CatFormer: Category-Level 6D Object Pose Estimation with Transformer

Sheng Yu[1], **Di-Hua Zhai**[1,2*], **Yuanqing Xia**[1]

[1]School of Automation, Beijing Institute of Technology, Beijing, China
[2]Yangtze Delta Region Academy of Beijing Institute of Technology, Jiaxing, China
yusheng@bit.edu.cn, zhaidih@bit.edu.cn, xia_yuanqing@bit.edu.cn.

Abstract

Although there has been significant progress in category-level object pose estimation in recent years, there is still considerable room for improvement. In this paper, we propose a novel transformer-based category-level 6D pose estimation method called CatFormer to enhance the accuracy pose estimation. CatFormer comprises three main parts: a coarse deformation part, a fine deformation part, and a recurrent refinement part. In the coarse and fine deformation sections, we introduce a transformer-based deformation module that performs point cloud deformation and completion in the feature space. Additionally, after each deformation, we incorporate a transformer-based graph module to adjust fused features and establish geometric and topological relationships between points based on these features. Furthermore, we present an end-to-end recurrent refinement module that enables the prior point cloud to deform multiple times according to real scene features. We evaluate CatFormer's performance by training and testing it on CAMERA25 and REAL275 datasets. Experimental results demonstrate that CatFormer surpasses state-of-the-art methods. Moreover, we extend the usage of CatFormer to instance-level object pose estimation on the LINEMOD dataset, as well as object pose estimation in real-world scenarios. The experimental results validate the effectiveness and generalization capabilities of CatFormer. Our code and the supplemental materials are available at https://github.com/BIT-robot-group/CatFormer.

1 Introduction

6D object pose estimation is a crucial task in computer vision, with applications ranging from robotic grasping (Tremblay et al. 2018; Wang et al. 2019a) to 3D scene understanding (Chen et al. 2019) and augmented reality (Su et al. 2019). While previous methods have primarily focused on instance-level pose estimation, such as (Xiang et al. 2018; Kehl et al. 2017; Peng et al. 2019; He et al. 2020; Lin et al. 2022b; Rad and Lepetit 2017), these approaches heavily rely on the availability of a 3D model for accurate estimation. Consequently, when faced with an unknown object, it becomes challenging to accurately estimate its 6D pose, which significantly affects pose estimation in real-world scenes.

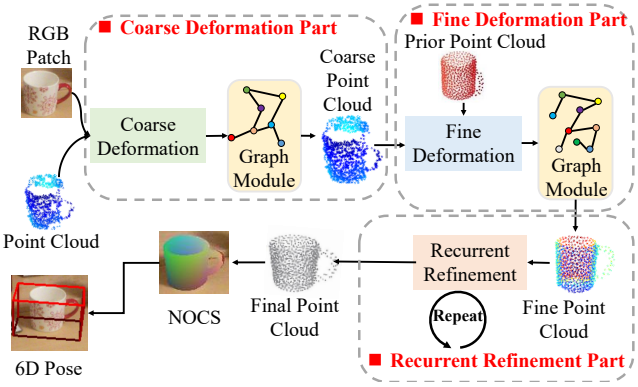

Figure 1: CateFormer mainly consists of three parts: coarse deformation, fine deformation, and recurrent refinement. The coarse deformation part is used to coarse deform and complement the point cloud. The fine deformation part is used to fine deform the prior point cloud. The recurrent refinement is used to recurrent refine the point cloud from the fine deformation part.

To solve this problem, researchers propose several model-independent methods for category-level object 6D pose estimation (Wang et al. 2019b; Tian, Ang, and Lee 2020; Chen et al. 2021, 2020a; Di et al. 2022; Lin et al. 2022a). Estimating the pose at the category level is more challenging than instance-level methods due to the lack of 3D models for objects. Some approaches, like (Wang et al. 2019b; Chen and Dou 2021), address this issue by introducing a "Normalized Object Coordinate Space" (NOCS) where they predict the 3D model of the object. Additionally, most methods rely on point clouds to capture the object's geometric structure. Recognizing the similarity in geometry among objects in the same category, certain methods, such as (Tian, Ang, and Lee 2020; Chen and Dou 2021; Lin et al. 2022a), utilize an average point cloud as prior knowledge, enabling rough estimation of the geometric information in the scene.

However, investigating how to handle intra-class object variety and accurately model objects based on prior point clouds is an important problem. Firstly, in real scenes, the camera's view can be disrupted, resulting in fragmented point cloud information. The key challenge is how to com-

*Corresponding author.

plete the point cloud with limited data. Additionally, deforming the prior point cloud appropriately to accurately fit the object in the scene is another crucial consideration. Furthermore, a single deformation may not suffice to accurately represent the object structure, necessitating multiple deformations. However, certain methods, like (Tian, Ang, and Lee 2020; Chen and Dou 2021; Chen et al. 2020a), have overlooked these issues. They directly input RGB images with point cloud information and combine features for pose estimation, leading to inaccurate predictions. Although some methods try to use feature fusion for improved pose estimation, a gap remains compared to the actual poses.

We introduce **CatFormer**, a transformer-based method for category-level 6D object pose estimation, aiming to address these challenges. Currently, there is a scarcity of transformer-based methods for category-level pose estimation, such as (Zou et al. 2022; Liu et al. 2023). Our proposed method leverages transformers and achieves SOTA performance on benchmark datasets.

As depicted in Figure 1, CatFormer comprises three main components: the coarse deformation part, the fine deformation part, and the recurrent refinement part. In the coarse and fine deformation parts, we introduce a transformer-based deformation module to deform and complement the point cloud, enabling a better fit with the target object in the scene. Additionally, we propose a transformer-based graph module to refine and adjust fused features, learning geometric relationships and topological information within the point cloud for improved understanding of the object's 3D structure. Furthermore, we propose an end-to-end multi-stage refinement method that utilizes RGB image and scene point cloud fusion features to guide multiple deformations of the fine deformed prior point cloud, resulting in a significantly better fit with the target object in the scene. Our proposed method demonstrates notable improvements over SOTA methods on the dataset, surpassing some existing SOTA methods by more than 10% in certain evaluation metrics. We have also successfully applied CatFormer to instance-level pose estimation and real object pose estimation.

In summary, the main contributions of this paper are summarized as follows:

- We propose a novel transformer-based deformation module to perform coarse deformation on the scene point cloud and fine deformation on the prior point cloud.
- A transformer-based graph module is proposed to help networks adjust fused features and construct geometric and topological relationships between points in point cloud features.
- We propose an end-to-end recurrent refinement module that guides the prior point cloud to perform multiple iterations of refinement based on the guide of the fusion features of RGB images and point cloud, so that the prior point cloud can largely fit the target object.

2 Related Works

2.1 Instance-Level 6D Object Pose Estimation

Recently, there has been extensive research on deep learning-based methods for instance-level 6D object pose estimation. These methods can be categorized into two groups: direct regression and keypoints correspondence. Direct regression methods, such as (Xiang et al. 2018; Kehl et al. 2017; Labbé et al. 2020; Li et al. 2018; Manhardt et al. 2019, 2018; Wang et al. 2019a), take RGB or RGB-D images as input and directly predict the 6D pose based on extracted features. While these methods are generally time-efficient, they may lack accuracy in certain cases. To address this, researchers propose keypoints correspondence methods, including (Li, Wang, and Ji 2019; Zakharov, Shugurov, and Ilic 2019; Park, Patten, and Vincze 2019; Peng et al. 2019). These methods predict predefined object coordinates or 2D keypoints and calculate the 6D pose using the Perspective-n-Point (PnP) algorithm (Lepetit, Moreno-Noguer, and Fua 2009) based on the correspondence between 2D and 3D points. Some approaches also utilize keypoint voting for 6D pose prediction, such as (He et al. 2020, 2021). However, the non-differentiable nature of PnP makes it challenging to apply this two-stage pipeline in tasks that require differentiable poses. Consequently, alternative techniques have been explored to learn the PnP step, such as (Di et al. 2021; Hu et al. 2020; Wang et al. 2021).

2.2 Category-Level 6D Object Pose Estimation

To address the limitations of instance-level methods, researchers have explored novel approaches that reduce reliance on 3D models. Category-level 6D object pose estimation methods have emerged in recent years, including (Wang et al. 2019b; Tian, Ang, and Lee 2020; Chen et al. 2021; Chen and Dou 2021; Lin et al. 2021; Chen et al. 2020a; Di et al. 2022; Lin et al. 2022a; You et al. 2022). In (Wang et al. 2019b), Wang et al. propose NOCS, a method that projects all objects into the same coordinate space and employs Umeyama's algorithm (Umeyama 1991) to calculate the 6D object pose. However, since objects within the same category can have distinct shapes, it is challenging to capture the shape variations between each object. To overcome this challenge, some methods introduce shape priors to mitigate the influence of shape variations (Tian, Ang, and Lee 2020; Chen and Dou 2021; Wang, Chen, and Dou 2021; Zou et al. 2022). For example, in (Tian, Ang, and Lee 2020), Tian et al. incorporate a prior point cloud during training and connect the features from the scene point cloud and RGB images. They then predict the NOCS coordinates of the target object and calculate its pose. In (Chen and Dou 2021), Chen et al. propose SGPA, which introduces a feature fusion module based on vision transformers (Vaswani et al. 2017) to merge the features from RGB images and point clouds. Some methods aim to directly predict the rotation, translation, and size of the object based on extracted features, as demonstrated in (Lin et al. 2021; Chen et al. 2021; Di et al. 2022).

Many methods try to deform the shape prior to fit the target object, such as (Tian, Ang, and Lee 2020; Lin et al. 2022a; Chen and Dou 2021; Wang, Chen, and Dou 2021; Zhang et al. 2022). Some of these methods leverage fused features to deform the shape prior, such as (Tian, Ang, and Lee 2020; Chen and Dou 2021; Wang, Chen, and Dou 2021). Others attempt to share the weights of the network during the feature extraction process for both the point cloud and

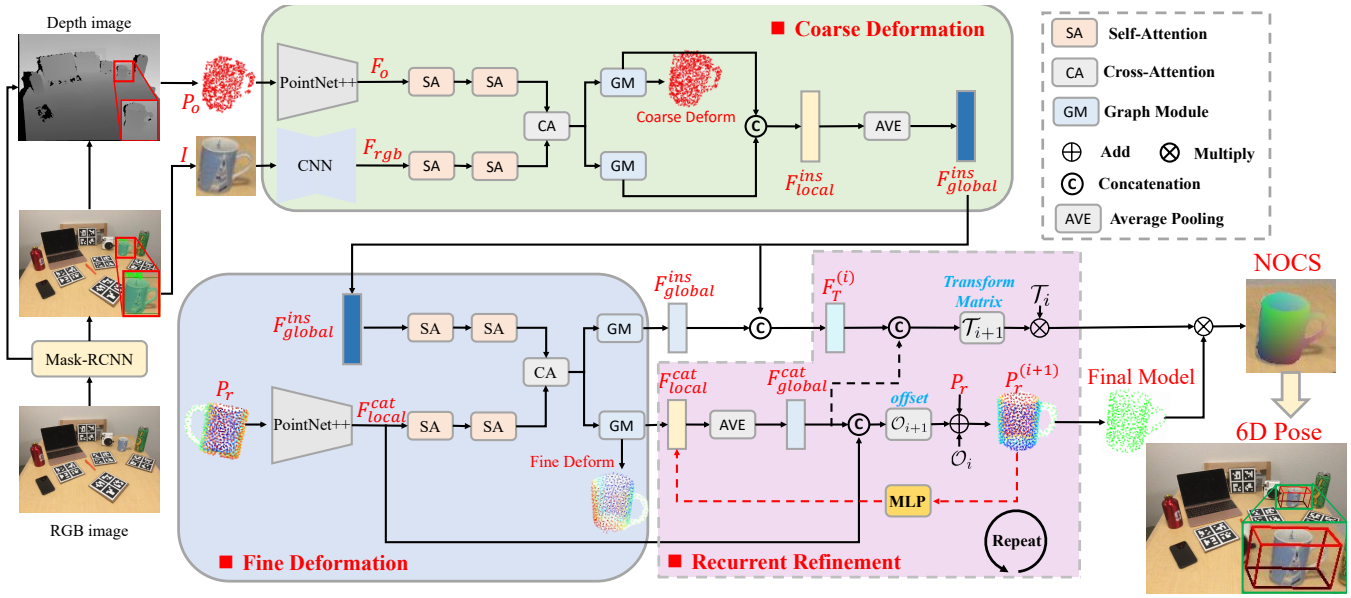

Figure 2: An overview of CatFormer for category-level 6D object pose estimation. We initially employ Mask-RCNN to predict the mask and category of the target object. CatFormer takes the object point cloud P_o, RGB image I, and prior point cloud P_r as inputs. Firstly, we utilize the coarse deformation module with the graph module to deform and complement P_o. Subsequently, employing the fine deformation module with the graph module and P_r generates relatively accurate point cloud features for the object. Ultimately, a recurrent refinement module is used to enhance the point cloud features, resulting in the final NOCS model of the object. Based on the predicted NOCS model, we generate the 6D pose of the object.

the shape prior, such as (Lin et al. 2022a; Zhang et al. 2022). However, in this paper, we propose a novel approach that differs from these methods. Instead of adopting the aforementioned ideas, our method utilizes a constructed feature graph and repetitive refinement to deform the shape prior.

3 Method

3.1 Pipeline Overview

The pipeline of CatFormer is provided in Figure 1. Initially, the input consists of the RGB image $I_o \in \mathbb{R}^{H\times W\times 3}$ and the point cloud $P_o \in \mathbb{R}^{N_o\times 3}$ of the target object. PSPNet (Zhao et al. 2017) is employed to extract features from the RGB image, resulting in $F_{rgb} \in \mathbb{R}^{N_o\times d}$. Simultaneously, PointNet++ (Qi et al. 2017) is used to extract features from the point cloud, yielding $F_o \in \mathbb{R}^{N_o\times d}$. Next, the deformation module is utilized to complement and deform the point cloud. Additionally, PointNet++ is applied to extract features from the prior point cloud $P_r \in \mathbb{R}^{N_r\times 3}$, generating $F_r \in \mathbb{R}^{N_r\times d}$. The fused feature from F_{rgb} and F_o is then employed to deform P_r within the fine deformation module. To capture geometrical information about the object and adjust the fused feature, a graph module establishes the graph feature after each point cloud deformation process. Finally, the recurrent refinement module is leveraged to further refine the prior point cloud.

3.2 Deformation Module

We propose a transformer-based deformation module consisting of self-attention (SA) and cross-attention (CA) modules to deform and complete the shape of the point cloud. The coarse deformation process applies coarse deformation to the scene point cloud, while the fine deformation process deforms the prior point cloud using the fused feature. This deformation process completes and deforms the point cloud shape in the feature space, resulting in feature maps of the deformed point cloud.

We begin by applying three MLP layers to F_{rgb} and F_o, generating the *query, key,* and *value* inputs for the SA module. which can be calculated by

$$\mathcal{F}_* = SA(F_*) \tag{1}$$

where $* \in \{rgb, o\}$ indicates the parameters of RGB and point cloud, respectively.

Next, we utilize another SA module based on $\mathcal{F}_{rgb}$ and $\mathcal{F}_o$ to perform further feature extraction using a similar operation. The resulting feature maps for the RGB images and point cloud are denoted as $\mathcal{F}_{rgb} \in \mathbb{R}^{N_o\times C}$ and $\mathcal{F}_o \in \mathbb{R}^{N_o\times C}$, respectively.

Then, we compute the *query, key,* and *value* inputs for the cross-attention (CA) module based on the extracted features $\mathcal{F}_{rgb}$ and $\mathcal{F}_o$. The objective of the CA model is to combine $\mathcal{F}_o$ and $\mathcal{F}_{rgb}$, resulting in the deformation offsets features $\mathcal{O}_o$ and $\mathcal{O}_{rgb}$. The fusion is computed as

$$\mathcal{O}_i = softmax\left(\frac{q_i^c \times (k_j^c)^T}{\sqrt{d_k}}\right) v_j^c \tag{2}$$

where $i, j \in \{rgb, o\}$ and $i \neq j$.

Finally, we adjust the feature maps by adding the deformation offsets, which can be calculated as $\mathcal{F}_* = F_* + \mathcal{O}_*$, where $* \in \{rgb, o\}$.

Once we have adjusted $\mathcal{F}_{rgb}$ and $\mathcal{F}_o$, we concatenate them along the channel dimension, resulting in instance local-wise features denoted as $\mathcal{F}^{ins}_{local} \in \mathbb{R}^{N_o \times \mathcal{C}}$. Subsequently, we employ an MLP layer to generate the instance global-wise features represented by $\mathcal{F}^{ins}_{global} \in \mathbb{R}^{N_o \times \mathcal{C}}$.

Both the coarse deformation and fine deformation modules function in a similar manner, as they deform objects by predicting the deformation offset in the feature space. For more detailed information about the fine deformation module, please refer to the supplementary material.

3.3 Graph Module

In order to establish a graph relationship between the fused features of RGB images and point clouds, we introduce a graph module inspired by ideas from graph convolution (Kipf and Welling 2017). This graph module incorporates a transformer structure, which is illustrated in Figure 3.

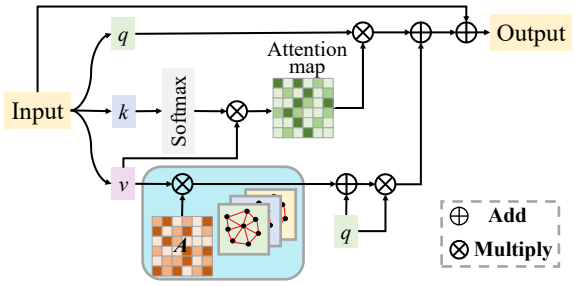

Figure 3: The structure of the graph module.

Using the fused feature $\mathcal{F} \in \mathbb{R}^{B \times N \times \mathcal{D}}$ as input, we generate the *query, key,* and *value*, which indicate by $q \in \mathbb{R}^{B \times N \times \mathcal{D}_q}$, $k \in \mathbb{R}^{B \times N \times \mathcal{D}_k}$, and $v \in \mathbb{R}^{B \times N \times \mathcal{D}_v}$ respectively, where B represents the batch size, and $\mathcal{D}_*, * \in \{q, k, v\}$ indicates the feature dimension.

Next, we normalize the q and v tensors, and apply the softmax function to the k tensor.

To generate multiple graph features, we first divide q into several heads, resulting in $q \in \mathbb{R}^{B \times N \times H \times (\mathcal{D}_q / H)}$, where H denotes the number of heads. We set $\mathcal{D}_q / H = \mathcal{D}_k = \mathcal{D}_v$ and utilize the Einstein summation convention (*einsum*) for performing matrix multiplication in the transformer.

Then, we follow the calculation process of the transformer and calculate the attention map, which can be got by $attn = k \otimes v$, where $\otimes$ indicates the *einsum*, $attn \in \mathbb{R}^{B \times \mathcal{D}_k \times \mathcal{D}_v}$ indicates the attention map.

Next, we generate the adjusted features with the attention map:

$$\mathcal{F}_{attn} = q \otimes attn \tag{3}$$

where $\mathcal{F}_{attn} \in \mathbb{R}^{B \times N \times H \times \mathcal{D}_v}$ indicates the adjusted feature map.

Then, we generate multi-graph features on the *value* branch. First, we create a random graph adjacency matrix A to establish a graph G, which can be computed by

$$G = v \otimes A \tag{4}$$

where $A \in \mathbb{R}^{\mathcal{D}_v \times \mathcal{D}_v \times \mathcal{D}_k}$, $G \in \mathbb{R}^{B \times N \times \mathcal{D}_k \times \mathcal{D}_v}$.

To establish a graph on itself, we add a self-loop to the v tensor by $v = v + I$, where I represents the identity matrix.

We reshape v into $\mathbb{R}^{B \times N \times \mathcal{D}_v \times \mathcal{D}_v}$ and add it to the graph G. Using G, we establish the relationship between v and q by

$$\mathcal{F}_G = q \otimes G \tag{5}$$

where $\mathcal{F}_G \in \mathbb{R}^{B \times N \times H \times \mathcal{D}_v}$ indicates the graph features.

Finally, we add the $\mathcal{F}_G$ and $\mathcal{F}_{attn}$ tensors together to obtain $\mathcal{F}_f$. Then, we can get the final graph feature $\mathcal{F}_{fin}$ by $\mathcal{F}_{fin} = MLP(\mathcal{F}_f) + \mathcal{F}$, where MLP indicates MLP layers.

Due to space limits, we have added more details and motivation of graph module into the supplementary material.

3.4 Recurrent Refinement Module

In this part, we deform the prior point cloud P_r by $\mathcal{F}^{ins}_{global}$ and $\mathcal{F}^{cat}_{global}$ in the feature space. We predict the deformation offset $\mathcal{O}$ and the point cloud transformation matrix T. We use $\mathcal{O}$ to adjust and deform the shape of P_r, while T generates the NOCS model from the deformed point cloud.

The initial transform matrix and deformation offset for objects are $T_0 \in \mathbb{R}^{N_o \times (N_r \times c)}$ and $\mathcal{O}_0 \in \mathbb{R}^{N_r \times (c \times 3)}$. Here, c represents the number of object categories.

For a specific object $k \in c$, the corresponding deformed point cloud is P_r. The initial point cloud transformation matrix and deformation offset are $T^{(k)}_0 \in \mathbb{R}^{N_o \times N_r}$ and $\mathcal{O}^{(k)}_0 \in \mathbb{R}^{N_r \times 3}$.

We perform the first deformation on P_r with the initial deformation offset, $P^{(1)}_r = P_r + \mathcal{O}^{(k)}_0$, where $P^{(1)}_r$ represents the deformed prior point cloud.

Then, we update the category local-wise and global-wise features based on the $P^{(1)}_r$ that is

$$\mathcal{F}^{cat}_{local(1)} = MLP(P^{(1)}_r), \mathcal{F}^{cat}_{global(1)} = AVE(\mathcal{F}^{cat}_{local}) \tag{6}$$

where AVE indicates average global pooling layer.

After that, we use $\mathcal{F}^{cat}_{global(1)}$ with previous features to further update the offset and transformation matrix:

$$\mathcal{O}_1 = MLP(\copyright(\mathcal{F}^{cat}_{global(1)}, \mathcal{F}^{cat}_{global(0)})) \tag{7}$$

$$T_1 = MLP(\copyright(\mathcal{F}^{cat}_{global(1)}, \mathcal{F}^{ins}_{global(0)})) \tag{8}$$

where $\copyright(*, *)$ indicates concatenation, $\mathcal{F}^{cat}_{global(0)} = \mathcal{F}^{cat}_{global}$ and $\mathcal{F}^{ins}_{global(0)} = \mathcal{F}^{ins}_{global}$.

Then we update the deformation offset and transformation matrix: $\mathcal{O}_1 \leftarrow \mathcal{O}_1 + \mathcal{O}_0$, $T_1 \leftarrow T_1 \times T_0$. By repeatedly performing such a process, we can deform the prior point cloud several times, which can be expressed as

$$\mathcal{O}_{i+1} = MLP(\copyright(\mathcal{F}^{cat}_{global(i+1)}, \mathcal{F}^{cat}_{global(0)})) \tag{9}$$

$$\mathcal{O}_{i+1} \leftarrow \mathcal{O}_{i+1} + \mathcal{O}_i \tag{10}$$

$$T_{i+1} = MLP(\copyright(\mathcal{F}^{cat}_{global(i+1)}, \mathcal{F}^{ins}_{global(i)})) \tag{11}$$

$$T_{i+1} \leftarrow T_{i+1} \times T_i \tag{12}$$

Due to space limits, we also provide the pseudocode of this algorithm in the supplementary material.

3.5 Loss Function

We can transform the instance object into the NOCS space using the final prediction of deformation offset $\mathcal{O}$ and point cloud transformation matrix T. The resulting equation is expressed as $P_{NOCS} = softmax(T) \times (P_r + \mathcal{O})$.

Correspondence Loss Based on P_{NOCS}, we can transform the object into NOCS space, we use the smooth L_1 loss to calculate the correspondence loss

$$L_{corr} = \mathcal{S}(P_{NOCS}, P_{gt})$$

where $\mathcal{S}$ indicates smooth L_1 loss, P_{gt} is the ground truth.

Reconstruction Loss To evaluate the performance of deformation of the network, following the ideas in (Tian, Ang, and Lee 2020), we use the Chamfer distance to penalize the deformation

$$L_r(M, M_{gt}) = \sum_{i \in M^i} \min_{j \in M^j_{gt}} ||i-j||_2^2 + \sum_{j \in M^j_{gt}} \min_{i \in M^i} ||i-j||_2^2$$

where $M = P_r + \mathcal{O}$ is the prediction of the object 3D model, M_{gt} is the ground truth 3D model.

Distribution Loss Following (Tian, Ang, and Lee 2020), We also try to encourage T to be a peaked distribution by minimizing the average cross-entropy loss, which can be calculated by

$$L_{dis} = \frac{1}{N_o} \sum_i \sum_j (-softmax(T_{i,j}) \log(softmax(T_{i,j})))$$

Deformation Loss To avoid overfitting and large deformations, we also use the $\mathcal{O}$ with L_2 regularization to realize it, $L_{def} = \frac{1}{N_o} \sum_{i \in \mathcal{O}} ||i||_2$

Total Loss The total loss of the CatFormer is the sum of the four losses

$$L = \sum_{k=0}^{n} \lambda_{corr} L_{corr} + \lambda_r L_r + \lambda_{dis} L_{dis} + \lambda_{def} L_{def}$$

where λ_* is the weight of the loss function, n is the number of the object in the scene.

4 Experiments

4.1 Datasets

The benchmark datasets for category-level object pose estimation are the REAL275 dataset and CAMERA25 dataset, proposed in (Wang et al. 2019b). The CAMERA25 dataset consists of 300K images, with 25K images used for evaluation. It is generated by rendering synthetic objects into real scenes. On the other hand, the REAL275 dataset contains 4300 real-world training images from 7 scenes, and 2750 real-world evaluation images from 6 scenes. Both datasets include 6 different categories of objects: bottle, bowl, camera, can, laptop, and mug. For instance-level 6D pose estimation, the LINEMOD dataset (Hinterstoisser et al. 2011) serves as the benchmark. It comprises 13 different objects randomly placed in real scenes.

4.2 Training Details

All experiments are conducted on a single NVIDIA GeForce RTX 3090 GPU with 24 GB memory, running Ubuntu 18.04 as the operating system. PyTorch 1.8.1 is utilized as the deep learning framework, and CUDA 11.1 is employed for accelerated training. The network is trained with a batch size of 16 for 60 epochs. The initial learning rate is set to 1×10^{-4}, gradually decreasing to 1×10^{-6}. We also set $\lambda_{corr} = \lambda_r = \lambda_{def} = 1$ in this paper.

4.3 Preprocessing

To process the dataset, we follow the procedures outlined in (Tian, Ang, and Lee 2020). We employ an instance segmentation network, such as Mask-RCNN (He et al. 2017), for object detection and segmentation. For each segmented instance, we crop the object and resize the image to 192×192 pixels. Using the RGB-D images and the camera's intrinsic matrix, we generate a point cloud of the scene. From this point cloud, we randomly select 1024 points for each object. The cropped images and selected points are then used as inputs for CatFormer.

4.4 Evaluation Metrics

We evaluate the performance of CatFormer using widely used evaluation metrics (Wang et al. 2019b; Tian, Ang, and Lee 2020; Lin et al. 2021; Di et al. 2022; Lin et al. 2022a). For rotation and translation evaluation, we utilize 3D Intersection-Over-Union (IoU) with thresholds of 0.25, 0.5, and 0.75. Additionally, we employ $5°2cm$, $5°5cm$, $10°2cm$, and $10°5cm$ to directly assess rotation and translation accuracy. If the errors fall within the thresholds, the predictions are deemed correct. Based on these evaluation metrics, we will use overall mAP to assess the performance of CatFormer compared to other SOTA methods.

4.5 Comparison with State-of-the-Art Methods

We present the quantitative results of CatFormer compared to recent state-of-the-art methods on the CAMERA25 and REAL275 datasets in Table 1.

CatFormer exhibits slightly superior performance to SOTA methods on the CAMERA25 dataset. This can be attributed to the accurate depth images generated in a simulated real scene, unaffected by environmental factors. Consequently, point cloud completion has not provided significant enhancements in pose estimation. However, on the REAL275 dataset, where all images are captured in the real world, the depth images can be influenced by object reflections or lighting conditions, resulting in incomplete information and imperfect point clouds. In such cases, transformer-based point cloud completion and deformation have demonstrated their effectiveness. Notably, for the IoU_{50} metric, CatFormer outperforms SOTA HS-Pose (Zheng et al. 2023) with a score of 83.1 compared to 82.1. Specifically, in the most challenging $5°2cm$ metric, CatFormer achieves a score of 47.7, surpassing the previous SOTA method HS-Pose at 46.5, indicating higher accuracy. Additionally, for the $10°2cm$ metric, CatFormer also demonstrates improved performance with a score of 69.0 compared to HS-Pose's 68.6.

Method	CAMERA25						REAL275						
	IoU_{50}	IoU_{75}	$5°2cm$	$5°5cm$	$10°2cm$	$10°5cm$	IoU_{25}	IoU_{50}	IoU_{75}	$5°2cm$	$5°5cm$	$10°2cm$	$10°5cm$
NOCS(Wang et al. 2019b)(R)	83.9	69.5	32.3	40.9	48.2	64.6	**84.9**	78.0	30.1	7.2	9.5	13.8	25.2
SPD(Tian, Ang, and Lee 2020)(R)	93.2	83.1	54.3	59.0	73.3	81.5	83.0	77.3	53.2	19.2	21.4	43.2	54.1
SGPA(Chen and Dou 2021)(R)	93.2	88.1	70.7	74.5	82.7	88.4	-	80.1	61.9	35.9	39.6	61.3	70.7
CR-Net(Wang, Chen, and Dou 2021)(R)	**93.8**	88.0	72.0	76.4	81.0	87.7		79.3	55.9	27.8	34.3	47.2	60.8
FS-Net*(Chen et al. 2021)(D)	-	-	-	-	-	-	84.0	81.1	63.5	19.9	33.9	-	69.1
DualPoseNet(Lin et al. 2021)(R)	92.4	86.4	64.7	70.7	77.2	84.7	-	79.8	62.2	29.3	35.9	50.0	66.8
SAR-Net(Lin et al. 2022a)(D)	86.8	79.0	66.7	70.9	75.3	80.3	-	79.3	62.4	31.6	42.3	50.3	68.3
GPV-Pose(Di et al. 2022)(D)	93.4	88.3	72.1	79.1	-	89.0	84.2	83.0	64.4	32.0	42.9	-	73.3
HS-Pose(Zheng et al. 2023)(D)	93.3	89.4	73.3	**80.5**	80.4	89.4	84.2	82.1	**74.7**	46.5	**55.2**	68.6	**82.7**
SPD+GM	93.6	87.9	61.4	66.3	78.0	85.4	83.6	80.5	59.5	20.4	24.4	47.7	58.7
CR-Net+GM	**93.9**	88.8	72.8	76.0	82.4	88.3	83.2	79.8	64.5	32.9	37.5	53.1	63.4
SGPA+GM	93.5	86.1	73.8	77.7	83.9	89.0	84.0	81.3	65.3	36.5	40.6	62.9	71.1
Ours(R)	93.5	**89.9**	**74.9**	79.8	**85.3**	**90.2**	84.3	**83.1**	73.8	**47.7**	53.7	**69.0**	79.5

Table 1: Comparison with state-of-the-art methods on CAMERA25 dataset and REAL275 dataset. GM indicates the graph module. R and D indicates RGB-D-based and depth-based methods respectively. *We use the results provided by the GPV-Pose (Di et al. 2022) and HS-Pose (Zheng et al. 2023).

Since CatFormer is a RGB-D-based method, we comapre it to the other RGB-D-based methods. Figure 4 shows the qualitative analysis of CatFormer and the state-of-the-art RGB-D based method SGPA on the REAL275 dataset. In comparison to SGPA, CatFormer exhibits higher accuracy in pose estimation. We provide more comprehensive details of the comparison experiments and object pose estimation in the the supplementary material.

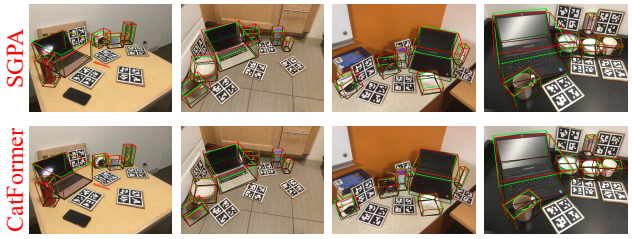

Figure 4: Qualitative results of SGPA and CatFormer. The bounding box in green line is the ground truth, and the red line is the prediction.

In this paper, we propose an innovative transformer-based graph module. By incorporating the graph module into CatFormer, we observe a significant improvement in performance, highlighting its effectiveness. To further validate the efficacy of the graph module, we also apply it to other RGB-D-based methods such as SPD, CR-Net, SGPA. The experimental results are presented in Table 1. The experimental results demonstrate that incorporating the graph module into the network leads to improved performance. For instance, when we apply the graph module after RGB-D future fusion in SPD, the $5°2cm$ metric shows a significant enhancement from 54.3 to 61.4, indicating a substantial improvement.

4.6 Ablation Studies

The REAL275 dataset, being more challenging than the CAMERA25 dataset, provides a better evaluation of the network's performance. Therefore, we primarily utilize the REAL275 dataset for conducting ablation studies. These studies focus on evaluating the proposed module, the number of refinement iterations, and the loss terms.

Module: We begin by conducting ablation studies on the proposed module, with the corresponding experimental results presented in Table 2(A). Firstly, when the graph module is removed, CatFormer's performance decreases, indicating the effectiveness of the graph module in establishing connections between different features. However, removing either the coarse deformation or fine deformation module causes a more significant drop in CatFormer's performance, highlighting the effectiveness of point cloud deformation and completion.

Repeat Times: In addition, we assess the efficacy of the recurrent refinement module by refining the initial point cloud multiple times. Specifically, we conduct refinements 1, 3, 4, 5, 6, and 7 times in this study, with the corresponding experimental results presented in Table 2(B). As the number of refinement iterations increases, there is a gradual improvement in the network's performance, reaching its peak at five iterations. However, further increasing the number of iterations leads to a decline in the network's performance.

Loss Terms: Given the necessity of predicting the NOCS model of the object, the L_{corr} term plays a crucial role. The experimental results are summarized in Table 2(C). It is evident from the results that relying solely on L_{corr} yields poor network performance. However, when L_{corr} is combined with L_{dis}, CatFormer demonstrates relatively improved performance.

Additionally, removing L_{dis} results in a significant decrease in network performance. Each row in matrix T can be considered as a relaxed one-hot vector, allowing a point in the NOCS space to be transformed by up to three points in the point cloud. We aim for T to have a peaked distribution, focusing on high-confidence transformations. This concentration of confidence enhances the accuracy of the predicted NOCS model.

Lastly, we utilize L_{def} to mitigate excessive deformation

Group	Module			Repeat Times	Loss Terms				REAL275					
	CD	FD	GM		L_{corr}	L_r	L_{dis}	L_{def}	IoU_{50}	IoU_{75}	$5°2cm$	$5°5cm$	$10°2cm$	$10°5cm$
(A)	✓			3	✓	✓	✓	✓	79.0	54.1	32.5	35.9	60.1	69.8
		✓		3	✓	✓	✓	✓	81.2	65.9	38.9	42.7	62.9	72.9
			✓	3	✓	✓	✓	✓	80.0	51.7	29.8	33.4	59.6	69.7
	✓	✓		3	✓	✓	✓	✓	79.8	67.7	37.7	42.5	63.9	74.0
		✓	✓	3	✓	✓	✓	✓	81.7	66.2	40.3	44.3	64.5	74.6
	✓		✓	3	✓	✓	✓	✓	80.9	56.0	34.9	38.5	61.1	71.9
(B)	✓	✓	✓	1	✓	✓	✓	✓	82.5	68.0	43.0	47.6	65.7	76.0
	✓	✓	✓	3	✓	✓	✓	✓	82.6	69.5	44.5	47.6	65.3	76.7
	✓	✓	✓	4	✓	✓	✓	✓	82.8	71.6	46.8	50.1	67.2	78.4
	✓	✓	✓	5	✓	✓	✓	✓	**83.1**	**73.8**	**47.7**	**53.7**	**69.0**	**79.5**
	✓	✓	✓	6	✓	✓	✓	✓	82.4	70.6	46.2	49.5	68.4	78.3
	✓	✓	✓	7	✓	✓	✓	✓	82.1	68.1	43.3	48.0	66.8	77.0
(C)	✓	✓	✓	3	✓			✓	74.9	34.9	41.6	46.0	66.0	75.4
	✓	✓	✓	3	✓	✓		✓	78.9	44.9	41.9	46.8	64.8	74.8
	✓	✓	✓	3	✓		✓	✓	82.2	68.0	42.0	47.1	64.3	75.4
	✓	✓	✓	3	✓				1.5	1.0	30.5	34.7	50.8	56.3
	✓	✓	✓	3	✓	✓			0.0	0.0	3.3	3.3	14.2	14.6
	✓	✓	✓	3	✓		✓		81.9	67.3	38.5	43.3	61.5	71.6
	✓	✓	✓	3	✓	✓	✓		81.8	68.1	42.0	47.1	63.8	75.4

Table 2: Ablation studies on different configurations of network on REAL275. CD, FD, and GM refer to coarse deformation module, fine deformation module, and graph module respectively.

in CatFormer's predictions. Applying L_{def} leads to an improvement in CatFormer's performance.

Due to space limits, we add more analysis of ablation studies in supplementary materials.

4.7 Category-level Object Pose Estimation in The Real World

To assess CatFormer's effectiveness and generalization, we apply it to real-world object pose estimation, specifically on objects that were not used during training. RGB-D images are obtained using an Intel RealSense D435i camera, while segmentation is performed using a pretrained Mask-RCNN model. The pose estimation results, shown in Figure 5, indicate that CatFormer exhibits good generalization and performs well in real-world object pose estimation tasks.

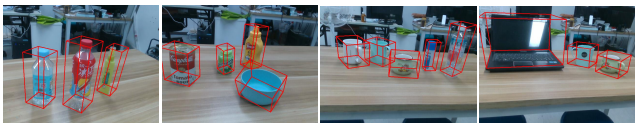

Figure 5: Object pose estimation results of CatFormer in the real world.

4.8 Instance-Level Object Pose Estimation

CatFormer is also applied to instance-level object pose estimation in this study. The LINEMOD dataset (Hinterstoisser et al. 2011) is used for conducting the instance-level object pose estimation experiment. By comparing CatFormer's performance with other state-of-the-art methods on the LINEMOD dataset, the experimental results, displayed in Table 3, indicate that CatFormer outperforms related category-level methods and achieves competitive performance compared to some state-of-the-art instance-level methods. For symmetric objects, we utilize ADD-S as the metric (Xiang et al. 2018), while for non-symmetric objects, we employ ADD as the metric (Hinterstoisser et al. 2012).

We also provide more details of instance-level object pose estimation in the supplementary material.

Method	C.L.	ADD-(S)	Speed(FPS)
PVNet(Peng et al. 2019)		86.3	27
G2L-Net(Chen et al. 2020b)		98.7	24
DenseFusion(Wang et al. 2019a)		94.3	15
PVN3D(He et al. 2020)		**99.4**	5
DualPose(Lin et al. 2021)	✓	98.2	3
FS-Net (Chen et al. 2021)	✓	97.6	22
GPV-Pose(Di et al. 2022)	✓	98.2	20
Ours	✓	**99.3**	8

Table 3: Instance-level object pose estimation on LINEMOD dataset. C.L. indicates category-level method.

5 Conclusion

In this paper, we introduce CatFormer, a novel category-level 6D object estimation network. CatFormer leverages transformer-based deformation modules for both coarse and fine deformations. Additionally, we propose a graph module to establish and extract graph features from fused features. To further refine the prior point's proximity to the target object, we introduce a recurrent refinement module. Compared to previous state-of-the-art methods, our approach demonstrates superior performance on both the CAMERA25 dataset and REAL275 dataset. Furthermore, CatFormer achieves successful results in instance-level object pose estimation and real-world object pose estimation tasks.

Acknowledgments

The work was supported by the National Natural Science Foundation of China under Grant 62173035, Grant 61803033 and Grant 61836001, and in part by the Xiaomi Young Scholars from Xiaomi Foundation, and in part by the BIT Research and Innovation Promoting Project under Grant 2023YCXY035.

References

Chen, D.; Li, J.; Wang, Z.; and Xu, K. 2020a. Learning canonical shape space for category-level 6D object pose and size estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 11973–11982.

Chen, K.; and Dou, Q. 2021. SGPA: Structure-guided prior adaptation for category-level 6D object pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2773–2782.

Chen, W.; Jia, X.; Chang, H. J.; Duan, J.; and Leonardis, A. 2020b. G2l-net: Global to local network for real-time 6d pose estimation with embedding vector features. In *IEEE Conference on Computer Vision and Pattern Recognition*, 4233–4242.

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 *IEEE Conference on Computer Vision and Pattern Recognition*, 1581–1590.

Chen, Y.; Huang, S.; Yuan, T.; Qi, S.; Zhu, Y.; and Zhu, S.-C. 2019. Holistic++ scene understanding: Single-view 3d holistic scene parsing and human pose estimation with human-object interaction and physical commonsense. In *IEEE International Conference on Computer Vision*, 8648–8657.

Di, Y.; Manhardt, F.; Wang, G.; Ji, X.; Navab, N.; and Tombari, F. 2021. So-pose: Exploiting self-occlusion for direct 6d pose estimation. In *IEEE International Conference on Computer Vision*, 12396–12405.

Di, Y.; Zhang, R.; Lou, Z.; Manhardt, F.; Ji, X.; Navab, N.; and Tombari, F. 2022. GPV-Pose: Category-level Object Pose Estimation via Geometry-guided Point-wise Voting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 6781–6791.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask R-CNN. In *IEEE International Conference on Computer Vision*, 2961–2969.

He, Y.; Huang, H.; Fan, H.; Chen, Q.; and Sun, J. 2021. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3003–3013.

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 *IEEE Conference on Computer Vision and Pattern Recognition*, 11632–11641.

Hinterstoisser, S.; Holzer, S.; Cagniart, C.; Ilic, S.; Konolige, K.; Navab, N.; and Lepetit, V. 2011. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *IEEE International Conference on Computer Vision*, 858–865.

Hinterstoisser, S.; Lepetit, V.; Ilic, S.; Holzer, S.; Bradski, G.; Konolige, K.; and Navab, N. 2012. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*, 548–562.

Hu, Y.; Fua, P.; Wang, W.; and Salzmann, M. 2020. Single-stage 6d object pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2930–2939.

Kehl, W.; Manhardt, F.; Tombari, F.; Ilic, S.; and Navab, N. 2017. SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again. In *IEEE International Conference on Computer Vision*, 1521–1529.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Labbé, Y.; Carpentier, J.; Aubry, M.; and Sivic, J. 2020. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, 574–591.

Lepetit, V.; Moreno-Noguer, F.; and Fua, P. 2009. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2): 155–166.

Li, Y.; Wang, G.; Ji, X.; Xiang, Y.; and Fox, D. 2018. Deepim: Deep iterative matching for 6d pose estimation. In *European Conference on Computer Vision*, 683–698.

Li, Z.; Wang, G.; and Ji, X. 2019. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *IEEE International Conference on Computer Vision*, 7678–7687.

Lin, H.; Liu, Z.; Cheang, C.; Fu, Y.; Guo, G.; and Xue, X. 2022a. SAR-Net: Shape Alignment and Recovery Network for Category-Level 6D Object Pose and Size Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 6707–6717.

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. In *IEEE International Conference on Computer Vision*, 3560–3569.

Lin, S.; Wang, Z.; Ling, Y.; Tao, Y.; and Yang, C. 2022b. E2EK: End-to-End Regression Network Based on Keypoint for 6D Pose Estimation. *IEEE Robotics and Automation Letters*, 7(3): 6526–6533.

Liu, J.; Sun, W.; Liu, C.; Zhang, X.; and Fu, Q. 2023. Robotic Continuous Grasping System by Shape Transformer-Guided Multiobject Category-Level 6-D Pose Estimation. *IEEE Transactions on Industrial Informatics*, 19(11): 11171–11181.

Manhardt, F.; Arroyo, D. M.; Rupprecht, C.; Busam, B.; Birdal, T.; Navab, N.; and Tombari, F. 2019. Explaining the ambiguity of object detection and 6d pose from visual data. In *IEEE International Conference on Computer Vision*, 6841–6850.

Manhardt, F.; Kehl, W.; Navab, N.; and Tombari, F. 2018. Deep model-based 6d pose refinement in rgb. In *European Conference on Computer Vision*, 800–815.

Park, K.; Patten, T.; and Vincze, M. 2019. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *IEEE International Conference on Computer Vision*, 7668–7677.

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

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems*, 30.

Rad, M.; and Lepetit, V. 2017. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In *IEEE International Conference on Computer Vision*, 3828–3836.

Su, Y.; Rambach, J.; Minaskan, N.; Lesur, P.; Pagani, A.; and Stricker, D. 2019. Deep multi-state object pose estimation for augmented reality assembly. In *IEEE International Symposium on Mixed and Augmented Reality Adjunct*, 222–227.

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.

Tremblay, J.; To, T.; Sundaralingam, B.; Xiang, Y.; Fox, D.; and Birchfield, S. 2018. Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects. In *Conference on Robot Learning*.

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

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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 *IEEE Conference on Computer Vision and Pattern Recognition*, 3343–3352.

Wang, G.; Manhardt, F.; Tombari, F.; and Ji, X. 2021. Gdrnet: Geometry-guided direct regression network for monocular 6d object pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 16611–16621.

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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2642–2651.

Wang, J.; Chen, K.; and Dou, Q. 2021. Category-level 6d object pose estimation via cascaded relation and recurrent reconstruction networks. In *IEEE International Conference on Intelligent Robots and Systems*, 4807–4814.

Xiang, Y.; Schmidt, T.; Narayanan, V.; and Fox, D. 2018. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In *Robotics: Science and Systems*.

You, Y.; Shi, R.; Wang, W.; and Lu, C. 2022. CPPF: Towards Robust Category-Level 9D Pose Estimation in the Wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 6866–6875.

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

Zhang, R.; Di, Y.; Lou, Z.; Manhardt, F.; Tombari, F.; and Ji, X. 2022. Rbp-pose: Residual bounding box projection for category-level pose estimation. In *European Conference on Computer Vision*, 655–672.

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J. 2017. Pyramid scene parsing network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2881–2890.

Zheng, L.; Wang, C.; Sun, Y.; Dasgupta, E.; Chen, H.; Leonardis, A.; Zhang, W.; and Chang, H. J. 2023. HS-Pose: Hybrid Scope Feature Extraction for Category-level Object Pose Estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17163–17173.

Zou, L.; Huang, Z.; Gu, N.; and Wang, G. 2022. 6d-vit: Category-level 6d object pose estimation via transformer-based instance representation learning. *IEEE Transactions on Image Processing*, 31: 6907–6921.