scene point clouds as shown in Fig. 17(a), and followed by a background subtraction and furthest point sampling, as shown in Fig. 17(b). The filter points are fed into PPR-Net++ to predict centroids with h_{optimal} computed via the $g_{S2R}(r_{80\%}^{\text{single}})$ and confidence filter [Fig. 17(c)]. The segmented instances and estimated poses are shown in Fig. 17(d) and (e), respectively.

We deployed the aforementioned pipeline on a UR3 robotic arm with Robotiq 2-Finger 85 gripper as end-effector as shown in Fig. 17(f). The grasping trajectory was generated with motion planning software MoveIt! We evaluated PPR-Net++ in the grasping trials and our pipeline was able to pick and place all graspable object instances in all trials, including in heavily cluttered scenes with significant occlusion.

In the near future, we will integrate advanced grasping affordance [36] and grasping point computation methods [37], to improve the stability of our grasping system.

VI. Conclusion

To address the Sim-to-Real problem, we designed a point-wise 6-D pose estimation network PPR-Net++ that maps points into centroid space, followed by a clustering and voting process. We also explored the function between the optimal bandwidth and compactness of the centroid distribution. The learned function can not only save lots of manual parameter adjustment work, but also allow to adapt the learned optimal bandwidth from the training dataset to the testing dataset. In addition, a confidence module was inserted to compute a point-wise confidence value based on its visibility and pose error, to screen out unreliable points. This further improved the object pose prediction accuracy. The accuracy of PPR-Net++ exceeds a large margin on the public benchmark dataset,

compared with the state-of-the-art methods. The code is available at https://github.com/lvwj19/PPR-Net-plus.

ACKNOWLEDGMENT

The authors would like to express their sincere thanks to the reviewers for their valuable comments and suggestions.

REFERENCES

[1] P. L. Varkonyi, "Estimating part pose statistics with application to industrial parts feeding and shape design: New metrics, algorithms, simulation experiments and datasets," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 658–667, Jul. 2014.

[2] Z. Dong *et al.*, "PPR-Net: Point-wise pose regression network for instance segmentation and 6d pose estimation in bin-picking scenarios," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1773–1780.

[3] K. Kleeberger and M. F. Huber, "Single shot 6D object pose estimation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6239–6245.

[4] 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/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11632–11641.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020.

[6] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4490–4499.

[7] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[8] R. Bregier, F. Devernay, L. Leyrit, and J. L. Crowley, "Symmetry aware evaluation of 3D object detection and pose estimation in scenes of many parts in bulk," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 2209–2218.

[9] K. Kleeberger, C. Landgraf, and M. F. Huber, "Large-scale 6D object pose estimation dataset for industrial bin-picking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 2573–2578.

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

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

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

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

[14] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis, "6-DoF object pose from semantic keypoints," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 2011–2018.

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

[16] H. Dong, D. K. Prasad, and I.-M. Chen, "Object pose estimation via pruned Hough forest with combined split schemes for robotic grasp," *IEEE Trans. Autom. Sci. Eng.*, early access, Sep. 17, 2020, doi: 10.1109/TASE.2020.3021119.

[17] M. Danielczuk *et al.*, "Segmenting unknown 3D objects from real depth images using mask R-CNN trained on synthetic data," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7283–7290.

[18] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 77–85.

[19] C. R. Qi, Y. Li, H. Su, and L. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[20] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation," 2018, *arXiv:1807.00652*. [Online]. Available: http://arxiv.org/abs/1807.00652

[21] W. Wang, R. Yu, Q. Huang, and U. Neumann, "SGPN: Similarity group proposal network for 3D point cloud instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2569–2578.

[22] 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. (ICCV)*, Oct. 2017, pp. 1521–1529.

[23] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Robotics: Science and Systems*. Cambridge, MA, USA: MIT Press, 2018.

[24] 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.

[25] S. James and E. Johns, "3D simulation for robot arm control with deep Q-learning," in *Proc. NeurIPS Workshop Deep Learn. Action Interact.*, 2016, pp. 1–6.

[26] E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 1321–1326.

[27] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robot. Rep.*, vol. 1, pp. 239–249, Sep. 2020.

[28] B. Planche *et al.*, "DepthSynth: Real-time realistic synthetic data generation from CAD models for 2.5D recognition," in *Proc. Int. Conf. 3D Vis. (3DV)*, Oct. 2017, pp. 1–10.

[29] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 23–30.

[30] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," in *Proc. ECCV*. Berlin, German: Springer, 2018, pp. 90–105.

[31] E. Tzeng *et al.*, "Adapting deep visuomotor representations with weak pairwise constraints," in *Proc. Workshop Algorithmic Found. Robot. (WAFR)*, 2016, pp. 688–703 .

[32] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 95–104.

[33] S. James *et al.*, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12627–12637.

[34] 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.* Berlin, German: Springer, 2012, pp. 548–562.

[35] J. Sock, K. I. Kim, C. Sahin, and T. K. Kim, "Multi-task deep networks for depth-based 6d object pose and joint registration in crowd scenarios," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2018, pp. 1–12.

[36] H. O. Song, M. Fritz, D. Goehring, and T. Darrell, "Learning to detect visual grasp affordance," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 798–809, Apr. 2016.

[37] X. Fu, Y. Liu, and Z. Wang, "Active learning-based grasp for accurate industrial manipulation," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1610–1618, Oct. 2019.

Long Zeng (Member, IEEE) received the B.S. degree from Zhejiang University, Hangzhou, China, in 2007, and the Ph.D. degree in mechanical engineering from The Hong Kong University of Science and Technology, Hong Kong, in 2012.

He is an Associate Professor with Shenzhen International Graduate School, Tsinghua University, Shenzhen, China. His research interest is intelligent Computer Aided Design/Computer Aided Manufacturing (CAD/CAM) with a recent focus on robotic vision and flexible assembly.

Wei Jie Lv received the bachelor's degree from Huazhong University of Science and Technology, Wuhan, China, in 2019. He is currently pursuing the master's degree with the Department of Mechanical Engineering, Tsinghua University, Beijing, China.

His current research interests include 6-D pose estimation and point clouds segmentation for industrial parts, with a recent focus on deep-learning-driven neural networks for point clouds.

Zhi Kai Dong received the bachelor's degree from Northwestern Polytechnical University, Xi'an, China, in 2017, and the master's degree from the Department of Mechanical Engineering, Tsinghua University, Beijing, China, in 2020.

He is currently working on research at Rockchip, Fuzhou, China. His current research interests include 6-D pose estimation based on point clouds, object detection based on images, and image processing.

Yong Jin Liu (Senior Member, IEEE) received the B.Eng. degree from Tianjin University, Tianjin, China, in 1998, and the Ph.D. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2004.

He is currently a Professor with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. His research interests include computational geometry, computer graphics, and computer-aided design.

Dr. Liu is a member of Association for Computing Machinery (ACM).