FS-Net: Fast Shape-based Network for Category-Level 6D Object Pose Estimation with Decoupled Rotation Mechanism

Wei Chen [1] Xi Jia[1] Hyung Jin Chang[1] Jinming Duan[1] Linlin Shen[2] Aleš Leonardis[1]

[1] School of Computer Science, University of Birmingham

[2] Computer Vision Institute, College of Computer Science and Software Engineering, Shenzhen University

✉wxc795@cs.bham.ac.uk

Abstract

In this paper, we focus on category-level 6D pose and size estimation from monocular RGB-D image. Previous methods suffer from inefficient category-level pose feature extraction which leads to low accuracy and inference speed. To tackle this problem, we propose a fast shape-based network (FS-Net) with efficient category-level feature extraction for 6D pose estimation. First, we design an orientation aware autoencoder with 3D graph convolution for latent feature extraction. The learned latent feature is insensitive to point shift and object size thanks to the shift and scale-invariance properties of the 3D graph convolution. Then, to efficiently decode category-level rotation information from the latent feature, we propose a novel decoupled rotation mechanism that employs two decoders to complementarily access the rotation information. Meanwhile, we estimate translation and size by two residuals, which are the difference between the mean of object points and ground truth translation, and the difference between the mean size of the category and ground truth size, respectively. Finally, to increase the generalization ability of FS-Net, we propose an online box-cage based 3D deformation mechanism to augment the training data. Extensive experiments on two benchmark datasets show that the proposed method achieves state-of-the-art performance in both category- and instance-level 6D object pose estimation. Especially in category-level pose estimation, without extra synthetic data, our method outperforms existing methods by 6.3% *on the NOCS-REAL dataset* [1].

1. Introduction

Estimating 6D object pose plays an essential role in many computer vision tasks such as augmented reality [20, 21], virtual reality [2], and smart robotic arm [47, 36]. For instance-level 6D pose estimation, in which training set and test set contain the same objects, huge progress has been made in recent years [42, 29, 22, 16, 10].

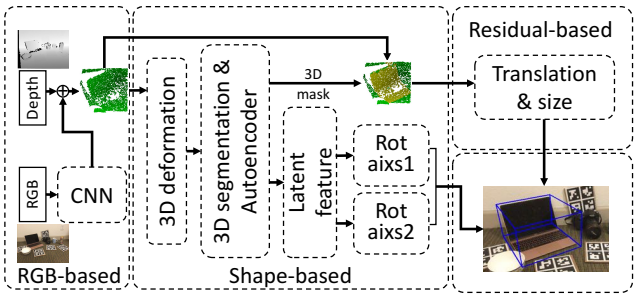

Figure 1. **Semantic illustration of FS-Net.** We use different networks for different tasks. The RGB-based network is used for 2D object detection, and the shape-based 3D graph convolution autoencoder is used for 3D segmentation and rotation estimation. The residual-based network is used for translation and size estimation with segmented points.

However, category-level 6D pose estimation remains challenging as the object shape and color are various in the same category. Existing methods addressed this problem by mapping the different objects in the same category into a uniform model via RGB feature or RGB-D fusion feature. For example, Wang *et al.* [41] trained a modified Mask R-CNN [9] to predict the normalized object coordinate space (NOCS) map of different objects based on RGB feature, and then computed the pose with observed depth and NOCS map by Umeyama algorithm [37]. Chen *et al.* [4] proposed to learn a canonical shape space (CASS) to tackle intra-class shape variations with RGB-D fusion feature [40]. Tian *et al.* [35] trained a network to predict the NOCS map of different objects, with the uniform shape prior learned from a shape collection, and RGB-D fusion feature [40].

Although these methods achieved state-of-the-art perfor-

[1]Paper code https://github.com/DC1991/FS-Net

mance, there are still two issues. Firstly, the benefits of using RGB feature or RGB-D fusion feature for category-level pose estimation are still questionable. In [38], Vlach et al. showed that people focus more on shape than color when categorizing objects, as different objects in the same category have very different colors but stable shapes (shown in Figure 3). Thereby the use of RGB feature for category-level pose estimation may lead to low performance due to huge color variation in the test scene. For this issue, to alleviate the color variation, we merely use the RGB feature for 2D detection, while using the shape feature learned with point cloud extracted from depth image for category-level pose estimation.

Secondly, learning a representative uniform shape requires a large amount of training data; therefore, the performance of these methods is not guaranteed with limited training examples. To overcome this issue, we propose a 3D graph convolution (3DGC) autoencoder [19] to effectively learn the category-level pose feature via observed points reconstruction of different objects instead of uniform shape mapping. We further propose an online box-cage based 3D data augmentation mechanism to reduce the dependencies of labeled data.

In this paper, the newly proposed FS-Net consists of three parts: 2D detection, 3D segmentation & rotation estimation, and translation & size estimation. In 2D detection part, we use the YOLOv3 [31] to detect the object bounding box for coarse object points obtainment [6]. Then in the 3D segmentation & rotation estimation part, we design a 3DGC autoencoder to perform segmentation and observed points reconstruction jointly. The autoencoder encodes orientation information in the latent feature. Then we propose the decoupled rotation mechanism that uses two decoders to decode the category-level rotation information. For translation and size estimation, since they are all point coordinates related, we design a coordinate residual estimation network based on PointNet [27] to estimate the translation residual and size residuals. To further increase the generalization ability of FS-Net, we use the proposed online 3D deformation for data augmentation. To summarize, the main contributions of this paper are as follows:

- We propose a fast shape-based network to estimate category-level 6D object size and pose. Due to the efficient category-level pose feature extraction, the framework runs at 20 FPS on a GTX 1080 Ti GPU.
- We propose a 3DGC autoencoder to reconstruct the observed points for latent orientation feature learning. Then we design a decoupled rotation mechanism to fully decode the orientation information. This decoupled mechanism allows us to naturally handle the circle symmetry object (in Section 3.3).
- Based-on the shape similarity of intra-class objects, we propose a novel box-cage based 3D deformation mechanism to augment the training data. With this mechanism, the pose accuracy of FS-Net is improved by 7.7%.

2. Related Works

2.1. Instance-Level Pose Estimation

In instance-level pose estimation, a known 3D object model is usually available for training and testing. Based on the 3D model, instance-level pose estimation can be roughly divided into three types: template matching based, correspondences-based, and voting-based methods. Template matching methods [11, 30, 22] aligned the template to the observed image or depth map via hand-crafted or deep learning feature descriptors. As they need the 3D object model to generate the template pool, their applications in category-level 6D pose estimation are limited. Correspondences-based methods trained their model to establish 2D-3D correspondences [29, 30, 24] or 3D-3D correspondences [6, 5]. Then they solved perspective-n-point and SVD problem with 2D-3D and 3D-3D correspondences [14], respectively. Some methods [5, 1] also used these correspondences to generate voting candidates, and then used RANSAC [8] algorithm for selecting the best candidate. However, the generation of canonical 3D keypoints is based on the known 3D object model that is not available when predicting the category-level pose.

2.2. Category-Level Pose Estimation

Compared to instance-level, the major challenge of category-level pose estimation is the intra-class object variation, including shape and color variation. To handle the object variation problem, [41] proposed to map the different objects in the same category to a NOCS map. Then they used semantic segmentation to access the observed points cloud with known camera parameters. The 6D pose and size are calculated by the Umeyama algorithm [37] with the NOCS map and the observed points. Shape-Prior [35] adopted similar method with [41], but both extra shape prior knowledge and dense-fusion feature [40], instead of RGB feature, are used. CASS [4] estimated the 6D pose via the learning of a canonical shape space with dense-fusion feature [40]. Since the RGB feature is sensitive to color variation, the performance of their methods in category-level pose estimation is limited. In contrast, our method is shape feature-based which is robust for this task.

2.3. 3D Data Augmentation

In 3D object detection tasks [6, 26, 32, 5], online data augmentation techniques such as translation, random flipping, shifting, scaling, and rotation are applied to original

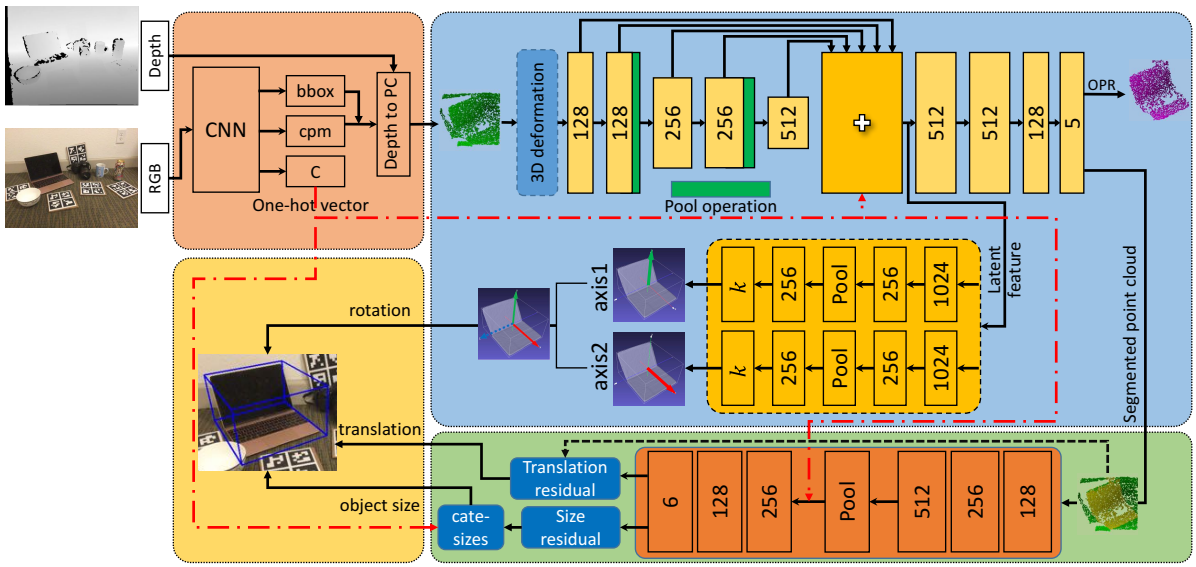

Figure 2. **Architecture of FS-Net.** The input of FS-Net is an RGB-D image. For RGB channels, we use a 2D detector to detect the object 2D location, category label 'C' (used as a one-hot feature for next tasks), and class probability map (cpm) (generate the 3D sphere center via maximum probability location and camera parameters). With this information and depth channel, the points in a compact 3D sphere are generated. Given the points in the 3D sphere, we first use the proposed 3D deformation mechanism for data augmentation. After that, we use a shape-based 3DGC autoencoder to perform observed points reconstruction (OPR), as well as point cloud segmentation, for orientation latent feature learning. Then we decode the rotation information into two perpendicular vectors from the latent feature. Finally, we use a residual estimation network to predict the translation and size residuals. 'cate-sizes' denotes the pre-calculated average sizes of different categories, 'k' is the rotation vector dimension, and the hollow '+' means feature concatenation.

point clouds for training data augmentation. However, these operations cannot change the shape property of the object. Simply adopting these operations on point clouds is not able to handle the shape variation problem in the 3D task. To address this, [7] proposed part-aware augmentation which operates on the semantic parts of the 3D object with five manipulations: dropout, swap, mix, sparing, and noise injection. However, how to decide the semantic parts are ambiguous. In contrast, we propose a box-cage based 3D data augmentation mechanism which can generate the various shape variants (shown in Figure 5) and avoid semantic parts decision procedure.

3. Proposed Method

In this section, we describe the detailed architecture of FS-Net shown in Figure 2. Firstly, we use the YOLOv3 to detect the object location with RGB input. Secondly, we use 3DGC autoencoder to perform 3D segmentation and observed points reconstruction, the latent feature can learn orientation information through the process. Then we propose a novel decoupled rotation mechanism for decoding orientation information. Thirdly, we use PointNet [27] to estimate the translation and object size. Finally, to increase the generalization ability of FS-Net, we propose the box-cage based 3D deformation mechanism.

Figure 3. **Stable shape and various color.** Top row: three bowl instances randomly chosen from the NOCS-REAL dataset. Bottom row: three bowl instances randomly cropped from the internet image search results (using the keyword 'bowl'). The color is varied, while the shape is relatively stable.

3.1. Object Detection

Following [6], we train a YOLOv3 [31] to fast detect the object bounding box in RGB images, and output class (category) labels. Then we adopt the 3D sphere to locate the point cloud of the target object quickly. With these techniques, the 2D detection part provides a compact 3D learning space for the following tasks. Different from other category-level 6D object pose estimation methods that need

semantic segmentation masks, we only need object bounding boxes. Since object detection is faster than semantic segmentation [31, 9], the detection speed of our method is faster than previous methods.

3.2. Shape-Based Network

The output points of object detection contain both object and background points. To access the points that belong to the target object and calculate the rotation of the object, we need a network that performs two tasks: 3D segmentation and rotation estimation.

Although there are many network architectures that directly process point cloud [27, 28, 46], most of the architectures calculate on point coordinates, which means their networks are sensitive to point clouds shift and size variation [19]. This decreases the pose estimation accuracy.

To tackle the point clouds shift, Frustum-PointNet [26] and G2L-Net [6] employed the estimated translation to align the segmented point clouds to local coordinate space. However, their methods cannot handle the intra-class size variation.

To solve the point clouds shift and size variation problem, in this paper, we propose a 3DGC autoencoder to extract the point cloud shape feature for segmentation and rotation estimation. 3DGC is designed for point cloud classification and object part segmentation; our work shows that 3DGC can also be used for category-level 6D pose estimation task.

3.2.1 3D Graph Convolution

3DGC kernel consists of m unit vectors. The m kernel vectors are applied to the n vectors generated by the center point with its n-nearest neighbors. Then, the convolution value is the sum of cosine similarity between kernel vectors and the n-nearest vectors. In a 2D convolution network, the trained network learned a weighted kernel, which has a higher response with a matched RGB value, while the 3DGC network learned the orientations of the m vectors in the kernel. The weighted 3DGC kernel has a higher response with a matched 3D pattern which is defined by the center point with its n-nearest neighbors. For more details, please refer to [19].

3.2.2 Rotation-Aware Autoencoder

Based on the 3DGC, we design an autoencoder for the estimation of category-level object rotation. To extract the latent rotation feature, we train the autoencoder to reconstruct the observed points transformed from the observed depth map of the object. There are several advantages to this strategy: 1) the reconstruction of observed points is view-based and symmetry invariant [33, 34], 2) the reconstruction of observed points is easier than that of a complete object model (shown in Table 2), and 3) more representative orientation feature can be learned (shown in Table 1).

In [33, 34], the authors also reconstructed the input images to observed views. However, the input and output of their models are 2D images that are different to our 3D point cloud input and output. Furthermore, our network architecture is also different from theirs.

We utilize Chamfer Distance to train the autoencoder, the reconstruction loss function $\mathcal{L}_{rec}$ is defined as

$$\mathcal{L}_{rec} = \sum_{x_i \in M_c} \min_{\hat{x}_i \in \hat{M}_c} \|x_i - \hat{x}_i\|_2^2 + \sum_{\hat{x}_i \in \hat{M}_c} \min_{x_i \in M_c} \|x_i - \hat{x}_i\|_2^2, \tag{1}$$

where M_c and $\hat{M}_c$ denote the ground truth point cloud and reconstructed point cloud, respectively. x_i and $\hat{x}_i$ are the points in M_c and $\hat{M}_c$. With the help of 3D segmentation mask, we only use the features extracted from the observed object points for reconstruction.

After the network convergence, the encoder learned the rotation-aware latent feature. Since the 3DGC is scale and shift invariant, the observed points reconstruction enforces the autoencoder to learn the scale and shift invariant orientation feature under corresponding rotation. In the next subsection, we will describe how we decode rotation information from this latent feature.

3.3. Decoupled Rotation Estimation

Given the latent feature which contains rotation information, our task is to decode the category-level rotation feature. To achieve this, we utilize two decoders to extract the rotation information in a decoupled fashion. The two decoders decode the rotation information into two perpendicular vectors under corresponding rotation. These two vectors can represent rotation information completely (shown in Figure 4).

Since the two vectors are orthogonal, the decoded rotation information related to them is independent; we can use one of them to recover part rotation information of the object. For example, in Figure 8, we use the green vector axis to recover the pose. We can see that the green boxes and blue boxes are aligned well in the recovered axis.

Each decoder only needs to extract the orientation information along corresponding vector which is easier than the estimation of the complete rotation. The loss function is based on cosine similarity that defined as

$$\mathcal{L}_{rot} = \frac{\langle \hat{\mathbf{v}}_1, \mathbf{v}_1 \rangle}{\|\hat{\mathbf{v}}_1\| \|\mathbf{v}_1\|} + \lambda_r \frac{\langle \hat{\mathbf{v}}_2, \mathbf{v}_2 \rangle}{\|\hat{\mathbf{v}}_2\| \|\mathbf{v}_2\|}, \tag{2}$$

where $\hat{\mathbf{v}}_1$ and $\hat{\mathbf{v}}_2$ are the predicted vectors. $\mathbf{v}_1$ and $\mathbf{v}_2$ are the ground truth, and λ_r is the balance parameter.

The balance parameter λ_r makes our network easy to handle circular symmetry object such as bottle, and for such circular symmetry object, the red vector is not necessary

(shown in Figure 4). Without loss of generality, we assume that the green vector is along the symmetry axis; then we set λ_r as zero to handle the circular symmetry objects. For other types of symmetric objects, we can employ the rotation mapping function used in [25, 35] to map the relevant rotation matrices to a unique one (see Appendix for details). Please note our decoupled rotation is quite different to the rotation representation proposed in [45]. They took the first two columns from a rotation matrix as the new representation, which has no geometric meaning. In contrast, our representation is defined based on the shape of the target object, and our representation can avoid the discontinuity issue mentioned in [45, 25].

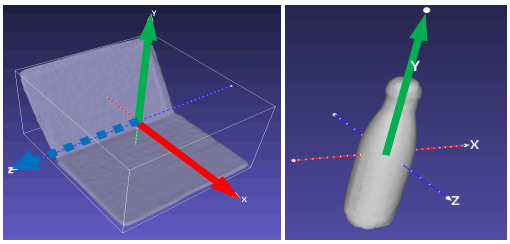

Figure 4. **Rotation represented by vectors.** Left: The object rotation can be represented by two perpendicular vectors (green vector and red vector); Right: For circular symmetry object like the bottle, only the green vector matters.

3.4. Residual Prediction Network

As both translation and object size are related to points coordinates, inspired by [26, 6], we train a tiny PointNet [27] that takes segmented point cloud as input. More concretely, the PointNet performs two related tasks: 1) estimating the residual between the translation ground truth and the mean value of the segmented point cloud; 2) estimating the residual between object size and the mean category size.

For size residual, we pre-calculate the mean size $[\overline{x}, \overline{y}, \overline{z}]^T$ of each category by

$$\begin{bmatrix} \overline{x} \\ \overline{y} \\ \overline{z} \end{bmatrix} = \frac{1}{N} \sum_{i=1}^{N} [x_i, y_i, z_i]^T, \tag{3}$$

where N is the amount of the object in that category. Then for object o in that category the ground truth $[\delta_x^o, \delta_y^o, \delta_z^o]^T$ of the size residual estimation is calculated as

$$[\delta_x^o, \delta_y^o, \delta_z^o]^T = [x_o, y_o, z_o]^T - [\overline{x}, \overline{y}, \overline{z}]^T. \tag{4}$$

We use mean square error (MSE) loss to predict both the translation and size residual. The total loss function $\mathcal{L}_{res}$ is defined as:

$$\mathcal{L}_{res} = \mathcal{L}_{tra} + \mathcal{L}_{size}, \tag{5}$$

where $\mathcal{L}_{tra}$ and $\mathcal{L}_{size}$ are sub-loss for translation residual and size residual, respectively.

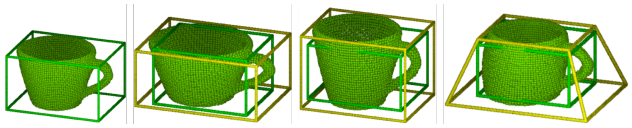

Figure 5. **3D deformed examples**. The new training examples can be generated by enlarging, shrinking, or changing the area of some surfaces of the box-cages. The left one is the original point could with original 3D box-cage, i.e. 3D bounding box. The right three ones are the deformed point clouds with deformed box-cages (shown in yellow color). The green boxes are the original 3D bounding boxes before deformation.

3.5. 3D Deformation Mechanism

One major problem in category-level 6D pose estimation is the intra-class shape variation. The existing methods employed two large synthetic datasets, i.e. CAMERA [41] and 3D model dataset [3] to learn this variation. However, this strategy not only needs extra hardware resources to store these big synthetic datasets but also increases the (pre-)training time.

To alleviate the shape variation issue, based on the fact that the shapes of most objects in the same category are similar [38] (shown in Figure 3), we propose an online box-cage based 3D deformation mechanism for training data augmentation. We pre-define a box-cage for each rigid object (shown in Figure 5). Each point is assigned to its nearest surface of the cage; when we deform the surface, the corresponding points move as well.

Though box-cage can be designed more refined, in experiments, we find that with a simple box cage, i.e. 3D bounding box of the object, the generalization ability of the proposed method is considerably improved (Table 1). Different to [43], we do not need the extra training process to obtain the box-cage of the object, and we do not need target shape to learn the deformation operation either. Our mechanism is totally online, which saves training time and storage space.

To make the deformation operation easier, we first transfer the points to the canonical coordinate system and then perform 3D deformation. Finally we transform them to global scene:

$$\{\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_n\} = R(\mathbb{F}_{3D}(R^T(\mathcal{P} - T))) + T, \tag{6}$$

where $\mathcal{P}$ is the points generated after the 2D detection step. R, T are the pose ground truth. $\{\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_n\}$ are the new generated training examples. $\mathbb{F}_{3D}$ is 3D deformation which includes cage enlarging, shrinking, changing the area of some surfaces.

4. Experiments

4.1. Datasets

NOCS-REAL [41] is the first real-world dataset for category-level 6D object pose estimation. The training set has 4300 real images of 7 scenes with 6 categories. For each category, there are 3 unique instances. In the testing set, there are 2750 real images spread in 6 scenes of the same 6 categories as the training set. In each test scene, there are about 5 objects which makes the dataset clutter and challenging.

LINEMOD [12] is a widely used instance-level 6D object pose estimation dataset which consists of 13 different objects with significant shape variation.

We use the automatic point-wise labeling techniques proposed in [5] to access the label of each point in both training sets.

4.2. Implementation Details

We use Pytorch [23] to implement our pipeline. All experiments are deployed on a PC with i7-4930K 3.4GHz CPU and GTX 1080Ti GPU.

First, to locate the object in RGB images, we fine-tune the YOLOv3 pre-trained on COCO dataset [18] with the training dataset. Then we jointly train the 3DGC autoencoder and residual estimation network. The total loss function is defined as

$$\mathcal{L}_{Shape} = \lambda_{seg}\mathcal{L}_{seg}+\lambda_{rec}\mathcal{L}_{rec}+\lambda_{rot}\mathcal{L}_{rot}+\lambda_{res}\mathcal{L}_{res}, \quad (7)$$

where λs are the balance parameters. We empirically set them as 0.001, 1, 0.001, and 1 to keep different loss values at the same magnitude. We use cross entropy for 3D segmentation loss function $\mathcal{L}_{seg}$.

We adopt Adam [15] to optimize the FS-Net. The initial learning rate is 0.001, and we halve it every 10 epochs. The maximum epoch is 50.

4.3. Evaluation Metrics

For category-level pose estimation, we adopt the same metrics used in [41, 4, 35]:

- IoU_X is Intersection-over-Union (IoU) accuracy for 3D object detection under different overlap thresholds. The overlap ratio larger than the threshold X is accepted.
- n° m **cm** represents pose estimation error of rotation and translation. The rotation error less than n° and the translation error less than m **cm** is accepted.

For instance-level pose estimation, we compare the performance of FS-Net with other state-of-the-art instance-level methods using the ADD-(S) metric [12].

Table 1. **Ablation studies on NOCS-REAL dataset**. We use two different metrics to measure performance. '3DGC' means the 3D graph convolution. 'OPR' means observed points reconstruction. 'DR' represents the decoupled rotation mechanism. 'DEF' denotes the online 3D deformation. In the last row, the values in the bracket are the performance for the reconstruction of the complete object model transformed by the corresponding pose. Please note, for the sake of ablation study, we provide the ground truth 2D bounding box for different methods.

Method	3DGC	DEF	OPR	DR	IoU_{50}	10°10 **cm**
G2L [6]	×	✓	×	×	94.65%	31.0%
G2L+DR	×	✓	×	✓	96.21%	47.81%
Med1	✓	✓	×	×	97.98%	46.4%
Med2	✓	✓	✓	×	95.61%	46.8%
Med3	✓	✓	×	✓	97.34%	61.1%
Med4	✓	×	✓	✓	97.30%	58.2%
Med5	✓	✓	✓	✓	98.04% (94.44%)	65.9% (58.0%)

4.4. Ablation Studies

We use the G2L-Net [6] as the baseline method which extracted the latent feature for rotation estimation via point-wise orientated vector regression, and the ground truth of rotation is the eight corners of 3D bounding box with corresponding rotation. The loss function for rotation estimation is the mean square error between predicted 3D coordinates and ground truth. Compared to baseline, our proposed work has three novelties: a) view-based 3DGC autoencoder for observed point cloud reconstruction; b) rotation decoupled mechanism; c) online 3D deformation mechanism.

In Table 1, we report the experimental results of three novelties on the NOCS-REAL dataset. Comparing Med3 and Med5, we find that reconstruction of the observed point cloud can learn better pose feature. The performance of Med2(Med1, G2L) and Med5(Med3, G2L+DR) shows that the proposed decoupled rotation mechanism can effectively extract the rotation information. The results of Med4 and Med5 demonstrate the effectiveness of the 3D deformation mechanism, which increases the pose accuracy by 7.7% in terms of 10°10 **cm** metric. We also compare the different reconstruction choices: the reconstruction of observed points and the complete object model with corresponding rotation. From the last row of Table 1, we can see that the observed points reconstruction can learn better rotation feature. Overall, Table 1 shows that the proposed novelties can improve the accuracy significantly.

4.5. Generalization Performance

NOCS-REAL dataset provides 4.3k real images that covers various poses of different objects in different categories for training. That means the category-level pose information is rich in the training set. Thanks to the effectively pose feature extraction, FS-Net achieves state-of-the-art performance even with part of the real-world training data. We randomly choose different percentages of the training set to train FS-Net and test it on the whole testing set. Figure

6 shows that: 1) FS-Net is robust to the size of the training dataset, and has good category-level feature extraction ability. Even with 20% of the training dataset, the FS-Net can still achieve state-of-the-art performance; 2) the 3D deformation mechanism significantly improves the robustness and performance of FS-Net.

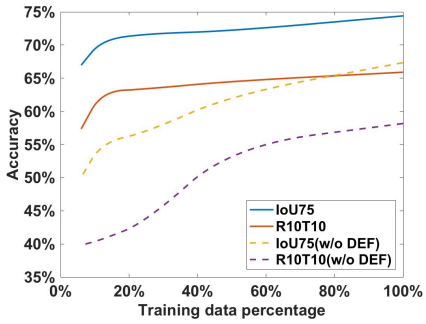

Figure 6. **Generalization performance.** With the given 2D bounding box and a randomly chosen 3D sphere center, we show how the training set size affects the pose estimation performance. 'w/o DEF' means no 3D deformation mechanism is adopted during training.

4.6. Evaluation of Reconstruction

Point cloud reconstruction has a close relationship with pose estimation performance. We compute the Chamfer Distance of the reconstructed point cloud with the ground truth point cloud and compared it with other reconstruction types used by other methods. From Table 2, we can see that the average reconstruction error of our method is 0.86, which is 72.9% and 18.9% lower than that of Shape-Prior [35] and CASS [4], respectively. It shows that our method achieves better pose estimation results via a simpler reconstruction task, i.e. observed points reconstruction rather than complete object model reconstruction.

4.7. Comparison with State-of-the-Arts

4.7.1 Category-Level Pose Estimation

We compare FS-Net with NOCS [41], CASS [4], Shape-Prior [35], and 6D-PACK [39] on NOCS-REAL dataset in Table 4. We can see that our proposed method outperforms the other state-of-the-art methods on both accuracy and speed. Specifically, on 3D detection metric IOU_{50}, our FS-Net outperforms the previous best method, NOCS, by 11.7% and the running speed is 4 times faster. In terms of 6D pose metric 5°**5cm** and 10°10 **cm**, FS-Net outperforms the CASS by the margins of 4.7% and 6.3%, respectively. FS-Net even outperforms 6D-PACK under 3D detection metric IOU_{50}, which is a 6D tracker and needs an initial 6D pose and object size to start. See Figure 7 for more quantitative details. The qualitative results are shown in Figure 8. Please note, we only use real-world data (NOCS-REAL) to train our pose estimation part. Other methods use both synthetic dataset (CAMERA) [41] and real-world data for training. The number of training examples in CAMERA is 275K, which is more than 60 times that of NOCS-REAL (4.3K). It shows that FS-Net can efficiently extract the category-level pose feature with fewer data.

Table 2. **Reconstruction type comparison.** The comparison is on the NOCS-REAL dataset with the Chamfer Distance metric ($\times 10^{-3}$). 'Complete' means the reconstruction of the complete 3D model. 'Observed' denotes only the reconstruction of the observed points.

Methods	CASS [4]	Shape-Prior [35]	Ours
	Complete	Complete	Observed
Bottle	0.75	3.44	1.2
Bowl	0.38	1.21	0.39
Camera	0.77	8.89	**0.44**
Can	0.42	1.56	0.62
Laptop	3.73	2.91	**2.23**
Mug	0.32	1.02	**0.29**
Average	1.06	3.17	**0.86**

Table 3. **Instance-level comparison on LINEMOD dataset.** Our method achieves a comparable performance with the state-of-the-art in both speed and accuracy.

Method	Input	ADD-(S)	Speed(FPS)
PVNet [24]	RGB	86.3%	25
CDPN [17]	RGB	89.9%	33
DPOD [44]	RGB	95.2%	33
G2L-Net [6]	RGBD	98.7%	23
Densefusion[40]	RGBD	94.3%	16
PVN3D [10]	RGBD	99.4%	5
Ours	RGBD	97.6%	20

4.7.2 Instance-Level Pose Estimation

We compare the instance-level pose estimation results of FS-Net on the LINEMOD dataset with other state-of-the-arts instance-level methods. From Table 3, we can see that FS-Net achieves comparable results on both accuracy and speed. It shows that our method can effectively extract both category-level and instance-level pose features.

4.8. Running Time

Given a 640× 480 RGB-D image, our method runs at 20 FPS with Intel i7-4930K CPU and 1080Ti GPU, which is 2 times faster than the previous fastest method 6D-PACK [39]. Specifically, the 2D detection takes about 10ms to proceed. The pose and size estimation takes about 40ms.

Table 4. **Category-level performance on NOCS-REAL dataset with different metrics.** We summarize the pose estimation results reported in the origin papers on the NOCS-REAL dataset. '-' means no results are reported under this metric.

Method	IoU_{25}	IoU_{50}	IoU_{75}	5°5**cm**	10°5 **cm**	10°10 **cm**	Speed(FPS)
NOCS [41]	84.9%	80.5%	30.1%	9.5 %	26.7%	26.7%	5
CASS [4]	84.2%	77.7%	-	23.5 %	58.0%	58.3%	-
Shape-Prior [35]	83.4%	77.3%	53.2%	21.4%	54.1%	-	4
6D-PACK [39]	94.2%	-	-	**33.3 %**	-	-	10
Ours	**95.1%**	**92.2%**	**63.5%**	28.2 %	**60.8%**	**64.6%**	**20**

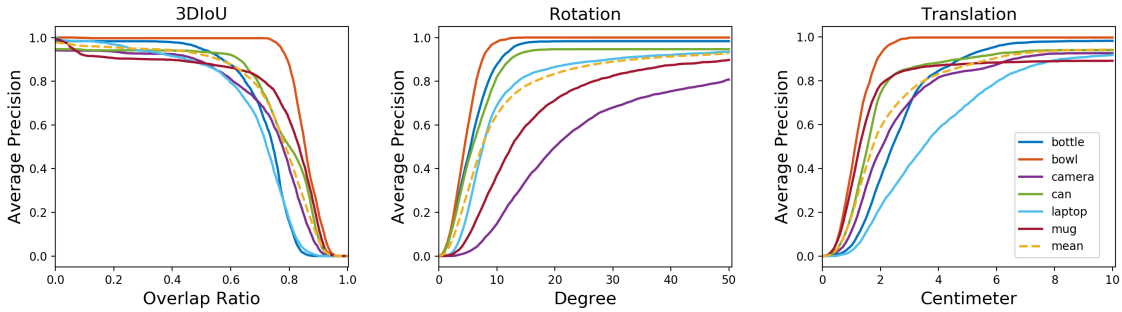

Figure 7. **Result on NOCS-REAL**. The average precision of different thresholds tested on NOCS-REAL dataset with 3D IoU, rotation, and translation error.

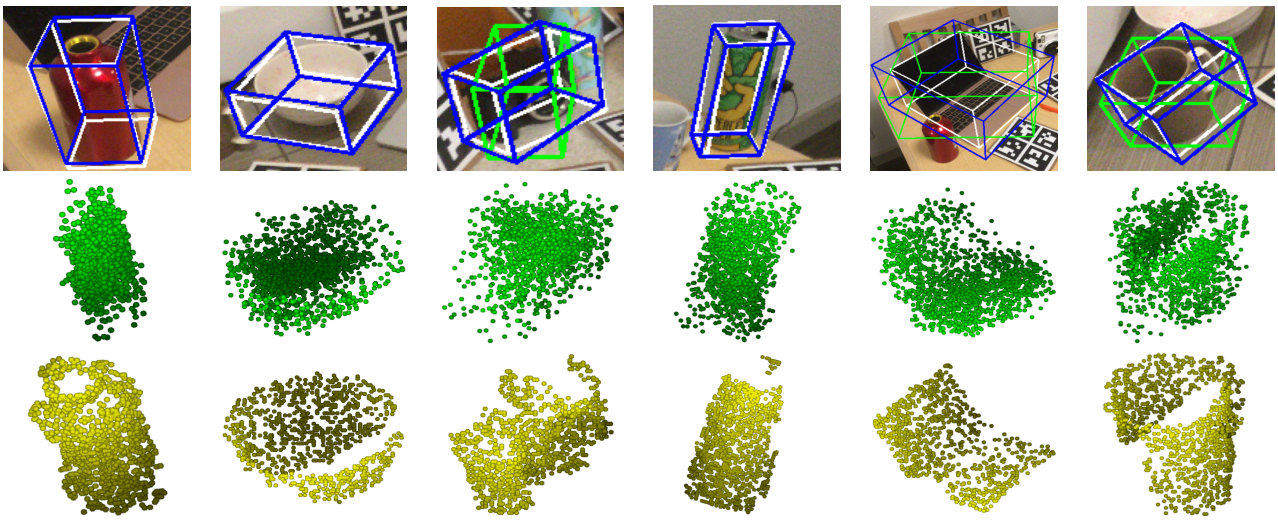

Figure 8. **Qualitative results on NOCS-REAL dataset**. The first row is the pose and size estimation results. White 3D bounding boxes denote ground truth. Blue boxes are the poses recovered from two estimated rotation vectors. The green boxes are the poses recovered from one estimated rotation vector. Our results match ground truth well in both pose and size. The second row is the reconstructed observed points under corresponding poses, although the reconstructed points are not perfectly in line with the target points, the basic orientation information is kept. The third row is the ground truth of the observed points transformed from the observed depth map.

5. Conclusion

In this paper, we propose a fast category-level pose estimation method that runs at 20 FPS which is fast enough for real-time applications. The proposed method first extracts the latent feature by the observed points reconstruction with a shape-based 3DGC autoencoder. Then the category-level orientation feature is decoded by the effective decoupled rotation mechanism. Finally, for translation and object size estimation, we use the residual network to estimate them based on residuals estimation. In addition, to increase the generalization ability of FS-Net and save the hardware source, we design an online 3D deformation mechanism for training set augmentation. Extensive experimental re-

sults demonstrate that FS-Net is less data-dependent, and can achieve state-of-the-art performance on category- and instance-level pose estimation in both accuracy and speed. Please note, our 3D deformation mechanism and decoupled rotation scheme are model-free, which can be applied to other pose estimation methods to boost the performance.

Although FS-Net achieves state-of-the-art performance, it relies on a robust 2D detector to detect the region of interest. In future work, we plan to adopt 3D object detection techniques to directly detect the objects from point clouds.

6. Appendix

This section provides more details about our FS-Net. Section 6.1 describes the details of the 3D deformation mechanism and deformed examples. Section 6.2 provides more quantitative results of the FS-Net on NOCS-REAL [41] dataset and comparison with state-of-the-art method. Section 6.3 demonstrates that the proposed vectors-based rotation representation can be easily extended to handle other symmetric types.

6.1. 3D Deformation Mechanism

As stated in Section 3.5 of the paper, the 3D deformation mechanism is box-cage based and the deformations are applied in a canonical space. In the canonical coordinate system, every box edge is parallel to an axis (shown in Figure 9). This property makes the 3D deformation calculation easier. For example, when we need to elongate/shrink the mug along Y axis by n times. We enlarge the distance between surface $S_{1,2,3,4}$ and surface $S_{5,6,7,8}$ by n times. Since these two surfaces are parallel to the XZ-plane, the x and z coordinates are unchanged. Then points coordinates are changed from $[\mathbf{x}, \mathbf{y}, \mathbf{z}]$ to $[\mathbf{x}, n\mathbf{y}, \mathbf{z}]$. The calculations are similar when we need to elongate/shrink the mug along X or Z axis by n times:

$$[\mathbf{x}, n\mathbf{y}, \mathbf{z}] = \mathbb{F}_x([\mathbf{x}, \mathbf{y}, \mathbf{z}]), \tag{8}$$

$$[n\mathbf{x}, \mathbf{y}, \mathbf{z}] = \mathbb{F}_y([\mathbf{x}, \mathbf{y}, \mathbf{z}]), \tag{9}$$

$$[\mathbf{x}, \mathbf{y}, n\mathbf{z}] = \mathbb{F}_z([\mathbf{x}, \mathbf{y}, \mathbf{z}]), \tag{10}$$

where $\mathbb{F}_{x,y,z}$ is the elongate/shrink operation along corresponding axis.

Further, if the object is the mug or bowl, we may need to change the top or bottom size to generate new shapes (shown in Figure 10). In this case, assuming we enlarge the bottom along X axis by n times, then from bottom to top, the coordinates are changed as:

$$\mathbf{x}_{new} = (1 + (n-1)\frac{l}{L})\mathbf{x}, \tag{11}$$

where l is the distance from a point to the top surface, i.e. $S_{1,2,3,4}$ in Figure 9. L is the height of the object. Please note, all the edges are keep straight while deformation.

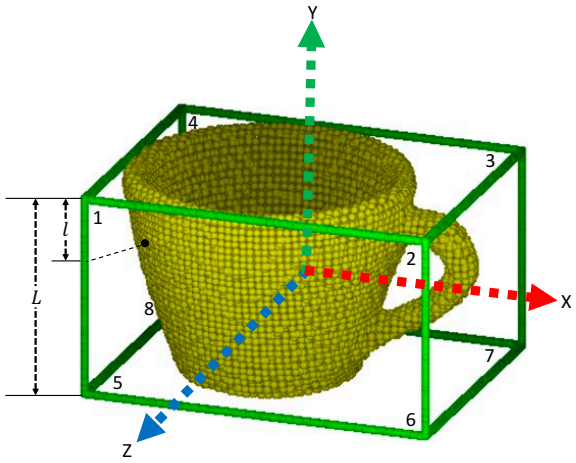

Figure 9. **3D object model**. We assume that the center of 3D bounding box is the origin point of the coordinate. The surface is represented by its four corners. For example, the top surface is represented by $S_{1,2,3,4}$.

6.2. Experimental Results

6.2.1 Detailed Results

We report the specific category pose estimation results under different metrics in Table 5. We also provide the rotation recovered by one/two vectors in Figure 11. We can see that the bounding boxes are well aligned in the recovered vector direction.

Table 5. **Category-Level results.** Object-wise experiments with different metrics.

Category	IoU_{75}	5°5 **cm**	10°5 **cm**	10°10 **cm**
Bottle	0.4710	0.4219	0.8134	0.8755
Bowl	0.9810	0.5916	0.9793	0.9793
Camera	0.5882	0.0176	0.1457	0.1480
Can	0.6334	0.4055	0.7820	0.8141
Laptop	0.3805	0.1659	0.5570	0.6859
Mug	0.7534	0.0874	0.3698	0.3706
Average	0.6345	0.2816	0.6078	0.6455

6.2.2 Comparison with State-of-The-Art

We compare FS-Net with the state-of-the-art method Shape-Prior [35], which utilized point cloud for category-level 6D object pose estimation. Shape-Prior [35] estimated the object size and 6D pose from dense-fusion feature [40], while we estimate the pose from point cloud feature. Figure 12 shows that our FS-Net is robust to color and shape variation, and can handle some failure cases of Shape-Prior. For Shape-Prior, we use the predicted results provided on their website: https://github.com/mentian/object-deformnet.

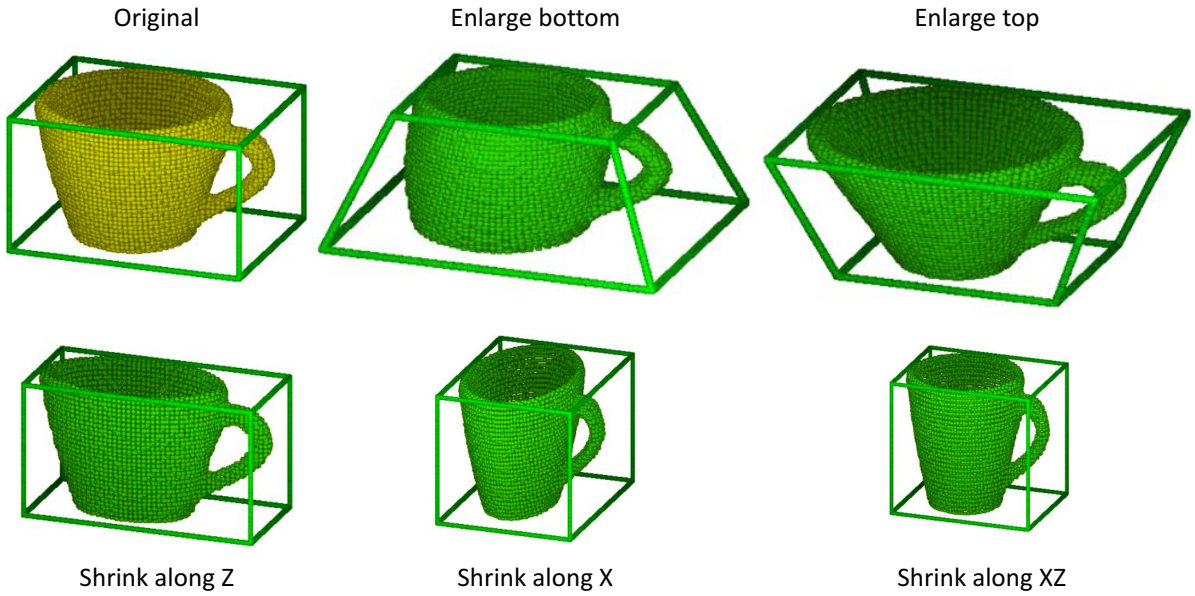

Figure 10. **Examples of different deformations.** We assume that the XYZ axis are the same as Figure 9. The upper right corner is the original point cloud with corresponding box-cage. The rest are the deformed box-cages and point clouds. The deformation operations are described on the top or bottom of the pictures.

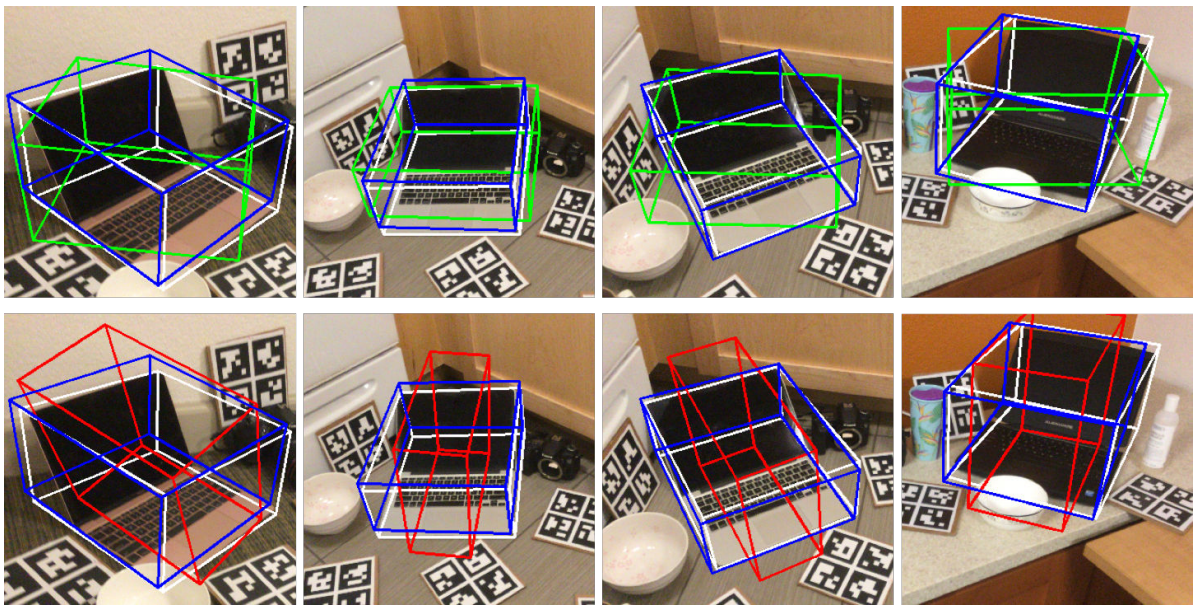

Figure 11. **Rotation recovered by different vectors.** The white boxes are the ground truth. Blue boxes are the rotation recovered by two estimated vectors. The green and red boxes are the rotation recovered by estimated green vector and estimated red vector (see Figure 4 in the paper), respectively. For better illustration, we use ground truth object size to calculate the final 3D bounding box.

6.3. Rotation Representation for Symmetry Object

The vector based rotation representation proposed in the paper can only handle the symmetry objects like bottle, however, in real-world the symmetric types are various (see Figure 13). In this section, we will show how to extend the vector based rotation representation for different symmetric types. Our strategy is inspired by the rotation mapping operation proposed in [25]. In the following, we will show how to find the rotation group (termed proper symmetries in [25]) of a single rotation for common symmetric objects.

Our basic idea is list all the ambiguous rotations of a

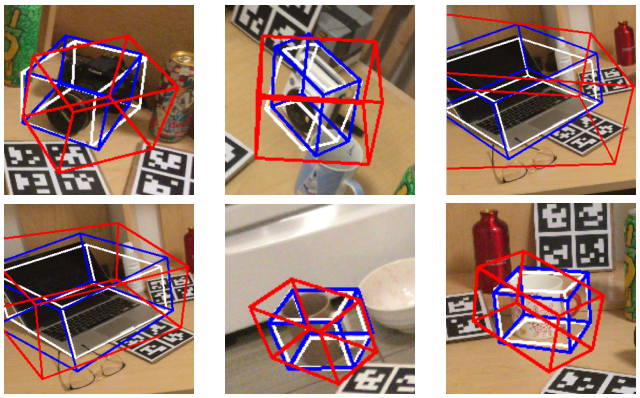

Figure 12. **Qualitative comparison with Shape-Prior**. The white boxes are the ground truth. Blue boxes are our results. Red boxes are the poses predicted by Shape-Prior [35]

single rotation and choose the rotations that has the closest distance with the identity matrix:

$$R^* = \underset{R \in \mathcal{G}(R_i)}{\operatorname{argmin}} \mathcal{D}(R, R_I), \tag{12}$$

where $\mathcal{D}(\cdot,\cdot)$ is the distance between two rotation matrix, $\mathcal{G}(R_i)$ is a group of rotation that can provide the same visual appearance of a given object as rotation R_i. Our goal is to find a rotation R^* that can minimize the rotation distance.

For symmetric object like bottle, we can avoid the rotation ambiguity by only using the green vector to represent the rotation (see Figure 4), however, the case is non-trivial for other symmetric type. In the following, we describe how we find symmetry rotation group for different symmetric types

6.3.1 Symmetry with two Axes

For this kind symmetric objects, in canonical space, when we rotate the object around one axis 180°, we can get the same appearance (see Figure for illustration). Assume that axis is Z axis, for arbitrary rotation R, the appearance $\mathcal{A}$:

$$\mathcal{A}^{R^{Z+}_{180}\mathcal{O}} = \mathcal{A}^{\mathcal{O}}, \tag{13}$$

where R^{Z+}_{180} means rotation the object around Z 180° in clockwise, $\mathcal{O}$ denotes the object. That means we can find the rotation group of each rotation by right multiplication operation R^{Z+}_{180}. Then we use Equation 12 to find the representative rotation in the rotation group.

6.3.2 Symmetry with N Axes

The idea can be easily extend to object with N symmetries around a single axis Z. For this kind of symmetric objects, when we rotate the object around axis Z by $K\frac{360}{N}°(K = 1, 2, \cdots, N)$ in canonical space, the appearance $\mathcal{A}$ of the object is unchanged:

$$\mathcal{A}^{R^{Z+}_{K\frac{360}{N}}\circ\mathcal{O}} = \mathcal{A}^{\mathcal{O}}. \tag{14}$$

Then, the symmetric rotation group $\mathcal{G}(R)$ of rotation R is: $RR^{K\frac{360}{N}°(K=0,1,2,\cdots,N)}$. We find the representative rotation in $\mathcal{G}(R)$ with Equation 12.

6.3.3 General Case

Most symmetric types are included in the description of Section 6.3.1 and 6.3.2. For any other symmetric object, the key idea here is to find the rotation operation that can produce the same appearance of the object. Then use Equation 12 to find the representative rotation.

6.3.4 Decoupled Rotation Representation

Given the representative rotation R^* of ambiguous rotation, we generate its corresponding vector-based representation $\mathcal{V}$ by:

$$\mathcal{V} = R^*[\mathbf{v}_1, \mathbf{v}_2], \tag{15}$$

where $\mathbf{v}_1$ is the vector along with the axis Z mentioned in Section 6.3.1 and 6.3.2, $\mathbf{v}_2$ is the vectors orthogonal with $\mathbf{v}_1$.

References

[1] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3364–3372, 2016. 2

[2] Grigore C Burdea and Philippe Coiffet. *Virtual reality technology*. John Wiley & Sons, 2003. 1

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[4] Dengsheng Chen, Jun Li, Zheng Wang, and Kai Xu. Learning canonical shape space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11973–11982, 2020. 1, 2, 6, 7, 8

[5] Wei Chen, Jinming Duan, Hector Basevi, Hyung Jin Chang, and Ales Leonardis. Ponitposenet: Point pose network for robust 6d object pose estimation. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2020. 2, 6

[6] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, and Ales Leonardis. G2l-net: Global to local network for real-time 6d pose estimation with embedding vector features. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 3, 4, 5, 6, 7

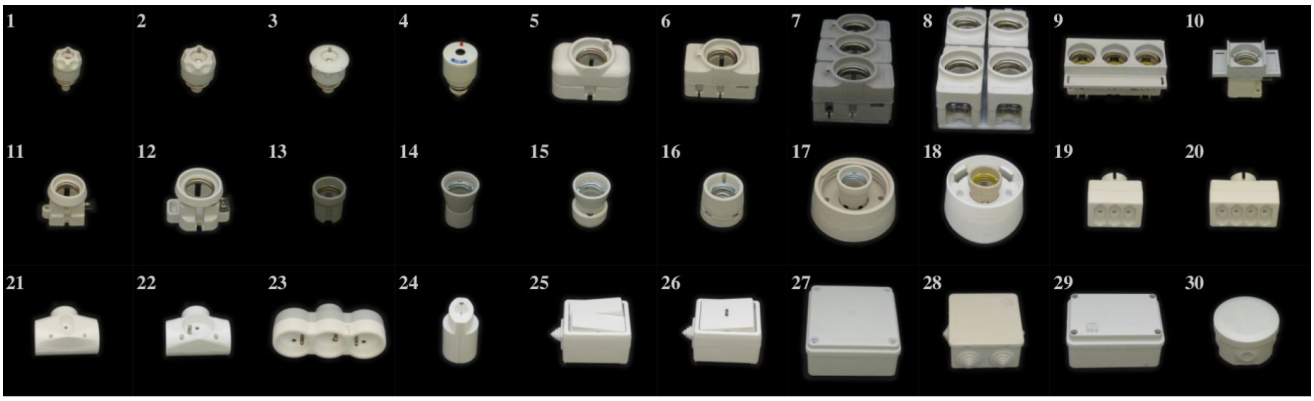

Figure 13. **Different symmetry types**. 30 industry-relevant objects in T-LESS dataset [13]. Object 1, 2, 3, 4 are circular symmetry, object 7, 8, 9, 10 have two symmetry axes, while object 27, 28 have four symmetry axes.

[7] Jaeseok Choi, Yeji Song, and Nojun Kwak. Part-aware data augmentation for 3d object detection in point cloud. *arXiv preprint arXiv:2007.13373*, 2020. 3

[8] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017. 1, 4

[10] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11632–11641, 2020. 1, 7

[11] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):876–888, 2012. 2

[12] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer, 2012. 6

[13] Tomáš Hodan, Pavel Haluza, Štepán Obdržálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 880–888. IEEE, 2017. 12

[14] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976. 2

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] Chi Li, Jin Bai, and Gregory D. Hager. A unified framework for multi-view multi-class object pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018. 1

[17] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7678–7687, 2019. 7

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[19] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1800–1809, 2020. 2, 4

[20] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2016. 1

[21] Eitan Marder-Eppstein. Project tango. In *ACM SIGGRAPH 2016 Real-Time Live!*, page 40. ACM, 2016. 1

[22] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In *The European Conference on Computer Vision (ECCV)*, pages 119–134, 2018. 1, 2

[23] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. Pytorch, 2017. 6

[24] Sida Peng, Yuan Liu, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Pvnet: Pixel-wise voting network for 6dof pose estimation. *arXiv preprint arXiv:1812.11788*, 2018. 2, 7

[25] Giorgia Pitteri, Michaël Ramamonjisoa, Slobodan Ilic, and Vincent Lepetit. On object symmetries and 6d pose estimation from images. In *2019 International Conference on 3D Vision (3DV)*, pages 614–622. IEEE, 2019. 5, 10

[26] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 4, 5

[27] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (CVPR)*, July 2017. 2, 3, 4, 5

[28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017. 4

[29] Mahdi Rad and Vincent Lepetit. Bb8: a scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3828–3836, 2017. 1, 2

[30] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Feature mapping for learning fast and accurate 3d pose inference from synthetic images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4663–4672, 2018. 2

[31] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2, 3, 4

[32] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 2

[33] Martin Sundermeyer, Maximilian Durner, En Yen Puang, Zoltan-Csaba Marton, Narunas Vaskevicius, Kai O Arras, and Rudolph Triebel. Multi-path learning for object pose estimation across domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13916–13925, 2020. 4

[34] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 699–715, 2018. 4

[35] Meng Tian, Marcelo H Ang Jr, and Gim Hee Lee. Shape prior deformation for categorical 6d object pose and size estimation. *arXiv preprint arXiv:2007.08454*, 2020. 1, 2, 5, 6, 7, 8, 9, 11

[36] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv preprint arXiv:1809.10790*, 2018. 1

[37] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (4):376–380, 1991. 1, 2

[38] Haley A Vlach. How we categorize objects is related to how we remember them: the shape bias as a memory bias. *Journal of experimental child psychology*, 152:12–30, 2016. 2, 5

[39] Chen Wang, Roberto Martín-Martín, Danfei Xu, Jun Lv, Cewu Lu, Li Fei-Fei, Silvio Savarese, and Yuke Zhu. 6-pack: Category-level 6d pose tracker with anchor-based keypoints. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10059–10066. IEEE, 2020. 7, 8

[40] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martin-Martin, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 7, 9

[41] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019. 1, 2, 5, 6, 7, 8, 9

[42] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017. 1

[43] Wang Yifan, Noam Aigerman, Vladimir G Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3d deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 75–83, 2020. 5

[44] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 1941–1950, 2019. 7

[45] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 5

[46] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499, 2018. 4

[47] Menglong Zhu, Konstantinos G Derpanis, Yinfei Yang, Samarth Brahmbhatt, Mabel Zhang, Cody Phillips, Matthieu Lecce, and Kostas Daniilidis. Single image 3d object detection and pose estimation for grasping. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3936–3943. IEEE, 2014. 1