ParametricNet: 6DoF Pose Estimation Network for Parametric Shapes in Stacked Scenarios

Long Zeng†, Wei Jie Lv†, Xin Yu Zhang, and Yong Jin Liu*

***Abstract*— Most industrial parts are parametric and their special properties are not fully explored yet. This paper proposes a new 6DoF pose estimation network for parametric shapes in stacked scenarios (ParametricNet). It treats a parametric shape, instead of a part object, as a category. The keypoints of individual instances are learned with point-wise regression and Hough voting scheme, from which specific parameter values are calculated. Then, the template keypoints are obtained based on the computed parameter values and the parametric shape templates. Finally, the 6DoF pose is estimated by least-square fitting between the individual instance's and the template's keypoints & centroid. On the public Siléane dataset, the average of APs of ParametricNet is 96%, compared with 82% for the state-of-the-art method. In addition, a new parametric dataset with four shape templates is constructed, in which the evaluated learning and generalization abilities of ParametricNet outperform the state-of-the-art methods. In particular, for the less symmetric shape, the mAP is improved by over 20%, which is an obvious improvement. Real-world experiments show that our method can grasp parametric shapes with unknown parameter values in stacked scenarios.**

I. Introduction

Parametric techniques are widely used in engineering product design [1]. A parametric shape is a shape template described by a set of shape parameters and constraints [2], [3]. A part object is instantiated from a shape template with a specific parameter value configuration. Thus, a parametric shape means a family of part objects. In robot-based assembly systems, different part objects from the same shape template, e.g., multiple types of nuts, are grasped from different stacked bins to assemble an industrial product. 6DoF object pose estimation (OPE), i.e., 3D translation and 3D rotation, is essential for such vision-guided robot grasping applications. The objective herein is to design a 6DoF OPE network for parametric shapes in stacked scenarios. It is challenging due to part objects' variety, similar appearance, heavy occlusion, and sensor noise.

Most existing 6DoF OPE methods are designed for non-parametric shapes, which can be roughly classified into template-, feature-, and learning-based methods. In template-based methods [4], [5], pre-computed templates were used to scan different points to compute similarity scores for each location. Then, the best match was obtained by comparing these similarity scores. However, the similarity computation deteriorated when objects had severe occlusions. In feature-based methods, the OPE task was formulated as a feature matching problem between the object and its corresponding 3D model. Many feature descriptors were proposed, e.g., VFH [6], LINEMOD [7], and PPF [8], [9]. However, their performance dropped significantly under complex scenarios with similar-looking part objects of the same shape template since they exhibited similar features. With the progress of feature learning on point clouds, e.g., PointNet [10], PointNet++ [11], and PointSIFT [12], the recent learning methods regarded the OPE task as a multitask comprising point-wise instance segmentation and pose regression. They had excellent OPE performance for stacked scenarios, e.g., PPR-Net [13] and OP-Net [14]. However, their generalization abilities were not satisfied due to the non-linearity of the rotation space [15]. Instead, He et al. [16] proposed a keypoint-based OPE network PVN3D, which estimated 6DoF pose by least-squares fitting between the predicted object keypoints and the corresponding CAD model's pre-defined keypoints. The main difference among various keypoint-based learning methods, e.g., CornerNet [17], ExtremeNet [18], and CenterNet [19], was the way that an object's keypoints were defined and detected.

However, if existing keypoint-based OPE learning methods are directly applied to parametric shapes, two challenges will arise. The previous methods treat a part object as a category to pre-define its keypoints. But the number of part objects generated from a shape template is infinite, resulting in category explosion. In addition, such methods are difficult to generalize to the unseen part objects if there are no corre-

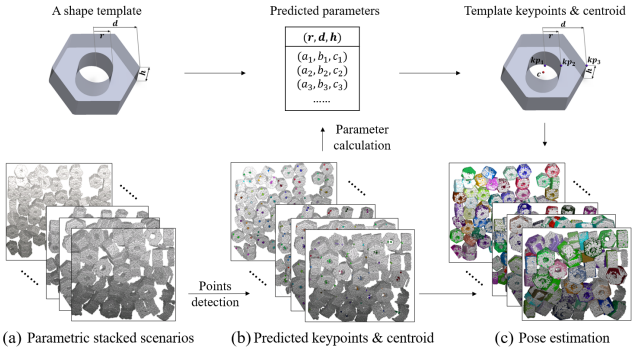


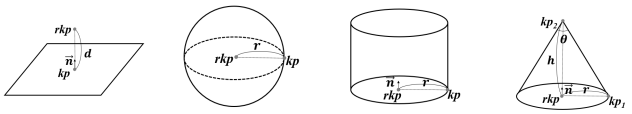
Fig. 1. Overview of ParametricNet, based on geometric unique keypoint learning and Hough voting scheme.

† these authors contribute equally. * corresponding author.

This work is supported by the National Natural Science Foundation of China (Grant No. 61972220, No. 61725204).

Long Zeng, Wei Jie Lv and Xin Yu Zhang are with the Department of Advanced Manufacturing, Shenzhen International Graduate School, Tsinghua University, Shenzhen, China (e-mail: zenglong@sz.tsinghua.edu.cn; lwj19@mails.tsinghua.edu.cn; zhangxy20@mails.tsinghua.edu.cn).

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



(a) kp-in-plane (b) kp-in-sphere (c) kp-in-cylinder (d) kp-in-cone

Fig. 2. Four types for Out-Prim.

sponding CAD models. Second, existing keypoint-definition strategies cannot generate unique keypoints because human-designed parametric shapes are usually symmetric, deteriorating a network's keypoint learning ability.

A new 6DoF pose estimation network for parametric shapes in stacked scenarios, denoted ParametricNet, is proposed in this paper. As Fig. 1(a) shows, the network treats a shape template as a category instead of a part object. For parametric shapes, a template is pre-defined how to compute geometric unique (GU) keypoints based on their driven parameters and symmetry, denoted template keypoints, and generates a family of part objects stacked in bins. Then, as shown in Fig. 1(b), the instance segmentation and keypoints of the individual instance are learned with point-wise regression and Hough voting scheme. From the predicted keypoints & centroid, the specific parameter values can be computed. Finally, as Fig. 1(c) shows, the 6DoF pose can be estimated by least-squares fitting [20] between the instance's and template's keypoints & centroid (dynamically computed on the template with the computed parameter values).

The learning and generalization abilities (i.e., whether testing part objects are seen in training phase) of ParametricNet are the subjects of experiments. For the learning ability, the public Siléane dataset [21] and a new parametric dataset with four shape templates are selected. Compared with the state-of-the-art methods OP-Net [14] and PPR-Net [13], our method outperforms by 14% in the average of APs (average precision) on Siléane dataset and by 7% in the average of mAPs (mean average precision) on parametric dataset. The generalization ability is tested on the parametric dataset only, and our method outperforms PPR-Net by over 9% in the average of mAPs. In addition, our method is also integrated into the real-world robot grasping system. Grasping experiments show that our method can estimate 6DoF poses of unseen part objects generated from a given parametric shape in stacked scenarios correctly and robustly.

In summary, the main contributions of our work are:

- A new 6DoF OPE network with keypoint learning and Hough voting scheme is proposed for parametric shapes.
- A new selection method for GU keypoints considering shape's driven parameters and symmetry is designed.
- A new parametric dataset with four shape templates is constructed and evaluated.

II. Method for Parametric Shapes

A. Keypoint selection

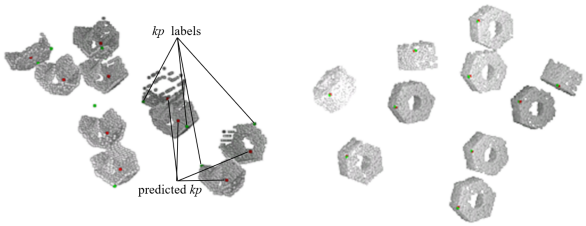
Keypoint definition. All shape parameters can be converted into distance-type parameters. In the shape template, a distance-type parameter is usually defined as the distance from a reference keypoint (rkp) to a primitive. For example, the radius of a cylinder can be defined as the distance from its centroid (i.e., rkp) to the cylindrical surface (i.e., primitive) where the other keypoint (i.e., kp) can be found. All situations are classified into In-Prim and Out-Prim roughly, depending whether rkp is also in the primitive or not. For In-Prim, kp is selected according to the shape of primitive, e.g., the hexagonal plane (Fig. 4(c)). For Out-Prim, four types exist widely in most engineering parts: kp-in-plane, kp-in-sphere, kp-in-cylinder, and kp-in-cone, as Fig. 2 shows.

Given a parameter and its rkp, the computation of the second kp of these four types are carried out as follows.

- kp-in-plane: The distance d is the vertical distance from rkp to the plane primitive, and the vertical foot is kp.
- kp-in-sphere: The radius r is the vertical distance from rkp to any tangent plane of the spherical primitive, and the vertical foot is kp.
- kp-in-cylinder: The radius r is the vertical distance from rkp to any tangent plane of the cylinder primitive, and the vertical foot is kp.
- kp-in-cone: The cone apex angle θ is an angle-type parameter and can be broken into two distance-type parameters r and h. kp_2 is the case of kp-in-cone and kp_1 lying in the plane with rkp is the case of In-Prim.

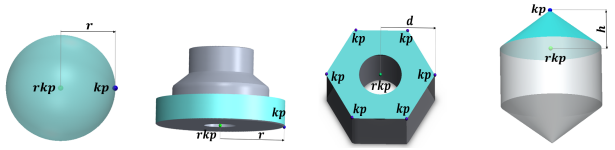
The selection process of keypoints based on driven parameters is as follows: Given the set of distance-type driven parameters $Paras = \{para_i\}_{i=1}^{k}$, first, the centroid c is selected as the current rkp and an empty set $Kp = \{\}$ is created to store keypoints. Then, select the parameters associated with current rkp in $Paras$ to find the second kp, store kp in Kp, and remove the processed parameters from $Paras$. Then, the appropriate known point from Kp is taken as current rkp, and this process is continued until $Paras = \{\}$.

Geometric unique keypoint. Given a parameter and its rkp, from Fig. 2, it is obvious that the second kp is usually not unique, especially for symmetric engineering parts. For keypoint-based learning methods, the ambiguity of keypoint labels will cause confusion in the learning phase. As Fig. 3(a) shows, when the given parameter for a hexagon nut is the outer radius and its rkp is the center of circle in the plane primitive, the kp labels (green points) can be one of the six hexagon vertices since the symmetry, so the predicted keypoints (red points) are chaotic. As shown in Fig. 4, for



(a) No geometric uniqueness (b) With geometric uniqueness

Fig. 3. Performance of geometric unique keypoint learning, where (a) and (b) are the same scene from different view angles.



(a) Spherical (b) Rotational (c) Finite (d) Non-symmetry

Fig. 4. Equivalent keypoints $\Re(kp)$ for (a) $\{\mathbf{R}(\alpha,\beta,\gamma)\cdot kp|\alpha,\beta,\gamma\in\mathbb{R}\}$, (b) $\{\mathbf{R}_{\mathbf{n}}^{\alpha}\cdot kp|\alpha\in\mathbb{R}\}$, (c) $\{kp_i\}_{i=1}^{m}$, and (d) $\{kp\}$, where $\mathbf{R}(\alpha,\beta,\gamma)$ is a rotation matrix with rotation angles α, β, γ around the X, Y, Z axes of a canonical local frame centered at the sphere primitive; $\mathbf{R}_{\mathbf{n}}^{\alpha}$ is a rotation matrix with angle α along the rotational axis $\mathbf{n}$.

a given keypoint kp, the equivalent keypoint set, $\Re(kp)$, has four cases, w.r.t., its associate primitives: spherical-, rotational-, finite-, and non-symmetry.

To define geometric unique keypoints, we select the keypoint with the smallest z coordinate in the camera frame as the keypoint label in the equivalent keypoint set $\Re(kp)$. This scheme has significant geometric meaning: in $\Re(kp)$, the keypoint with the smallest z coordinate is that closest to the camera, which is highly identifiable. So for any object in a scene, the keypoint labels are computed as:

$$kp_{closest} = \{kp|z_{kp} = \min_{p\in\Re(kp)} z_p\}, \tag{1}$$

where z_{kp} is the z coordinate of kp in the camera frame.

If $\Re(kp)$ is finite, $kp_{closest}$ is obtained by an exhaustive comparing method. Otherwise, it is calculated by solving the smallest z coordinate from a parametric equation of a circle or sphere. In particular, if z coordinates are the same, take the keypoint with the smallest y coordinate as $kp_{closest}$, and if z, y coordinates are the same, take the keypoint with the smallest x coordinate as $kp_{closest}$. The network trained with GU keypoints can predict keypoints correctly (Fig. 3(b)).

B. Architecture design

The architecture of ParametricNet is shown in Fig. 5. PointSIFT is adopted as the backbone to extract both global and local features from point cloud, and there are other alternative backbones, e.g., PointNet [10] and PointNet++ [11]. This backbone consumes an unordered point cloud of a stacked scene of size $N_p\times 3$ directly and learns to extract features of size $N_p\times N_f$. It jointly learns the tasks of centroids, keypoints, parameters, and visibilities in a stacked scene of parametric shapes:

$$L = \lambda_{M_C}\cdot L_{M_C} + \lambda_{M_{KP}}\cdot L_{M_{KP}} + \lambda_{M_P}\cdot L_{M_P} + \lambda_{M_V}\cdot L_{M_V}, \tag{2}$$

where λ_{M_C}, $\lambda_{M_{KP}}$, λ_{M_P}, and λ_{M_V} are loss weights to ensure that the four losses in L are roughly equally weighted.

Centroid prediction module M_C. The learned $N_p\times N_f$ features are fed into M_C to regress the point-wise offsets to the centroid of individual instance to which each point belongs. Then, the point-wise predicted centroids of size $N_p\times 3$ is obtained by adding the predicted offsets to the point cloud coordinates. Similar to PPR-Net [13], if points belong to the same individual instance, their predicted centroids will be close to each other in the centroid space. Thus, an unsupervised density-based clustering algorithm, e.g., Mean Shift [22], is applied in the centroid space to achieve instance segmentation, which divides the point cloud into d clusters. Before voting, the radius filter algorithm is applied in each cluster to remove unreliable predicted centroids without changing the clusters' number. Finally, each cluster votes for the final predicted centroids of size $d\times 3$ by averaging the remaining predicted centroids. The loss L_{M_C} for the module M_C is L_2 loss between predicted centroids and centroid labels of all points, which is normalized by the number of points:

$$L_{M_C} = \frac{1}{N_p}\sum_{i=1}^{N_p}\|c_i-\hat{c}_i\|^2, \tag{3}$$

where $\hat{c}_i$ and c_i are point-wise predicted centroid and centroid label ($\hat{c}_i$, $c_i\in\mathbb{R}^3$).

Keypoint prediction module M_{KP}. Similar to M_C, the learned $N_p\times N_f$ features are passed into MLPs to regress the offsets to m keypoints of the individual instance to which each point belongs. Then the point-wise predicted keypoints of size $N_p\times 3m$ are obtained. With the instance segmentation result from M_C, predicted keypoints are divided into $m\times d$ clusters in the keypoint space. In addition, the same radius filter algorithm is applied in each cluster to filter the unreliable predicted keypoints. Then, each cluster votes for the final predicted keypoints of the individual instances of size $d\times 3m$. The loss $L_{M_{KP}}$ for the module M_{KP} is L_2 loss:

$$L_{M_{KP}} = \frac{1}{N_p}\sum_{i=1}^{N_p}\left\|kp_i-\hat{kp}_i\right\|^2, \tag{4}$$

where $\hat{kp}_i$ and kp_i are the point-wise concatenated vector of predicted keypoints and keypoint labels ($\hat{kp}_i$, $kp_i\in\mathbb{R}^{3m}$).

Parameter calculation module M_P. With point-wise predicted centroids from M_C and point-wise predicted keypoints from M_{KP}, the point-wise predicted parameters of size $N_p\times k$ can be computed according to the shape template with k parameters. With instance segmentation result, the point-wise predicted parameters are divided into $k\times d$ clusters in the parameter space. Then, each cluster votes for the final predicted parameters of the individual instances of size $d\times k$. With the predicted parameters, the template keypoints & centroid of size $d\times 3(m+1)$ are recovered in the canonical local frame for pose fitting. To ensure the reliablility of the module M_P, the loss L_{M_P} is L_1 loss as follows:

$$L_{M_P} = \frac{1}{N_p}\sum_{i=1}^{N_p}\|para_i-\hat{para}_i\|, \tag{5}$$

where $\hat{para}_i$ and $para_i$ are the point-wise predicted parameters and parameter labels ($\hat{para}_i$, $para_i\in\mathbb{R}^k$).

Pose fitting module M_F. The shape template is predefined within a canonical local frame. Once the individual instance's and the template's keypoints & centroid are prepared, the module M_F computes the 6DoF pose ($\mathbf{R}$ and $\mathbf{t}$) of each individual instance with a least-squares fitting algorithm [20], by minimizing the following function:

$$L_{least-squares} = \sum_{i=0}^{m}\|p_i-(\mathbf{R}\cdot tp_i+\mathbf{t})\|^2, \tag{6}$$

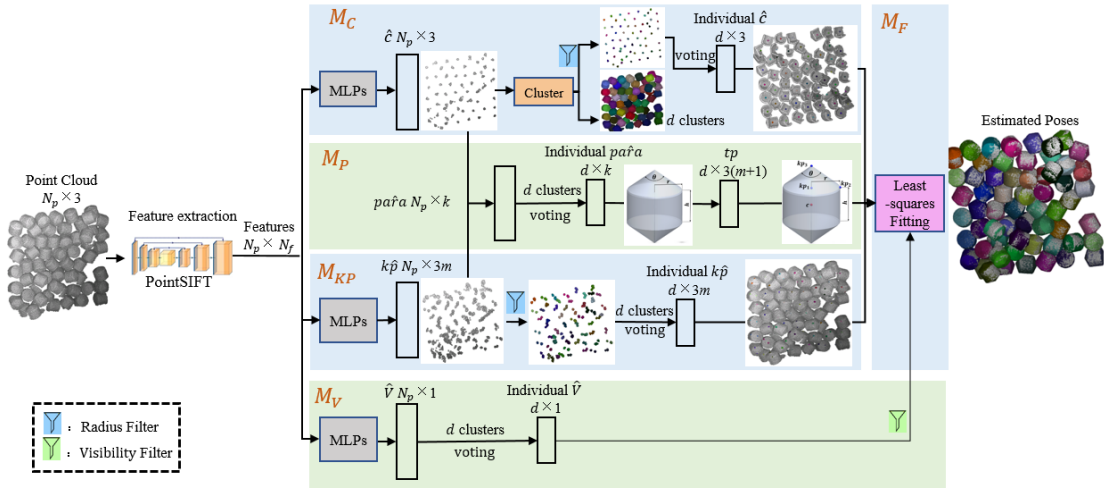


Fig. 5. ParametricNet architecture, where $\hat{c}$, $p\hat{a}ra$, $k\hat{p}$, and $\hat{V}$ represent predicted centroids, parameters, keypoints, and visibilities, respectively; k is the number of parameters; *d clusters* is the instance segmentation result to assign the point-wise prediction to d individual instances for voting.

where tp_i is the template keypoints & centroid, and p_i is the predicted keypoints & centroid of the individual instance ($tp_i, p_i \in \mathbb{R}^3$).

The spatial distribution of an individual instance's predicted keypoints may vary due to different camera angles. Fortunately, they can be summarized into two chirality structures at most; that is, two point sets of different chirality structures cannot coincide with each other only by rotation and translation transformation. If the predicted keypoints form two chirality structures, two template keypoint sets of opposite chirality structure are selected to fit a pose, respectively, and the one with less fitting error is selected.

Visibility prediction module M_V. In a seriously stacked scene, an individual instance with low visibility (i.e., high occlusion rate) is probably ungraspable. Visibility V can be simply approximated by N_j / N_{max}, where N_j is the number of points of jth individual instance and N_{max} is the number of points of the individual instance with the most points in the scene. The point-wise visibility is regressed and divided into d clusters in the visibility space. Then, each cluster votes for the predicted visibilities of individual instances of size $d \times 1$, helpful for filtering the ungraspable individual instances with visibilities that are less than a visibility threshold T_v. The loss L_{M_V} for the module M_V is L_1 loss:

$$L_{M_V} = \frac{1}{N_p} \sum_{i=1}^{N_p} \left\| V_i - \hat{V}_i \right\|, \tag{7}$$

where $\hat{V}_i$ and V_i are the point-wise predicted visibility and the visibility label (V_i, $\hat{V}_i \in [0, 1]$).

C. Training and implementation

In the implementation, 16,384 points are sampled for each scene using furthest point sampling and each point contains only camera coordinate information. PointSIFT is chosen as the feature extraction backbone and N_f = 128, T_v = 0.45, $\lambda_{M_C} = 3m \times \lambda_{M_{KP}} = 200$, and $\lambda_{M_V} = k \times \lambda_{M_P} = 50$ are set by default, where m is the number of the keypoints and k is the number of parameters. ParametricNet is prototyped with TensorFlow 1.5 on a GeForce RTX2080Ti and optimized by the Adam optimizer with batch size of 6 and initial learning rate of 0.001. The learning rate decays every 200,000 steps by a factor of 0.5. On a GeForce RTX2080Ti, the forward-pass time of ParametricNet for a single scene is 250 ms.

TABLE I

STATISTICS OF SILÉANE [21] AND OUR PARAMETRIC DATASET, WHERE L AND G DENOTE THE NUMBER OF TESTING PART OBJECTS FOR LEARNING AND GENERALIZATION ABILITIES. ONE CYCLE MEANS OBJECTS ARE DROPPED ONE BY ONE UNTIL DROP LIMIT IS REACHED.

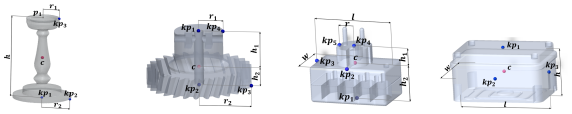
Dataset	Object	# Number		# Drop limit	# Training cycles	# Test cycles
		L	G			
Siléane dataset	C.stick	1	-	60	500	2
	Gear	1	-	60	500	2
	T-Less20	1	-	99	500	2
	T-Less29	1	-	79	500	2
Parametric dataset	TN06	64	64	60	29	1
	TN16	64	64	60	29	1
	TN42	64	64	60	29	1
	TN34	16	16	60	29	1

III. EXPERIMENTS

A. Datasets and evaluation metrics

The performance of ParametricNet is evaluated on two benchmark datasets. Their contents and formats are summarized in Table I.

Siléane dataset. Considering the symmetry types (Fig. 4), four of eight objects in the public Siléane dataset, i.e., Candlestick (C.stick), Gear, T-Less20, and T-Less29, are selected. To apply ParametricNet to non-parametric shapes, a non-parametric object can be converted into a part object from a shape template consisting of several simple bounding primitives. Taking the T-Less20 in Fig. 6(c) as an example, it can be regarded as a part object with specific parameter values, generated from a shape template bounded by a cuboid and a cylinder. More examples are shown in Fig. 6.



(a) C.stick (b) Gear (c) T-Less20 (d) T-Less29

Fig. 6. Siléane dataset and its shape parameters and keypoints.

Parametric dataset. A parametric dataset of stacked scenarios with four shape templates selected from Zeng's parametric database [1] is constructed, as shown in Fig. 7. Each parameter of shape templates is uniformly sampled with four values. Then, the results of their combinations are used to generate part objects. For each part object, it is piled into a bin one by one to generate a cycle of stacked scenes until *drop limit* is reached, similar to the Siléane dataset.

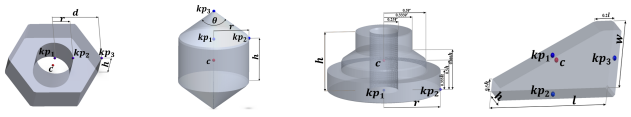
To test the learning and generalization abilities of ParametricNet, two testing datasets, i.e., the L-dataset and G-dataset, are constructed. In the L-dataset, the parameters of the part objects are the same as those of the training set. In the G-dataset, the part objects' parameters are different from those of the training set, i.e., by approximately a 2%–37% difference in each parameter value.

The pose estimation metric proposed by Brégier [21] is adopted. The individual instances with visibilities larger than 50% are relevant for the retrieval. The overall performance of pose estimation is measured by AP, which is the area under the precision-recall curve. For a parametric shape, the evaluation metric adopts mAP (mean average precision), which is the average of the APs of all part objects.

B. Evaluation on Siléane and parametric datasets

Some qualitative examples of ParametricNet on both the Siléane and parametric datasets are shown in Fig. 8. The quantitative comparison and evaluation are given below.

Evaluation on Siléane dataset. To facilitate comparison with other state-of-the-art OPE methods, ParametricNet is tested on all noisy data from the Siléane dataset while being trained on the original synthetic data from the Fraunhofer IPA Bin-Picking dataset [23]. As shown in Table II, the average of the APs of ParametricNet is 96% (the average of 97%, 100%, 92%, and 94%), compared with 82% (the average of 97%, 84%, 88%, and 58%) for the state-of-the-art method OP-Net$_1$ [14]. Significantly, PPR-Net [13] and OP-Net$_1$ are trained by adding random noise, while ParametricNet is trained entirely on the original synthetic data without noise. Moreover, ParametricNet without iterative refinement is better than those methods with iterative refinement, which shows ParametricNet has strong learning ability.



(a) TN06 (b) TN16 (c) TN34 (d) TN42

Fig. 7. Parametric dataset shape parameters and keypoints.

TABLE II

EVALUATION ON SILÉANE DATASET, WHERE OP-NET$_1$ IS OP-NET WITH *Lori*$_1$ AND *PP* AND OP-NET$_2$ IS OP-NET WITH *Lori*$_2$ AND *PP* [14].

Object	C.stick	Gear	T-Less20	T-Less29
PPF [8], [21]	0.16	0.62	0.20	0.19
PPF PP [8], [21]	0.22	0.63	0.23	0.23
LINEMOD+ [24], [21]	0.38	0.44	0.25	0.20
LINEMOD+ PP [24], [21]	0.49	0.50	0.31	0.26
Sock et al. [25]	0.64	–	–	–
PPR-Net [13]	0.91	–	0.81	–
PPR-Net with ICP [13]	0.95	–	0.85	–
OP-Net$_1$ [14]	**0.97**	0.84	0.88	0.58
OP-Net$_2$ [14]	0.96	0.60	0.58	0.39
ParametricNet	**0.97**	**1.00**	**0.92**	**0.94**

Learning ability evaluation on parametric dataset. PPR-Net, the winner method of the "Object Pose Estimation Challenge for Bin-Picking" at IROS 2019 [14], is selected as a baseline. Both ParametricNet and PPR-Net learn all part objects in the training set and are tested by the L-dataset where part objects' parameters are the same as those of the training set. Table III shows that the learning ability of ParametricNet outperforms PPR-Net on all shape templates by 7% in the average of mAPs, especially on the finite symmetric template TN06 and the non-symmetric template TN42. The reason for the high APs of the revolutional symmetric templates is that their equivalent poses are infinite, according to Brégier's AP computation method [21], e.g., TN16 and TN34.

TABLE III

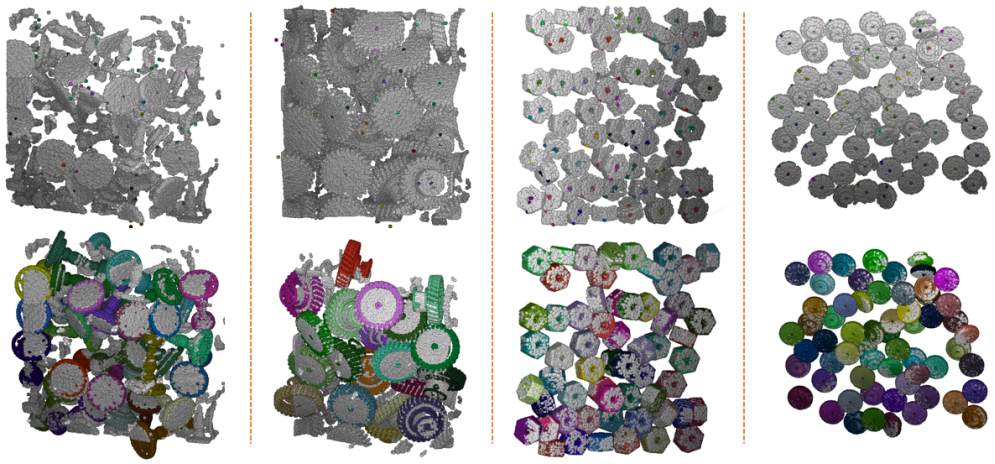
LEARNING ABILITY EVALUATION ON PARAMETRICNET DATASET.

Object	TN06	TN16	TN34	TN42
PPR-Net	0.80	0.99	**1.00**	0.39
ParametricNet	**0.94**	**1.00**	**1.00**	**0.52**

Generalization ability evaluation on parametric dataset. Both ParametricNet and PPR-Net learn all part objects in the training set and are tested by the G-dataset where part objects' parameters are different from those of the training set (by approximately a 2%–37% difference in each parameter value). From Table IV, it can be seen that the generalization ability of ParametricNet is 9.5% better than that of the PPR-Net in the average of the mAPs of the four parametric shapes. In addition, the following is noteworthy.

- The generalization ability of ParametricNet is almost the same as its learning ability, while that of PPR-Net is deteriorated to a certain level, especially for TN42.
- With the same generalization test set (G-dataset), we downsample the number of learned part objects in the training set by one-third (1/3) and one-fifth (1/5), and the generalization ability of ParametricNet is better than PPR-Net all the time. In particular, for the finite symmetric template TN06 and the non-symmetric template TN42, the mAP of ParametricNet learning 1/5 part objects is even better than that of PPR-Net learning all part objects.

Ablation study. Further experiment results for Paramet-



(a) Candlestick (b) Gear (c) TN06 (d) TN34

Fig. 8. Performance on Siléane dataset and parametric dataset, in which the first row represents the predicted keypoints and centroid and the second row represents the estimated poses overlapping the point cloud.

TABLE IV

GENERALIZATION ABILITY EVALUATION ON PARAMETRIC DATASET.

Object	Learn all		Learn 1/3		Learn 1/5	
	PPR-Net	Ours	PPR-Net	Ours	PPR-Net	Ours
TN06	0.79	**0.93**	0.78	**0.86**	0.77	**0.86**
TN16	0.99	**1.00**	0.94	**0.98**	0.56	**0.63**
TN34	**1.00**	**1.00**	0.96	**1.00**	0.83	**0.86**
TN42	0.28	**0.51**	0.22	**0.41**	0.18	**0.39**

TABLE V

ABLATION STUDY FOR GENERALIZATION ABILITY EVALUATION. GU, GEOMETRIC UNIQUENESS; RF, RADIUS FILTER; W/O, WITHOUT.

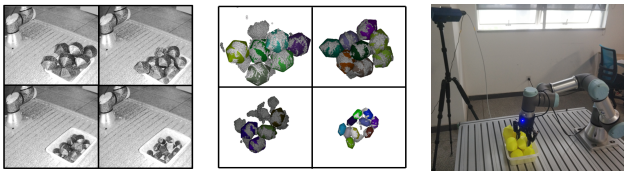
Object	TN06	TN16	TN34	TN42
ParametricNet (w/o GU)	0.03	0.01	0.82	0.33
ParametricNet (w/o RF)	**0.93**	0.99	**1.00**	0.49
ParametricNet (w/o M_V)	0.85	0.99	0.99	0.36
ParametricNet	**0.93**	**1.00**	**1.00**	**0.51**

ricNet are shown in Table V, all of which learn all part objects and are evaluated on the G-dataset. ParametricNet (w/o GU) learns keypoints without geometric uniqueness, which decreases its mAPs dramatically. ParametricNet (w/o RF) means that radius filter is not applied in M_C and M_{KP}, which slightly worsens the performance. ParametricNet (w/o M_V) removes the M_V module and the results show that M_V is helpful to filter the ungraspable individual instances.

C. Real-world Experiment

To demonstrate the effectiveness of ParametricNet in real-world scenarios, it was integrated into a robot grasping system. A total of 30,720 synthetic scenes were annotated with noise, in which 28,800 scenes (30 cycles, *drop limit* 16, *part object number* 60) were used for training and the other 1,920 scenes (30 cycles, *drop limit* 16, *part object number* 4 with unseen parameter values) were used for generalization testing. The datasets were generated in a synthetic platform [13], in which the virtual camera had the same configuration of an actual Ensenso N35 camera. Quantitative evaluation showed that ParametricNet achieved AP values of 0.99, 0.99, 0.98, and 0.98 for the four testing part objects, respectively.

In the grasping experiment, the four unseen part objects in the test set were printed and put into a box randomly; then, the point clouds were captured by the aforementioned fixed Ensenso N35 range camera, as shown in Fig. 9. The stacked scenes for the four part objects were built as shown in Fig. 9(a). The scene point cloud was then obtained by background subtraction and furthest point sampling, which was fed into ParametricNet to estimate their 6DoF poses, as shown in Fig. 9(b). As shown in Fig. 9(c), ParametricNet was evaluated in the grasping experiment and the robot was able to pick and place all graspable individual instances in each stacked scene, including those with significant occlusion.



(a) Stacked scenes (b) Estimated poses (c) Robotic grasp

Fig. 9. Grasping experiment on parametric shapes in stacked scenes, where parameters $r/h/\theta$ (mm/mm/rad) of four part objects are 16/34/1.68, 22/34/1.68, 28/34/1.68, and 34/34/1.68.

IV. CONCLUSIONS

In this paper, ParametricNet, a new 6DoF OPE network, is designed for parametric shapes. We design a new method to define geometric unique keypoints based on driven parameters and shape symmetry. In addition, a new parametric dataset with four shape templates is constructed. The experiment results demonstrate the outstanding performance of ParametricNet, compared with other state-of-the-art 6DoF OPE methods. We will release the code and dataset soon (`https://github.com/lvwj19/ParametricNet`).

REFERENCES

[1] L. Zeng, Z.-k. Dong, J.-y. Yu, J. Hong, and H.-y. Wang, "Sketch-based retrieval and instantiation of parametric parts," *Computer-Aided Design*, vol. 113, pp. 82–95, 2019.

[2] V. Shapiro and D. L. Vossler, "What is a parametric family of solids?" in *Symposium on Solid Modeling and Applications*, 1995, pp. 43–54.

[3] C. M. Hoffmann and K. J. Kim, "Towards valid parametric CAD models," *Computer-Aided Design*, vol. 33, no. 1, pp. 81–90, 2001.

[4] 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 *ICCV*, 2011, pp. 858–865.

[5] Y. Konishi, K. Hattori, and M. Hashimoto, "Real-time 6D object pose estimation on CPU," in *IEEE/RSJ IROS*, 2019, pp. 3451–3458.

[6] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *IEEE/RSJ IROS*, 2010, pp. 2155–2162.

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

[8] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *IEEE Computer Society Conference on CVPR*, 2010, pp. 998–1005.

[9] H. Dong, D. K. Prasad, and I. M. Chen, "Object pose estimation via pruned hough forest with combined split schemes for robotic grasp," *IEEE Transactions on Automation Science and Engineering*, pp. 1–8, 2020.

[10] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on CVPR*, 2017, pp. 77–85.

[11] C. R. Qi, L. Yi, H. Su, and L. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, pp. 5099–5108, 2017.

[12] M. Jiang, Y. Wu, and C. Lu, "PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation," *arXiv preprint arXiv:1807.00652*, 2018.

[13] Z. Dong, S. Liu, T. Zhou, H. Cheng, L. Zeng, X. Yu, and H. Liu, "Ppr-net: point-wise pose regression network for instance segmentation and 6d pose estimation in bin-picking scenarios," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1773–1780.

[14] K. Kleeberger and M. F. Huber, "Single shot 6d object pose estimation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6239–6245.

[15] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *IEEE/CVF Conference on CVPR*, 2019, pp. 4556–4565.

[16] 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 *IEEE Conference on CVPR*, 2020, pp. 11 629–11 638.

[17] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," *International Journal of Computer Vision*, vol. 128, no. 3, pp. 642–656, 2020.

[18] X. Y. Zhou, J. C. Zhuo, and P. Krahenbuhl, "Bottom-up object detection by grouping extreme and center points," *2019 IEEE/CVF CVPR*, pp. 850–859, 2019.

[19] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6569–6578.

[20] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3D point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5, pp. 698–700, 1987.

[21] R. Brégier, 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 *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017, pp. 2209–2218.

[22] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[23] K. Kleeberger, C. Landgraf, and M. F. Huber, "Large-scale 6D object pose estimation dataset for industrial bin-picking," in *2019 IEEE/RSJ IROS*, 2019, pp. 2573–2578.

[24] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Asian conference on computer vision*. Springer, 2012, pp. 548–562.

[25] J. Sock, K. I. Kim, C. Sahin, and T. Kim, "Multi-task deep networks for depth-based 6d object pose and joint registration in crowd scenarios." BMVC, 2018.