Self-Supervised Category-Level 6D Object Pose Estimation with Deep Implicit Shape Representation

Wanli Peng Jianhang Yan Hongtao Wen Yi Sun*

Dalian University of Technology, China
{1136558142, yjh97, wenht}@mail.dlut.edu.cn, lslwf@dlut.edu.cn

Abstract

Category-level 6D pose estimation can be better generalized to unseen objects in a category compared with instance-level 6D pose estimation. However, existing category-level 6D pose estimation methods usually require supervised training with a sufficient number of 6D pose annotations of objects which makes them difficult to be applied in real scenarios. To address this problem, we propose a self-supervised framework for category-level 6D pose estimation in this paper. We leverage DeepSDF as a 3D object representation and design several novel loss functions based on DeepSDF to help the self-supervised model predict unseen object poses without any 6D object pose labels and explicit 3D models in real scenarios. Experiments demonstrate that our method achieves comparable performance with the state-of-the-art fully supervised methods on the category-level NOCS benchmark. Our code is available at https://github.com/swords123/SSC-6D.

1 Introduction

6D object pose estimation is an important task in computer vision that provides object location and orientation. It is widely applied in robotic manipulation and 3D scene understanding. Currently, most existing works (Peng et al. 2019; He et al. 2020; Zakharov, Shugurov, and Ilic 2019; Xiang et al. 2017; Wang et al. 2019a) focus on instance-level pose estimation, relying on 6D pose annotations and exact 3D object models available beforehand. However, it is difficult to annotate the 6D pose and build a 3D model for every object in real scenarios. Recently, category-level 6D pose estimation (Wang et al. 2019b; Chen et al. 2021, 2020; Tian, Ang, and Lee 2020) that can predict both the 6D pose and size of unseen objects without explicit 3D models has started to gain attention.

Category-level 6D pose estimation commonly learns a categorical shape prior shared by all instances within a category, which enables 6D pose and size estimation even for unseen object instances. However, existing methods usually require supervised training with a sufficient amount of 6D pose annotations of objects. Yet, annotating 6D poses for every object in the real world is an unfeasible task. The key to solving this problem is to adopt self-supervised learning for pose estimation. Generally, self-supervised methods employ a *render-and-compare* pipeline (Wang et al. 2020; Yang, Yu, and Yang 2021; Manhardt et al. 2020) that is trained by the correspondence between the reconstructed object 3D model and object observation to derive pose parameters (R, t and s). The high-quality 3D geometric representation of objects is therefore crucial to self-supervised pose estimation, especially for category-level pose estimation. However, existing discrete 3D representations such as voxels, point clouds and meshes are not enough to represent high-quality shapes. At present, self-supervised category-level pose estimation without 6D pose labels and suitable 3D models is a challenging task and there are few studies in this field.

In our work, we move forward along this challenging direction to explore self-supervised category-level 6D pose estimation. An overview of the architecture is illustrated in Fig. 1. Inspired by the recently proposed continuous deep implicit functions which are more expressive and efficient for representing complex shapes than existing discrete 3D representations, we use a deep implicit function called DeepSDF (Park et al. 2019) as 3D object representation in this paper. To enable the self-supervised method to estimate the pose of unseen objects in a category without an explicit 3D model for each object, we first train a DeepSDF decoder on synthetic data to learn categorical shape prior, then build a self-supervised framework to estimate the pose parameters and shape latent vector for shape reconstruction. The self-supervised framework also needs to be pretrained with synthetic data in order to have a good initial state. Given the initial network output (pose and shape latent vector) in the self-supervised training, we first reconstruct the object's shape from the DeepSDF decoder and then design several novel loss functions based on DeepSDF to enforce an optimal alignment in 3D space between the reconstructed shape and ground truth point cloud with the predicted rotation, translation, and scale factor. The discrepancy in shape produces error gradients that backpropagate to the pose parameters and the shape latent vectors so that both pose and shape can be optimized.

In summary, the contributions of our work are as follows:

- We propose a self-supervised framework with a deep implicit 3D surface representation for category-level 6D pose estimation, which can predict unseen object poses without an explicit 3D model and pose annotation for

*Corresponding author.

each object in real scenarios.

- We leverage DeepSDF as a 3D object representation and design several novel loss functions based on DeepSDF which makes the pose and shape estimation simultaneously converge to the optimal solution.
- Our network generalizes well to the intra-category unseen objects and achieves performance almost on par with the state-of-the-art fully supervised methods on the category-level NOCS benchmark (Wang et al. 2019b).

2 Related Work

In this section, we briefly review the recent literatures on fully supervised and self-supervised methods for category-level pose estimation, while we also review the literatures for shape representation.

Fully Supervised Methods: The greatest challenge for estimating 6D poses at the category-level is to tackle the intra-class object variation problem. Wang et al. (2019b) built a CNN to map the observed pixels to the NOCS representation and the full 6D pose and size were calculated using the Umeyama algorithm (Umeyama 1991). Similar to (Wang et al. 2019b), Tian, Ang, and Lee (2020) estimated the deformation field of the mean shape to predict the correspondences between the observed depth map and the points in NOCS. CASS (Chen et al. 2020) and FS-Net (Chen et al. 2021) built an encoder-decoder network to effectively learn the category-level features and estimated 6D pose and size directly. DualPoseNet (Lin et al. 2021) constructed an additional implicit pose decoder to reconstruct the input point cloud in canonical pose which enables refinement during testing. However, these methods require a sufficient amount of labeled data for training, which is time-consuming and laborious. In our work, we build a self-supervised framework that does not require any 3D annotations in the real world but solely takes object observations as self-supervision.

Self-Supervised Methods: Since few works focus on self-supervised category-level pose estimation, we also introduce some instance-level methods. Self-6D (Wang et al. 2020) and DSC-PoseNet (Yang, Yu, and Yang 2021) built instance-level 6D pose estimation frameworks by rendering the corresponding CAD models of targets. However, obtaining exact 3D models for every object is also difficult in real scenarios. LatentFusion (Park et al. 2020) performed pose estimation using a few reference images rather than 3D scans, which is also inflexible in real applications. CPS++ (Manhardt et al. 2020) introduced a self-supervised extension to bridge the synthetic-to-real domain gap by rendering the reconstructed mesh from AtlasNet (Groueix et al. 2018). Considering that the quality of objects' 3D geometric representation is crucial for our self-supervised method, we utilize the DeepSDF (Park et al. 2019) as the shape representation which enables higher quality shape representation and makes it easier to discover the common properties of objects in a certain category.

Shape Representation and Rendering: There have been several representations of 3D models. Early works used voxels to represent 3D shapes and utilized 3D CNNs for feature extraction or object reconstruction (Wu et al. 2015; Choy et al. 2016). However, due to the limitations of computation and memory resources, voxels cannot represent 3D shapes with high resolution. Point cloud (Qi et al. 2017; Fan, Su, and Guibas 2017) is a lightweight representation and easy to be collected by sensors, but it is difficult to obtain watertight surfaces from point clouds since there is no topological relationship among points. Mesh-based methods (Groueix et al. 2018; Wang et al. 2018) reconstructed the 3D model from a simple topology, which may lead to nonclosed surface or low performance for complex topologies. For higher-quality 3D geometric representation, recently proposed deep implicit functions (Park et al. 2019; Mescheder et al. 2019; Chen and Zhang 2019) learned accurate geometry through a decoder and produced continuous surfaces with unlimited resolution and complex topologies. Therefore, we adopt one of the deep implicit functions called DeepSDF (Park et al. 2019) as the shape representation in our work and learn the category-level shape prior to find the common properties for a specific category.

3 Method

We present our framework of the self-supervised category-level 6D pose estimation with a 3D shape prior represented as DeepSDF (Park et al. 2019). The framework is designed end-to-end, taking the RGB image and point cloud as input and output the object pose and shape (DeepSDF latent vector). We first leverage DeepSDF to learn the categorical shape prior from synthetic CAD models (Sec. 3.1) and then present our proposed architecture for self-supervised 6D pose and shape estimation (Sec. 3.2). In the self-supervised training for deriving the pose parameters (R, t, s), we introduce our novel losses based on DeepSDF to align the reconstructed 6D pose and shape in 3D space with the object observation. Moreover, we also pretrain our network on the synthetic dataset to provide good initialization. Finally, more implementation details are given in Sec. 3.3.

3.1 Category-Level Shape Prior Learning

The greatest challenge for category-level pose estimation is solving the intra-class object variation problem and predicting the pose for previously unseen objects. Although the shape varies among different instances, objects of the same category generally have common attributes and semantic structures. For example, cans are usually cylindrical, and the differences among cans are just the height and diameter of the cylinder. Inspired by the fact that humans can easily infer a reasonable 3D shape of unseen objects from their extensive experience and prior knowledge, we first train a DeepSDF decoder to learn the category-level shape prior from the synthetic dataset as shown at the bottom of Fig. 1.

SDF means Signed Distance Function where SDF value of a point represents the distance to the surface boundary and the sign indicates whether the region is inside(-) or outside(+) of the shape. A shape's boundary is encoded as the zero-level-set. The DeepSDF uses a deep neural network to regress the continuous SDF value $sdf \in \mathbb{R}$ from the query points $p \in \mathbb{R}^3$ and shape latent vector $\mathbf{v} \in \mathbb{R}^k$ as shown in Eq. 1:

$$\mathcal{F}(p; \mathbf{v}) = sdf \tag{1}$$

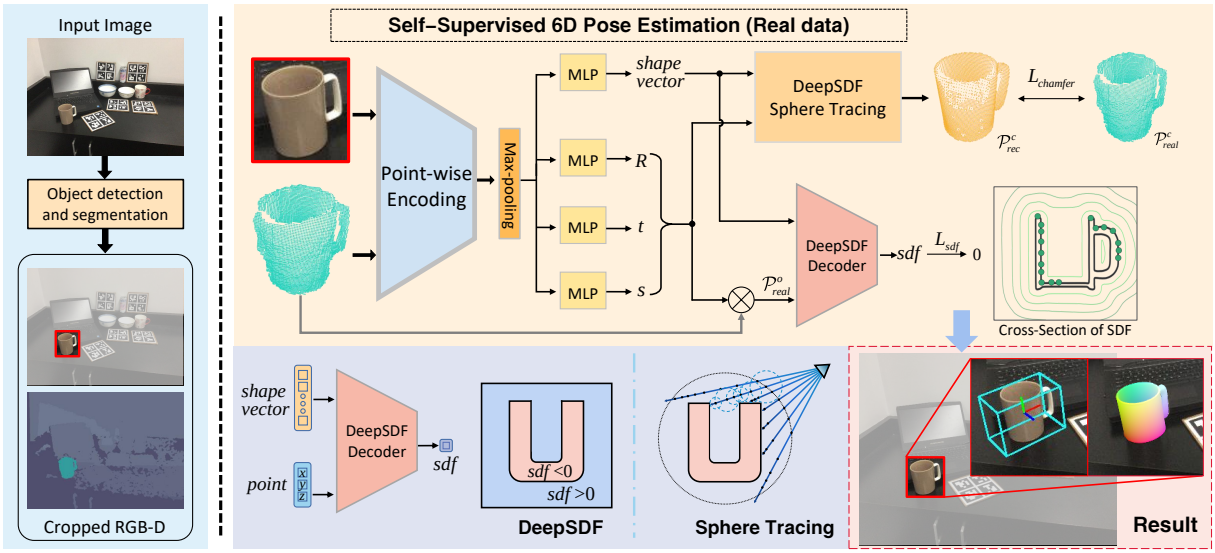


Figure 1: Overview of the proposed method. After segmenting individual object instances from the cluttered scene, we build an end-to-end network to jointly predict the rotation, translation, scale and shape latent vector of the object. Several self-supervised loss functions are designed to constrain consistency between predicted parameters and object observations in real-world data.

where $\mathcal{F}$ is the DeepSDF decoder which is composed of eight fully connected layers. After training, the decoder can learn the common properties of a certain category and each instance is mapped to a low-dimensional shape latent vector. We set the dimension k to 16 for all categories. With the decoder, the object surface can be implicitly represented by the iso-surface of $\mathcal{F}(\mathbf{v}) = 0$ and extracted using the Marching Cubes (Lorensen and Cline 1987) algorithm. Note that the reconstructed model is in the normalized target coordinate system and contained within a unit sphere.

In our work, we use the CAD models in the synthetic dataset of NOCS (Wang et al. 2019b) to train the DeepSDF decoder $\mathcal{F}$. Since objects of different categories have their specific shape prior, we train the network separately for each category. In the next stage for self-supervision, the learned weights of the decoder will be frozen so that the network can remember the common properties of each category and the shape predictor only needs to estimate the low-dimensional latent vector for each instance thus reducing the difficulty of shape prediction.

3.2 Self-Supervised Pose and Shape Estimation

Network Architecture: An overview of our self-supervised pipeline is depicted in Fig. 1. For an observed cluttered scene, we first detect and segment the individual objects from the images. For each detected instance, we crop the RGB image using the bounding boxes and segment the depth map with the masks, which is converted to a point cloud with camera intrinsics. Taking the image patch and point cloud patch as input, we establish an end-to-end network to predict the object's rotation $R \in SO(3)$, translation $t \in \mathbb{R}^3$ and scale factor $s \in \mathbb{R}$ along with shape latent vector $\mathbf{v} \in \mathbb{R}^k$ as shown in Fig. 1. Specifically, following Densefusion (Wang et al. 2019a), we use a CNN-based encoder-decoder network (Zhao et al. 2017) to extract the appearance features from the RGB image patch and utilize an architecture like PointNet (Qi et al. 2017) to extract the geometric features from the point cloud. To perform pixel-wise fusion, the geometric feature of each point is concatenated with its corresponding image feature using known camera intrinsics. Due to the limitation of computation and memory resources for reconstruction, we aggregate the features of all points using a max-pooling layer and estimate only one pose and latent vector for the whole object using several fully connected layers, rather than predict one pose for each pixel-wise feature like (Wang et al. 2019a). In addition, considering that the translation t is related to the points coordinates, we predict the residual translation relative to the average position of the segmented point patch, which is easier than estimating the absolute translation directly.

Self-Supervised Training: To train our network without 3D annotations in the real world, we design several self-supervised loss functions to enforce the alignment between the reconstructed models and the object's real observations. Inspired by the recently proposed *render-and-compare* methods (Wang et al. 2020), we first use the chamfer loss $L_{chamfer}$ to provide coarse guidance of 6D pose and scale factor and establish the geometric alignment between the object's observed point cloud $\mathcal{P}^c_{real}$ and the visible surface of the reconstructed model $\mathcal{P}^c_{rec}$, where the superscript c represents the camera coordinate system. Specifically, to obtain the visible surface from the DeepSDF representation, we run the sphere tracing algorithm proposed in DIST (Liu et al. 2020) to obtain the minimal absolute SDF value along each pixel's ray direction with its corresponding point, as shown at the bottom of Fig. 1. The visible point cloud $\mathcal{P}^o_{rec}$ in the normalized target coordinate system can be acquired by checking whether the minimal absolute SDF value is less

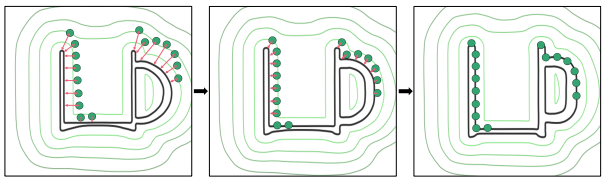


Figure 2: The process of pose and shape optimization using L_{sdf}. The observed point cloud (green points) gradually coincides with the surface of the object (black mug).

than a small convergence threshold. we use the predicted rotation R, translation t, and scale s to transform $\mathcal{P}^o_{rec}$ to the camera coordinate system $\mathcal{P}^c_{rec}$ and build the symmetric chamfer loss as:

$$\begin{aligned} L_{chamfer} = & \frac{1}{|\mathcal{P}^c_{rec}|} \sum_{p_i \in \mathcal{P}^c_{rec}} \min_{p_j \in \mathcal{P}^c_{real}} \|p_i - p_j\|_2 \\ & + \frac{1}{|\mathcal{P}^c_{real}|} \sum_{p_j \in \mathcal{P}^c_{real}} \min_{p_i \in \mathcal{P}^c_{rec}} \|p_j - p_i\|_2 \end{aligned} \tag{2}$$

with

$$\mathcal{P}^c_{rec} = sR\mathcal{P}^o_{rec} + t \tag{3}$$

To simplify gradient propagation and stabilize convergence, we disable backpropagation for $\mathcal{P}^o_{rec}$, while gradients only backpropagate through R, t, and s.

$L_{chamfer}$ can only adjust the pose and scale, but shape is also indispensable for pose estimation. Therefore, an additional loss function is introduced to constrain the object's pose and shape at the same time and expects the network to converge to a global optimization for both pose and shape. Specifically, considering that the point clouds collected by a depth sensor are the surface points of objects, their corresponding SDF values should be zero. To make full use of this property, we first transform the collected point cloud $\mathcal{P}^c_{real}$ to the normalized target coordinate system $\mathcal{P}^o_{real}$ using the predicted R, t, and s. Then, we feed the transformed points and shape latent vector into the DeepSDF decoder to enforce the output SDF values closer to zero. This process is illustrated in Fig. 2. The SDF alignment loss L_{sdf} is represented as follows:

$$L_{sdf} = \frac{1}{|\mathcal{P}^o_{real}|} \sum_{p_j \in \mathcal{P}^o_{real}} |\mathcal{F}(p_j; \mathbf{v})| \cdot \mathbb{1}(\|p_j\|_2 < 1) \tag{4}$$

with

$$\mathcal{P}^o_{real} = \frac{1}{s} R^T (\mathcal{P}^c_{real} - t) \tag{5}$$

where $\mathbb{1}(\cdot)$ is the indicator function that indicates only the points inside the unit sphere in the target coordinate system are considered, which is to avoid the influence of both the 6D pose with large deviations and outliers of the point cloud on shape adjustment. The discrepancy in the SDF value from zero produces error gradients that backpropagate through the decoder to the pose parameters and the shape latent vectors so that both poses and shapes are simultaneously optimized. We do so on the basis of such fact that shapes and poses are inseparable, as correct poses can guide the estimation of shapes which in turn helps to infer the poses more accurately. We repeat this iterative procedure until convergence.

In addition, to constrain the magnitude of the latent vector $\mathbf{v}$, we define a regulation loss L_{reg} for training:

$$L_{reg} = e^{\beta \|\mathbf{v}\|_2} - 1 \tag{6}$$

where we select the hyper-parameter β manually.

Overall, the total self-supervised loss is defined as:

$$\begin{aligned} L_{self} = & \lambda_{chamfer} L_{chamfer} + \lambda_{sdf} L_{sdf} \\ & + \lambda_{reg} L_{reg} \end{aligned} \tag{7}$$

where $\lambda_{chamfer}$, λ_{sdf} and λ_{reg} are the trade-off parameters to balance each loss. In this paper, we set $\lambda_{chamfer} = 100$, $\lambda_{sdf} = 1.0$ and $\lambda_{reg} = 0.01$.

Pretraining and Joint Learning: For the training of our self-supervised framework, a good initialization is very important for pose and shape estimation. In our paper, we first pretrain our network on the synthetic dataset in a fully supervised manner, which can provide good initialization and learn prior information about the approximate distribution of the target's pose and shape. The fully supervised loss function on the synthetic dataset is defined as:

$$L_{sup} = L_{pose} + L_{\mathbf{v}} \tag{8}$$

with

$$L_{pose} = \frac{1}{N} \sum_{i=1}^{N} \|(sRx_i + t) - (s_{gt} R_{gt} x_i + t_{gt})\|_2 \tag{9}$$

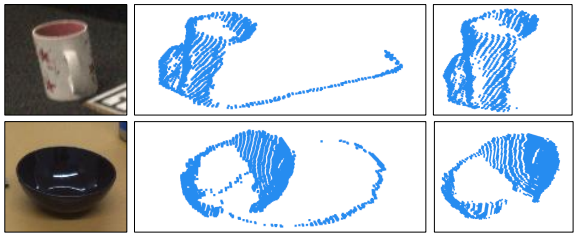
and

$$L_{\mathbf{v}} = \|\mathbf{v} - \mathbf{v}_{gt}\|_1 \tag{10}$$

where x_i is the i^{th} point of the N randomly sampled points from the 3D model. R_{gt}, t_{gt} and s_{gt} are the ground truth rotation, translation and scale factor, respectively, while $\mathbf{v}_{gt}$ is the ground truth shape latent vector generated from the training stage of the DeepSDF decoder as described in Sec.3.1. For symmetrical objects, we follow the solution (Pitteri et al. 2019) to map both the predicted rotation R and ground truth R_{gt} to canonical rotations R^* and R^*_{gt}, respectively, and L_{pose} is calculated as:

$$L_{pose} = \frac{1}{N} \sum_{i=1}^{N} \left\|(sR^*x_i + t) - (s_{gt} R^*_{gt} x_i + t_{gt})\right\|_2 \tag{11}$$

After pretraining, we train our network in a self-supervised manner in the real dataset. However, when a trained neural network is transferred to a new dataset for training, it will gradually forget the knowledge learned from the old dataset. Considering that the prior information learned on the synthetic dataset is very important for training in real scenes, we propose a simple joint learning strategy to make our network remember the old knowledge. Specifically, for each batch, we feed both the labeled synthetic samples and unlabeled real samples to our network simultaneously, and the loss function is computed by the combination of the fully supervised loss L_{sup} for the synthetic data and self-supervised loss L_{self} for the real data. In summary, the



(a) (b) (c)

Figure 3: Visualization before and after our point cloud filter pipeline. (a) is the RGB images. (b) and (c) are the point clouds before and after our point cloud filter pipeline. It can be observed that our point cloud filter pipeline is effective in removing the outliers for different categories.

total loss for training our self-supervised framework in each batch is defined as:

$$L = \frac{1}{M}\sum_{i=1}^{M} L_{sup}^{i} + \frac{1}{N}\sum_{j=1}^{N} L_{self}^{j} \tag{12}$$

where M and N are the number of synthetic and real samples in each batch. As shown in Eq. 12, the contributions of the two types of data to the total loss L are equivalent, so the impact of their ratio is negligible.

Point Cloud Filtering: In our work, we mainly use the collected point cloud to supervise our network. However, due to the inaccurate segmentation masks and the displacement between the depth sensor and RGB camera, the segmented foreground point clouds often contain some background points, which may make network learning deviate. Therefore, we present a point cloud filter pipeline to remove these outliers. As shown in Fig. 3, the outliers are mainly the points collected from the desktop. we first remove the points on the desktop using the RANSAC algorithm (Zhou, Park, and Koltun 2018) and then the sparse outliers that far away from their neighbors with the statistical outlier removal algorithm (Zhou, Park, and Koltun 2018). Finally, the remaining points are clustered using the DBSCAN algorithm (Ester et al. 1996) where the cluster closest to the camera is selected as the final point cloud.

3.3 Implementation Details

To segment individual object instances from the cluttered scene, we use the predicted results from Mask R-CNN (He et al. 2017), which is fine-tuned on the NOCS dataset (Wang et al. 2019b) for inference. For each instance, we crop and resize the image patch to 192×192 and randomly sample 1024 points as the input of our network. We train the prediction network for each category separately so that the estimated poses and shapes do not interfere with each other among different classes.

Benefited from our self-supervised method that does not require ground truth pose and shape for supervision, we flip the images in the real dataset for data augmentation, which is easier to implement than the fully supervised method since it is difficult to calculate the corresponding 6D pose if the image is flipped. Note that this data augmentation method can only be applied for mirror-symmetric targets since the image flip means the corresponding target model is mirrored.

We adopt three stages to train our self-supervised framework. The first stage trains the DeepSDF decoder using the CAD models for 2000 epochs. Then, we sample 30000 images on the synthetic dataset and pretrain our prediction network for 10 epochs with the fully supervised loss L_{sup}. In the third state, we use the proposed joint learning strategy in which each batch contains 12 synthetic samples and 2 real samples with the overall loss L as Eq.12 for self-supervised training. We use the Adam optimizer (Kingma and Ba 2014) with an initial learning rate 0.0001 for training which takes additional 8 epochs on one NVIDIA 2080Ti GPU for approximately 5 hours for each category.

4 Experiment

4.1 Experimental Setup

Dataset: We conduct our experiments on the NOCS (Wang et al. 2019b) dataset, which contains six object categories including *bottle*, *bowl*, *can*, *camera*, *laptop*, and *mug*. It consists of two parts: a *synthetic dataset* named CAMERA, which contains 275K rendered images with 1085 CAD models, and a *real-world dataset* named REAL, which consists of 4.3K real-world images from 7 scenes for training and 2.75K from 6 scenes for testing. There are 3 unseen instances per category in each set of the REAL dataset. We learn the shape prior and pretrain our network using the synthetic CAMERA dataset while applying the self-supervision on the REAL dataset.

Evaluation Metrics: We evaluate our method and report the performance on both 3D object detection and 6D pose estimation following (Wang et al. 2019b). We compute the average precision of the 3D Intersection-Over-Union (IoU) at different thresholds for object detection. For 6D pose estimation, we report the average precision of $n^{\circ}m$ cm, which denotes that the error is less than n° for rotation and m cm for translation. For symmetric categories (*bottle*, *bowl* and *can*), the rotation error around the symmetry-axis is not considered. We use the average result from 6 repeated experiments as the final result in this paper.

4.2 Comparison with Other Methods

Since there are few studies in self-supervised category-level pose estimation, we list the performances of the state-of-the-art fully supervised methods in order to see the gap between the two kinds of learning models, as shown in Table 1. From the experimental results, we can see that our method without ICP outperforms NOCS (Wang et al. 2019b) by a large margin in all metrics and achieves almost on par performance compared to DeformNet (Tian, Ang, and Lee 2020), although our method does not rely on any 3D annotations in the real world for training. Both our method and CASS (Chen et al. 2020) use an end-to-end prediction network to estimate the object pose directly, while CASS also learns pose-independent features to assist pose estimation. Even so, we still achieve comparable performance. The success of our method is mainly due to our effective self-supervised loss functions and training scheme,

Dataset	Method	3D Labels	mAP (%)						
			IoU_{25}	IoU_{50}	5°2cm	5°5cm	10°2cm	10°5cm	10°10cm
NOCS-REAL	NOCS	✓	78.72	47.23	7.26	9.86	14.04	24.25	24.51
	CASS	✓	68.09	25.95	19.66	23.60	51.61	59.02	59.25
	DeformNet	✓	80.22	68.73	19.11	21.15	43.39	54.08	56.00
	Ours	×	**83.16**	**72.95**	16.76	19.62	44.12	54.53	56.24
	Ours w/ ICP	×	83.11	72.67	**28.60**	**33.39**	**51.82**	**62.93**	**65.07**

Table 1: Quantitative comparison with other fully supervised methods on the NOCS-REAL test set, where IoU_{25}, IoU_{50} denote the 3D IoU at 25% and 50% respectively. We recalculate all the metrics using the revised evaluation source code.

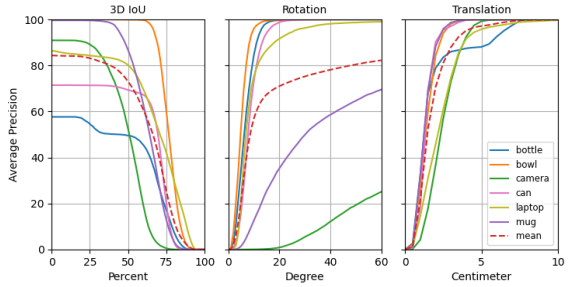


Figure 4: Results on the NOCS-REAL test set. The average precision at different thresholds on 3D IoU, rotation error, and translation error.

which can provide great guidance for pose estimation. In addition, training the network separately for each category is indeed an assistance for the performance improvement. Fig. 4 shows the more detailed average precision at different thresholds for each category. It should be noted that there is a small mistake in the original evaluation source code, thus we revised it and recalculated all the metrics. The evaluation source code after revision is given in our released code.

Our method can also optionally reconstruct the models from the predicted latent vectors using the Marching Cubes (Lorensen and Cline 1987) and conduct the ICP (Besl and McKay 1992) refinement to further improve the accuracy of pose estimation. The result shows that employing ICP can strongly enhance our performance for the pose estimation metrics, especially for stricter metrics such as 5°2cm with an 11.84% performance improvement. There is no obvious impact for the 3D IOU metrics since this metric is not sensitive to the performance of pose estimation when it is good enough. With ICP refinement, our method outperforms the others in all metrics, which indicates that our proposed self-supervised framework is effective in real applications.

4.3 Ablation Study

Evaluations of the training scheme: To verify the effectiveness of our self-supervised framework, we report the experimental results before and after self-supervised training on the NOCS-REAL dataset, as shown in Table 2. *Test-directly* denotes only pretraining our network on the synthetic dataset and testing on the real-world data directly. Due to the large domain gap between synthetic and real-world data, it results in an extremely poor performance. After self-supervised training on the real-world data, represented as

Model	mAP (%)			
	IoU_{50}	5°5cm	10°5cm	10°10cm
Test-Directly	38.67	4.34	9.07	9.15
Self-Sup w/o Aug	68.20	15.21	48.42	49.80
Self-Sup w/ Aug	**72.95**	**19.62**	**54.53**	**56.24**

Table 2: Evaluation of our self-supervised training scheme.

Model	mAP (%)			
	IoU_{50}	5°5cm	10°5cm	10°10cm
w/o $L_{chamfer}$	17.32	15.31	29.28	33.93
w/o L_{sdf}	62.28	13.81	38.54	39.66
w/o L_{sup}	69.71	11.70	49.61	50.93
Full Loss	**72.95**	**19.62**	**54.53**	**56.24**

Table 3: The impact of each loss function on the NOCS-REAL test set.

Self-Sup w/o Aug, its performance significantly improves, e.g., the performance increases approximately 40% at the 10°10cm metric from 9.15% to 49.80%, which demonstrates that our proposed self-supervised loss functions provide enough constraint to estimate correct 6D poses of objects even if no 3D annotations are provided in the real world.

In addition, benefited from our self-supervised method that does not rely on labeled poses, we also apply data augmentation by flipping the RGB-D images together with its corresponding segmentation masks, denoted as *Self-Sup w/ Aug*. Note that we do not flip the data for the *camera* category since not all cameras are mirror-symmetrical. Based on this strategy, our method obtains strongly promising performance as shown in the last row of Table 2.

Evaluations of the loss functions: We analyze the impact of each loss function in our self-supervised framework, as shown in Table 3. Without $L_{chamfer}$, the performance is extremely degraded, especially for the IoU_{50} because $L_{chamfer}$ provides coarse guidance of the translations, rotations, and scale factors of objects. Moreover, L_{sdf} works well only when the reconstructed models approximately align with the collected point cloud under the constraint of chamfer loss. Therefore, $L_{chamfer}$ is so critical that when it is absent, the network exhibits very poor performance. However, $L_{chamfer}$ is unable to adjust the object shape and a poor shape estimation will result in low performance of the 6D pose. Therefore, L_{sdf} that can jointly adjust the pose and shape will play an important role. As shown in the second row of Table 3, when L_{sdf} is removed, the performance drops significantly. This indicates that the

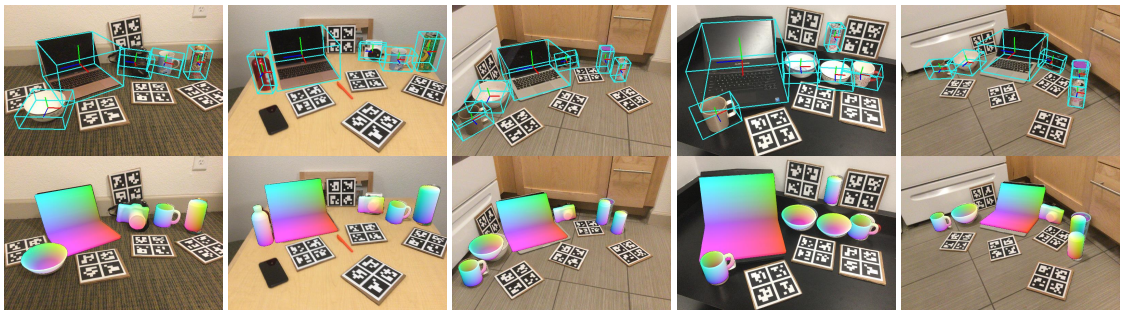


Figure 5: Qualitative results on NOCS-REAL test set. The top row shows the estimated pose and size of every object with the axis and tight bounding box. The bottom row shows its reconstructed model rendered on the corresponding RGB image.

Category	Chamfer Distance (mm)	
	Test-Directly	Self-Sup w/ Aug
Bottle	0.7023	**0.0602**
Bowl	0.4251	**0.0415**
Camera	1.0074	**0.3752**
Can	0.1092	**0.1004**
Mug	0.1089	**0.0335**
Laptop	1.2227	**0.4520**
Average	0.5959	**0.1771**

Table 4: Evaluations of the reconstructed models with Chamfer Distance metric on the NOCS-REAL test set.

reconstructed shape is very important for our self-supervised training. In addition, we also conduct an experiment to verify the benefits of the joint learning strategy. As shown in the third row of Table 3, the performance drops obviously if the L_{sup} is removed. This shows that the knowledge learned from the synthetic dataset is very useful for self-supervised learning on real-world data.

Evaluations of Shape Reconstruction: To evaluate the quality of the shape reconstruction, we compute the Chamfer Distance of the scaled reconstructed models with the ground truth 3D models. Table 4 shows the results for each categories. From this table, we can see that the average reconstruction error decreases significantly for all category after our self-supervised training, which indicates that our self-supervised framework is effective in improving the quality of the shape reconstruction.

4.4 Qualitative Results

Fig. 5 shows the qualitative results of our method on the NOCS-REAL test set. From the top row of Fig. 5, we can see that our method can accurately detect the objects and estimate the 6D pose and size for different categories even if no 3D annotations are provided for training in the real world. In addition, we also reconstruct models from the predicted shape latent vectors with the Marching Cubes and render them to the corresponding RGB images using the predicted poses and scale factors, as shown in the bottom row of Fig. 5. It can be seen that the rendered results are well aligned to their respective targets, which further indicates that our proposed loss functions can provide sufficient constraints for both pose estimation and shape reconstruction.

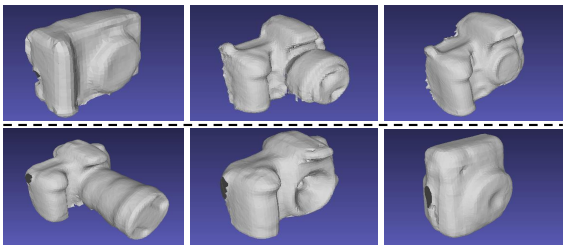


Figure 6: The ground truth 3D models of the *camera* category from the NOCS-REAL training set (top) and test set (bottom).

4.5 Limitations and Failure Cases

As shown in the middle figure of Fig. 4, the *camera* category shows poor performance in rotation estimation. This is mainly due to the larger geometry differences of the camera between the training and test set compared to other categories as shown in Fig. 6, which leads to poor performance on the test set even for some fully supervised methods. In addition, although our method can learn the intra-class variations, it is still difficult to accurately estimate the target shape when a large geometry difference occurs, which may bring about incorrect rotation guidance through the chamfer loss and result in local minima of rotation during training. Finally, since our prediction network is trained separately for each category, it requires more parameters when estimating multiple classes of targets at the same time.

5 Conclusion

We present a self-supervised approach for category-level 6D pose estimation, which can predict unseen object poses without pose annotations and exact 3D models in real scenarios for training. In our work, we leverage DeepSDF as a 3D object representation and train a DeepSDF decoder to learn the category-level shape prior. For self-supervised training, we design several self-supervised loss functions to make the pose and shape simultaneously optimized. Experiments show that our proposed self-supervised framework is effective for pose estimation and achieves comparable performance with the state-of-the-art fully supervised methods.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No.61873046 and No.U1708263).

References

Besl, P. J.; and McKay, N. D. 1992. Method for registration of 3-D shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, 586–606. International Society for Optics and Photonics.

Chen, D.; Li, J.; Wang, Z.; and Xu, K. 2020. 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*, 11973–11982.

Chen, W.; Jia, X.; Chang, H. J.; Duan, J.; Shen, L.; and Leonardis, A. 2021. FS-Net: Fast Shape-based Network for Category-Level 6D Object Pose Estimation with Decoupled Rotation Mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1581–1590.

Chen, Z.; and Zhang, H. 2019. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5939–5948.

Choy, C. B.; Xu, D.; Gwak, J.; Chen, K.; and Savarese, S. 2016. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, 628–644. Springer.

Ester, M.; Kriegel, H.-P.; Sander, J.; and Xu, X. 1996. Density-based spatial clustering of applications with noise. In *Int. Conf. Knowledge Discovery and Data Mining*, volume 240, 6.

Fan, H.; Su, H.; and Guibas, L. J. 2017. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 605–613.

Groueix, T.; Fisher, M.; Kim, V. G.; Russell, B.; and Aubry, M. 2018. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, Y.; Sun, W.; Huang, H.; Liu, J.; Fan, H.; and Sun, J. 2020. 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*, 11632–11641.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lin, J.; Wei, Z.; Li, Z.; Xu, S.; Jia, K.; and Li, Y. 2021. DualPoseNet: Category-level 6D Object Pose and Size Estimation using Dual Pose Network with Refined Learning of Pose Consistency. *arXiv preprint arXiv:2103.06526*.

Liu, S.; Zhang, Y.; Peng, S.; Shi, B.; Pollefeys, M.; and Cui, Z. 2020. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019–2028.

Lorensen, W. E.; and Cline, H. E. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics*, 21(4): 163–169.

Manhardt, F.; Wang, G.; Busam, B.; Nickel, M.; Meier, S.; Minciullo, L.; Ji, X.; and Navab, N. 2020. CPS++: Improving Class-level 6D Pose and Shape Estimation From Monocular Images With Self-Supervised Learning. *arXiv preprint arXiv:2003.05848*.

Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; and Geiger, A. 2019. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4460–4470.

Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; and Lovegrove, S. 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 165–174.

Park, K.; Mousavian, A.; Xiang, Y.; and Fox, D. 2020. Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10710–10719.

Peng, S.; Liu, Y.; Huang, Q.; Zhou, X.; and Bao, H. 2019. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4561–4570.

Pitteri, G.; Ramamonjisoa, M.; Ilic, S.; and Lepetit, V. 2019. On object symmetries and 6d pose estimation from images. In *2019 International Conference on 3D Vision (3DV)*, 614–622. IEEE.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Tian, M.; Ang, M. H.; and Lee, G. H. 2020. Shape prior deformation for categorical 6d object pose and size estimation. In *European Conference on Computer Vision*, 530–546. Springer.

Umeyama, S. 1991. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380.

Wang, C.; Xu, D.; Zhu, Y.; Martín-Martín, R.; Lu, C.; Fei-Fei, L.; and Savarese, S. 2019a. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3343–3352.

Wang, G.; Manhardt, F.; Shao, J.; Ji, X.; Navab, N.; and Tombari, F. 2020. Self6d: Self-supervised monocular 6d object pose estimation. In *European Conference on Computer Vision*, 108–125. Springer.

Wang, H.; Sridhar, S.; Huang, J.; Valentin, J.; Song, S.; and Guibas, L. J. 2019b. Normalized object coordinate space for

category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2642–2651.

Wang, N.; Zhang, Y.; Li, Z.; Fu, Y.; Liu, W.; and Jiang, Y.-G. 2018. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 52–67.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1912–1920.

Xiang, Y.; Schmidt, T.; Narayanan, V.; and Fox, D. 2017. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*.

Yang, Z.; Yu, X.; and Yang, Y. 2021. DSC-PoseNet: Learning 6DoF Object Pose Estimation via Dual-Scale Consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3907–3916.

Zakharov, S.; Shugurov, I.; and Ilic, S. 2019. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1941–1950.

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J. 2017. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2881–2890.

Zhou, Q.-Y.; Park, J.; and Koltun, V. 2018. Open3D: A Modern Library for 3D Data Processing. *arXiv:1801.09847*.