applied sciences

MDPI

Article

Refined Prior Guided Category-Level 6D Pose Estimation and Its Application on Robotic Grasping

Huimin Sun, Yilin Zhang, Honglin Sun and Kenji Hashimoto *

Graduate School of Information, Production and Systems, Waseda University, Kitakyushu 808-0135, Japan; hannah_sun@akane.waseda.jp (H.S.); zhangyilin@moegi.waseda.jp (Y.Z.); hsun@akane.waseda.jp (H.S.)
* Correspondence: kenji.hashimoto@waseda.jp

Abstract: Estimating the 6D pose and size of objects is crucial in the task of visual grasping for robotic arms. Most current algorithms still require the 3D CAD model of the target object to match with the detected points, and they are unable to predict the object's size, which significantly limits the generalizability of these methods. In this paper, we introduce category priors and extract high-dimensional abstract features from both the observed point cloud and the prior to predict the deformation matrix of the reconstructed point cloud and the dense correspondence between the reconstructed and observed point clouds. Furthermore, we propose a staged geometric correction and dense correspondence refinement mechanism to enhance the accuracy of regression. In addition, a novel lightweight attention module is introduced to further integrate the extracted features and identify potential correlations between the observed point cloud and the category prior. Ultimately, the object's translation, rotation, and size are obtained by mapping the reconstructed point cloud to a normalized canonical coordinate system. Through extensive experiments, we demonstrate that our algorithm outperforms existing methods in terms of performance and accuracy on commonly used benchmarks for this type of problem. Additionally, we implement the algorithm in robotic arm-grasping simulations, further validating its effectiveness.

Keywords: pose estimation; robotic grasping; grasp detection; channel attention; scene understanding

check for updates

Citation: Sun, H.; Zhang, Y.; Sun, H.; Hashimoto, K. Refined Prior Guided Category-Level 6D Pose Estimation and Its Application on Robotic Grasping. *Appl. Sci.* **2024**, *14*, 8009. https://doi.org/10.3390/app14178009

Academic Editor: Keun-Chang Kwak

Received: 20 August 2024
Revised: 4 September 2024
Accepted: 4 September 2024
Published: 7 September 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Robotic arm grasping is widely applied in both industrial production and daily life. Before planning a grasp, it is crucial to accurately estimate the object's translation, rotation, and size to effectively set the values for the various joints of the robotic arm and gripper. Many studies have attempted to perform grasp detection by predicting planar grasping boxes [1,2]. However, these planar pose estimations are conducted only in 2D planes, which leads to the loss of many potentially effective grasps in 3D space. The 6D pose estimation, which takes RGB images or RGB images combined with depth images as inputs, predicts the object's translation, rotation, and the size of the 3D bounding box in the camera coordinate system. Therefore, it is more suitable for robotic arm-grasping tasks. Moreover, 6D pose estimation, as a critical problem in computer vision, is also widely applied in 3D reconstruction, augmented reality, and virtual reality [3].

Previous studies have shown that applying 6D pose estimation to robotic arm grasping is important [4,5]. Significant progress has been made in 6D pose estimation, including effectively addressing occlusion issues [6]. However, these algorithms require prior knowledge of the object's CAD model and cannot predict the pose of unknown objects, which significantly limits the applicability of instance-level 6D pose estimation algorithms. Category-level 6D pose estimation algorithms can predict the pose of different objects within the same category, overcoming the limitations of known models. Densefusion [4] was proposed, which uses a pointwise approach for 6D pose estimation and applies it to robotic arm grasping. This method, which directly regresses the object's translation and

rotation through a network, overlooks the object's geometric features. Wang [7] proposed the Normalized Object Coordinate Space (NOCS), a shared canonical representation for all objects within the same category. In NOCS, each vertex of an object is scaled into a unit-length cube, with all objects within the same category aligned in orientation and centered. The authors also provided the dense correspondence between the NOCS coordinates and the real point cloud. Therefore, if the NOCS coordinates of an object can be accurately predicted, the real point cloud image can be obtained. Since NOCS coordinates are within a unit-length cube, they are more conducive to neural network regression. Building on NOCS, further research [8] introduced category priors as input. The network learns feature information from the observed point cloud and the prior, deforms the prior point cloud, and maps it to NOCS through dense correspondence to estimate the 6D object pose and size.

Despite the significant progress made by previous researchers in this area, we observe that these studies primarily focus on the extraction and fusion of feature information from the observed object and the prior point clouds [9,10], while lacking explicit deformation of the point clouds. Furthermore, deforming the prior point cloud and establishing correspondence with the NOCS model is a dense regression problem. The direct regression of high-dimensional features through a single network can easily lead to information loss and local optima. Although [11] attempts have been made to compensate for the missing feature information, the lack of RGB image data reduces the network's generalization capability. Moreover, the absence of RGB guidance makes the results susceptible to interference when the target object is occluded, which is particularly dangerous in robotic arm-grasping tasks.

To address these issues, we propose a grasping detection method based on refined prior-guided category-level 6D object pose estimation. This method extracts the 6D pose of the target object from the category-level pose estimation network, which includes the object's translation and rotation in 3D space, as well as the object's 3D bounding box. We introduce category priors to predict the offsets between the prior point cloud and the observed object point cloud. The deformed point cloud is then mapped to the NOCS coordinate system through dense correspondence. Finally, the RANSAC algorithm [12] is used to account for outliers, and the Umeyama algorithm [13] is employed to solve the rigid transformation, including translation, rotation, and the object's size. We refine the deformation network by using a multi-stage network structure, explicitly incorporating the deformed point cloud into the dense correspondence matrix regression network, enhancing the network's robustness. Additionally, we propose a new attention structure that focuses the network's attention on the differences between point cloud features, addressing the intra-class variation problem:

- We propose a refined prior-guided category-based 6D pose estimation framework. Based on RGB and depth images, this framework effectively handles occlusion and lighting variations and can estimate the pose of unseen objects within the same category.
- We refine high-dimensional feature information through a multi-stage structure and refine the deformation results of the point cloud. By explicitly incorporating the transformation of the point cloud, the learned features gain stronger semantic information.
- We introduce a novel attention mechanism, enabling the network to allocate more focus on the differences between point clouds, thereby addressing the intra-class variation problem.
- Extensive experiments on the CAMERA25 and REAL275 datasets demonstrate that our method outperforms existing approaches and is capable of estimating the 6D pose of objects under partial occlusion and varying lighting conditions. Furthermore, the effectiveness of our algorithm is also tested and validated in vision-based robotic arm-grasping experiments.

2. Related Works

2.1. Manipulator Grasping

As 6D pose estimation is a crucial component of robotic arm grasping, numerous studies have focused on enhancing the accuracy of pose estimation to improve the performance

of robotic grasping. Ref. [5] proposed a method for semantic grasping and pose estimation of household objects through deep learning. The core of this approach involves predicting key points from RGB images using a deep learning network, which are then matched with key points on the 3D model using the PnP algorithm, directly regressing the 6D pose of the object. This algorithm demonstrates strong robustness even in cluttered environments. Ref. [4] introduced depth images as input, proposing an end-to-end 6D pose detection algorithm and applying it to robotic grasping. DenseFusion first extracts features from RGB and depth images as input, using a pointwise approach to fuse the color and depth features of each point. The model then estimates a local 6D pose candidate for each pixel, and these per-point pose candidates are subsequently aggregated to generate a global, accurate object pose estimation.

2.2. Instance-Level 6D Pose Estimation

Instance-level 6D pose estimation aims to predict the translation and rotation of an object based on its precise 3D CAD mode l [14,15]. Methods such as [16–18] compare feature points extracted from all pixels within the region of interest with those on the CAD model to obtain 2D–3D correspondences, after which the 6D pose of the target object is calculated using the Perspective-n-Point (PnP) algorithm [19]. Other approaches like [9,20,21] render a series of images from different angles using the 3D CAD model, and then apply a sliding window algorithm or voting mechanism to match the RGB image with the generated renderings, selecting the highest similarity match as the object pose. Additionally, methods like [22,23] directly regress the 6D pose of the object through neural networks.

2.3. Category-Level 6D Pose Estimation

Many methods do not utilize prior point clouds. Refs. [24,25] combine 2D detection and 3D segmentation for feature extraction, and then use a translation and rotation regression network to obtain the 6D pose of the object. Ref. [26] introduces a transformer structure where prior information is used as the query, and instance features as the key and value to obtain object embeddings. These embeddings are then constrained based on the relationships between neighboring points in the point cloud, ultimately regressing the object pose estimation. Ref. [7] proposes the NOCS and provided the correspondence between NOCS and the real 6D pose. A CNN network is used to simultaneously predict the target object's mask and NOCS map, with the pose calculated through the established correspondences. Similarly, [27] also proposes a new unified shape space for addressing the intra-class variation. Other studies like [8,28] incorporate priors as input and learn to deform the priors to obtain the object's coordinates in the normalized space. Refs. [24,29] propose various data augmentation techniques to train networks, addressing overfitting caused by limited dataset sizes, enhancing the model's generalization ability, and improving its robustness. CR-Net [30] proposes a cascaded network for estimating the 6D pose of objects. The first stage of the network is designed to extract instance features, while the second stage incorporates prior information. SAR-Net [31] focuses on solving the pose estimation of symmetric objects by aligning prior information with the observed instance to predict rotation. Subsequently, it employs symmetry modeling based on the observed parts of the instance and compares this with the observed instance information to predict translation and size.

3. Materials and Methods

3.1. Category Prior

A category prior serves as the geometric representation of objects within a specific category. Randomly selecting the point cloud of a particular object to represent the entire category can negatively impact the results due to the issue of intra-class variation. For example, if the selected object significantly deviates from the average shape of the category, the training process of the deformation network may become unstable and difficult to

converge. Computing an average point cloud for a given category is also challenging, as manually establishing pointwise correspondence between different point clouds is nearly impossible. However, it is important to note that objects within the same category typically share similar geometric features. For instance, a laptop generally consists of two flat rectangular prisms connected by a hinge, while a bottle is usually composed of two stacked cylinders of different sizes. Therefore, we still aim to use an average shape to represent a category. Following the approach of SPD [8], we utilize a neural network as an encoder to encode the point cloud of an object into an embedding and employ a simple neural network as a decoder to decode this embedding. The neural network is trained such that the decoded point cloud matches the input point cloud.

Figure 1 illustrates the structure of the encoder and decoder used for the generation of category prior.

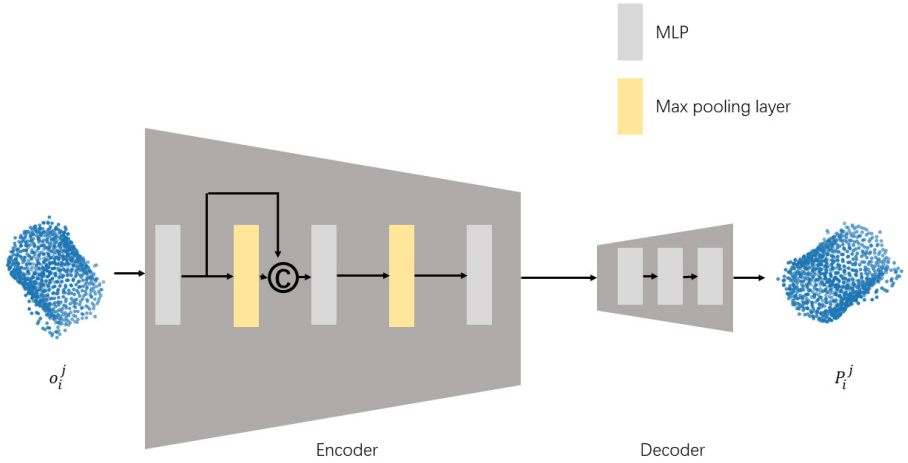

Figure 1. Illustration of structure of the encoder and decoder used for the generation of category prior.

We input the point cloud $o_i^j \in \{o_1^j, \ldots, o_n^j\}$ of a given object into the encoder to obtain the embedding EM_i^j of the point cloud:

$$EM_i^j = E(o_i^j), \tag{1}$$

where E is the encoder, o_i^j is the point cloud of a given object, and EM_i^j is the embedding of the point cloud.

We input EM_i^j into the decoder to obtain a new point cloud P_i^j:

$$P_i^j = D(EM_i^j), \tag{2}$$

where p_i^j is the reconstructed point cloud, and D is the decoder.

We aim for the input point cloud o_i^j and the reconstructed point cloud p_i^j to be as consistent as possible. To measure the distance between o_i^j and p_i^j, we use the Chamfer Distance loss. Therefore, the loss function is defined as follows:

$$CD(p_i^j, o_i^j) = \sum_{x \in p_i^j} \min_{y \in o_i^j} \|x - y\|_2^2 + \sum_{y \in o_i^j} \min_{x \in p_i^j} \|y - x\|_2^2. \tag{3}$$

We input the point cloud $o_i^j \in \{o_1^j, \ldots, o_n^j\}$ into the trained encoder to obtain EM_i^j. This process is repeated for all point clouds belonging to the same category, yielding $EM_i^j, \ldots, EM_n^j$,

where n denotes the number of the objects in the category O^j. Subsequently, these n embeddings are summed and averaged to obtain the category's average embedding EM^j:

$$EM^j = \frac{1}{n}\sum_i EM_i^j, \tag{4}$$

where EM^j represents the average embedding of category O_j.

By using EM^j as the input to the trained decoder, we obtain the output p^j, which can be regarded as the representation of the average shape for that category, i.e., the prior shape of the category. Figure 2 illustrates the derived category priors for bow, cup, bottle, laptop, camera, and can.

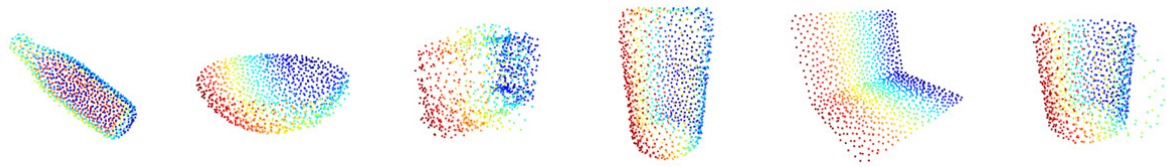

Figure 2. Illustration of the priors of six selected categories, which are bottle, bow, camera, can, laptop, and mug, arranged from left to right.

3.2. Network Structure

We denote the observed object's RGB image, depth image, and point cloud as I_o, D_o, and P_o, respectively, where $P_o \in \mathbb{R}^{N_o \times 3}$ and $I_o \in \mathbb{R}^{H \times W \times 3}$. $P_p \in \mathbb{R}^{N_p \times 3}$ denotes the category prior point cloud, and N_o represents the number of pixels in the foreground with valid depth values. N_p denotes the number of points in the category prior point cloud P_p.

Given a scene image captured by the camera, we first obtain the target object's mask using Mask R-CNN [32], which corresponds to the object in the image I_o. We then extract the corresponding region in the scene's depth image with the same coordinates in the image coordinate system, resulting in the target object's depth image. By applying the camera's intrinsic parameters, the depth image of the target object can be converted into the point cloud P_o.

After obtaining the observed object's I_o and P_o, we use PSPNet [33] to extract features from the RGB image, resulting in $F_{\text{rgb}} \in \mathbb{R}^{N_o \times C}$. We also use PointNet++ [34] to extract features from P_o, resulting in $F_o^p \in \mathbb{R}^{N_o \times C}$, and similarly, we use PointNet++ to extract features from P_p, resulting in $F_p^p \in \mathbb{R}^{N_p \times C}$. The features F_{rgb}, F_o^p, and F_p^p will serve as inputs to the entire model.

In this section, to facilitate the understanding of the following content, we will first provide an overview of our model structure. Then, we will detail each component of the model, which includes the Prior Guided Observation Network, Deformation Refine Module, Low-rank Transformer, and Feature Fusion Attention Network.

3.2.1. Overview

Figure 3 illustrates the network structure of the refined prior-guided category-level 6D pose estimation proposed in this paper. The model takes the observed object's RGB image I_i, depth image D_i, and category prior P_p as inputs. Through the Prior Guided Observation Network, the model obtains the instance local feature embedding, instance global feature embedding, category local feature embedding, and category global feature embedding. These feature embeddings are then utilized in the Deformation Refine Module to obtain the deformation matrix D and in the Correspondence Refine Module to obtain the dense correspondence matrix M.

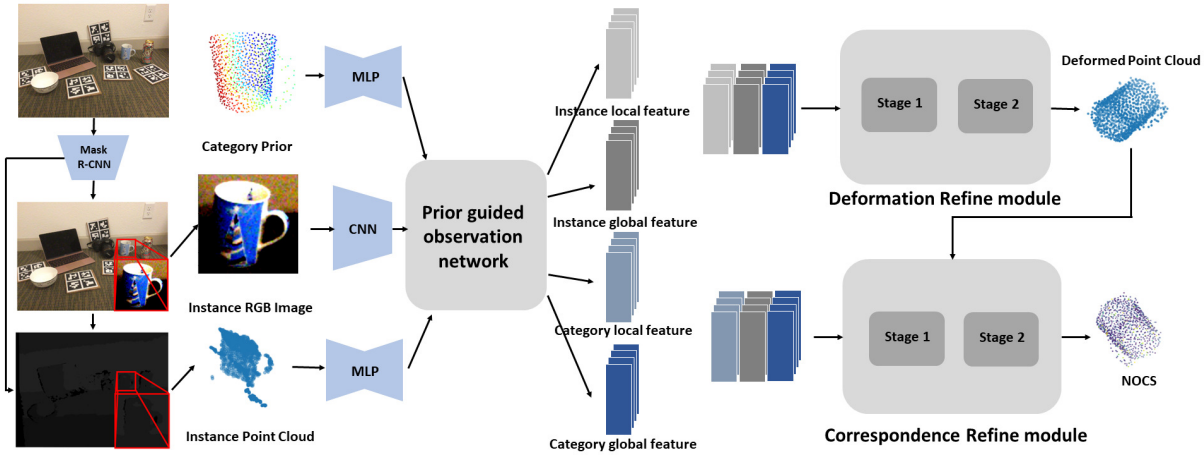

Figure 3. The figure illustrates the overall structure of the model proposed in this paper. The RGB image of the scene is processed through Mask R-CNN to crop the target object's RGB image. Using this patch, the corresponding depth image of the target object can also be obtained from the scene's depth image, leading to the extraction of the target object's point cloud. The target object's RGB image, point cloud, and category prior are then input into the Prior Guided Observation Network. The outputs are fed into the Deformation Refine Module and the Correspondence Refine Module, respectively, yielding the deformed point cloud and dense correspondence matrix. Through these computations, the NOCS coordinates of the target object are ultimately obtained.

3.2.2. Prior Guided Observation Network

The structure of the Deformation Refine Module is shown in Figure 4. We propose this module inspired by SGPA [28]. Our goal is to introduce the semantic features of the category prior through the Prior Guided Observation Network, providing additional information for the subsequent deformation process. Unlike SGPA, we inject the category prior into the instance point cloud, guiding the transformation of semantic features by generating a heatmap from the transformed point cloud and the original point cloud. The heatmap can be considered a guide for how the instance features change after incorporating the prior information. This same guidance can also be applied to direct image transformations.

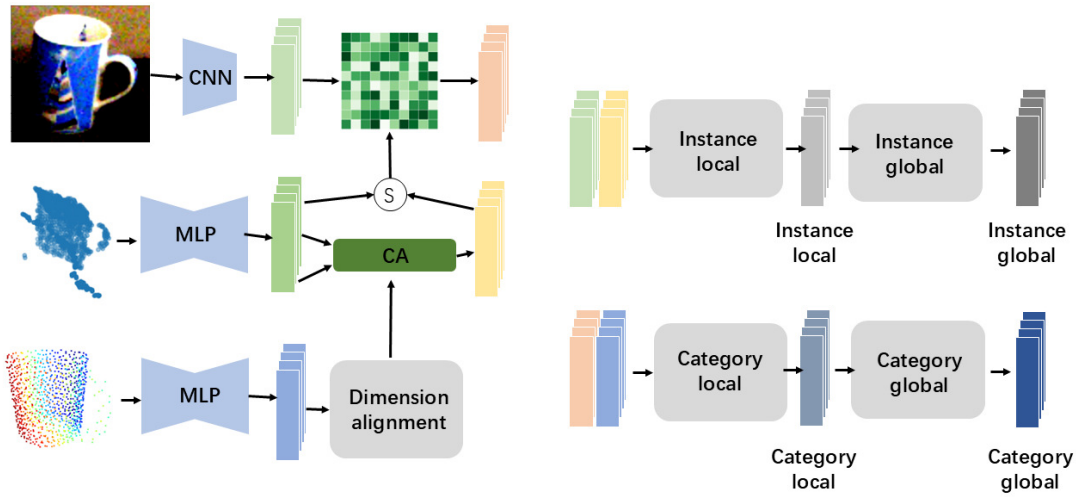

Figure 4. Illustration of the framework of the Prior Guided Observation Network.

First, we pass F_p^p through the Dimension Alignment Module, which consists of a two-sided MLP (Multi-layer Perceptron), aimed at reducing the dimensionality of F_p^p to facilitate its subsequent matching with F_o^p.

Next, we input F_p^p and F_o^p into the CrossAttention (CA) module, where F_p^p serves as the query, and F_o^p as the key and value. The computation in this module is performed as follows:

$$\mathbf{Q} = \mathbf{F}_p^p \mathbf{W}^q, \quad \mathbf{K} = \mathbf{F}_o^p \mathbf{W}^k, \quad \mathbf{V} = \mathbf{F}_o^p \mathbf{W}^v, \tag{5}$$

$$F = \text{softmax}((\mathbf{Q}\mathbf{K}^\top)\mathbf{V}). \tag{6}$$

In this context, **Q**, **K**, and **V** represent the query, key, and value, respectively. W^q, W^k, and W^v are learnable projection matrices for the query, key, and value, respectively. We utilize a total of M attention blocks, implementing a multi-head attention mechanism, to thoroughly model the information in F_p^p and fully inject it into F_o^p, facilitating the extraction of more informative semantic features from the category prior. The output features from these M attention blocks are concatenated as follows:

$$F_d = \text{Concat}(F^{(1)}, F^{(2)}, \ldots, F^{(M)}), \tag{7}$$

Next, we input F_d into a Feed Forward Network (FFN) to obtain the features F_o^d of the instance point cloud, transformed by the prior point cloud.

To measure the degree of change in F_o^d relative to F_o^p, we compute a heatmap (**H**) using the cosine similarity between F_o^d and F_o^p:

$$\mathbf{H} = cosine \quad similarity = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}||\mathbf{B}|}, \tag{8}$$

$$|\mathbf{A}| = \sqrt{\sum_{i=1}^{n} A_i^2}, \quad |\mathbf{B}| = \sqrt{\sum_{i=1}^{n} B_i^2}, \tag{9}$$

where **A** and **B** are vectors that represent F_o^d and F_o^p, respectively. Subsequently, the corresponding category prior semantic feature F_p^s can be generated by

$$(F_p^s) = \text{softmax}(\mathbf{H}) \times F_{rgb}, \tag{10}$$

Next, we concatenate F_o^s and F_o^p along the channel dimension, and then pass them through the instance local network, which consists of three layers of MLP, to obtain the local features of the observed object, denoted as F_o^l. Subsequently, an instance global network, composed of two layers of MLP and an average pooling layer, is used to generate the global features of the observed object, denoted as F_o^g.

Similarly, for F_p^s and F_p^p, we concatenate them along the channel dimension and pass them through the category local network, also composed of three layers of MLP, to obtain the local features of the observed object, denoted as F_p^l. Following this, the category global network, consisting of two layers of MLP and an average pooling layer, is used to generate the global features of the observed object, denoted as F_p^g.

3.2.3. Deformation Refine Module

The structure of the Deformation Refine Module is shown in Figure 5. The purpose of the Deformation Refine Module is to regress a deformation matrix $D \in \mathbb{R}^{N_p \times 3}$, which is used to deform P_r into the predicted shape of the target object, $\hat{P}_o$. The predicted shape $\hat{P}_o$ is obtained by directly adding the prior model and the deformation field as follows:

$$\hat{P}_o = D + P_r. \tag{11}$$

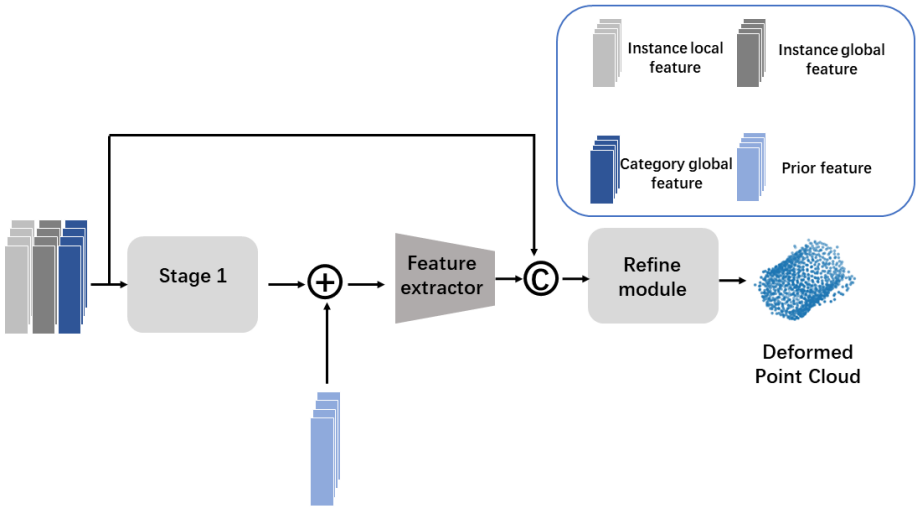

Figure 5. Illustration of the Deformation Refine Module.

In previous methods, regressing D from high-dimensional complex feature information using a single network was insufficient, which is why we propose this Deformation Refine Module.

We first concatenate F_o^l, F_o^g, and F_p^g along the channel dimension. The first stage of the network, which is the same as the network used in previous methods, consists of three MLP layers and produces the first-stage deformation matrix D_{mid}. By incorporating the information from P_r, we obtain a coarsely deformed point cloud shape P_{mid}. However, this deformation is not sufficient, so P_{mid} is passed into the second stage of the network. The second stage extracts features from P_{mid} to obtain F_{mid}. Considering the possibility of information loss after passing through the first stage, we reintroduce F_o^g and F_p^g, obtained from the Prior Guided Observation Network, into the third stage of the network. The third stage, also consisting of three MLP layers, outputs the final deformation matrix D.

Theoretically, the accuracy of the result would improve with the repeated stacking of networks in the second and third stages of the refine module. However, experiments show that the results saturate after a single repetition. Therefore, we employ a three-stage network in this module.

3.2.4. Correspondence Refine Module

The structure of the Correspondence Refine Module is shown in Figure 6. The purpose of the Correspondence Refine Module is to regress a dense correspondence matrix $M \in \mathbb{R}^{N_o \times N_r}$ that maps the predicted point cloud $\hat{P}_o$ of the target object to the NOCS. Thus, the final NOCS model is obtained by left-multiplying M with $\hat{P}_o$ as follows:

$$P_{\text{nocs}} = M \times \hat{P}_o. \tag{12}$$

Similarly, considering that regressing a dense matrix directly from high-dimensional complex information is challenging for a single-stage network, a refine module is introduced here as well.

We first concatenate F_p^l, F_o^g, and F_p^g along the channel dimension. The first stage of the network produces M_{mid}, and this network, like previous methods, consists of three MLP layers. Given the potential for information loss, it is necessary to reintroduce the previously obtained information. At this stage, we also acquire information about the transformed point cloud $\hat{P}_o$, so we introduce F_{mid} as well. Explicitly considering the weights of F_{mid} and F_p^l is not appropriate, as M is a dense matrix. Therefore, we use the Feature Fusion Attention Network to fuse the feature information, obtaining F_f^l. We then concatenate F_f^l,

M_{mid}, F_o^g, and F_p^g along the feature dimension and pass them into the second stage of the network, which consists of three MLP layers, to obtain the final matrix M.

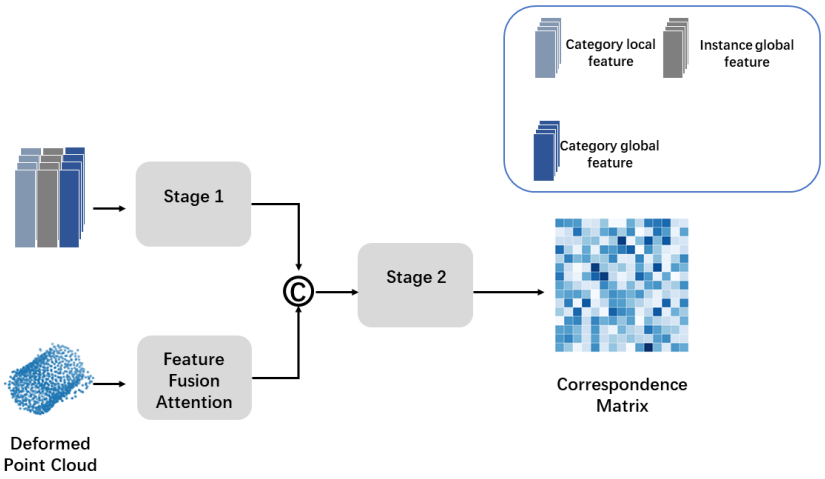

Figure 6. Illustration of the Correspondence Refine Module.

3.2.5. Feature Fusion Attention Network

The structure of the Feature Fusion Network is illustrated in Figure 7. Inspired by the Self-Attention Layer used in Transformers [35], the relationships between the input matrices are first identified and then concatenated with the original information. The final fused information is obtained through a Feed Forward Network (FFN).

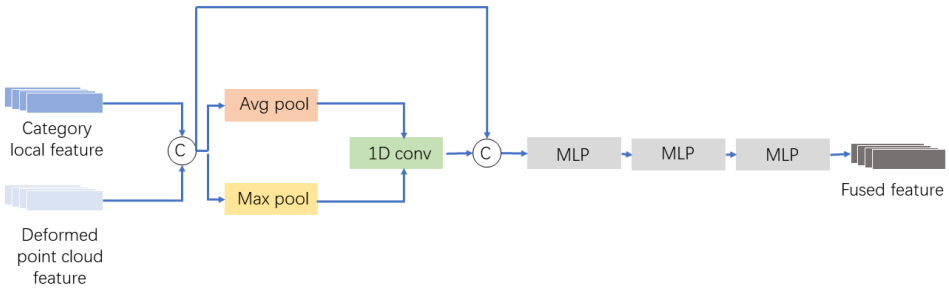

Figure 7. Illustration of the structure of the Feature Fusion Attention Network.

F_p^l and F_{mid} are concatenated along the channel dimension, followed by processing through an average pooling layer and a maximum pooling layer, with the results assigned certain weights:

$$\text{emb} = f_{\text{conv}}(\lambda_1 \text{out}_{\text{avg}} + \lambda_2 \text{out}_{\text{max}}), \tag{13}$$

where emb is the output of the 1D convolutional layer.

Subsequently, the result is passed through a one-dimensional convolutional layer and then concatenated with the initially concatenated embedding. Finally, the output is obtained through a three-layer MLP.

3.3. Loss Function

3.3.1. Reconstruction Loss

The reconstruction loss is employed to indirectly supervise the deformation matrix. During training, the ground truth of the observed object's point cloud is known, and we aim to minimize the discrepancy between the deformed point cloud and the ground truth point cloud of the observed object P_{gt}. We use the Chamfer Distance to measure the distance between the two point clouds. Additionally, in the refined prior-guided category-level 6D

object pose estimation model, the predicted point cloud of the observed object is given by $P = D + P_p$. Therefore, the reconstruction loss used to supervise D is defined as follows:

$$L_r = \sum_{i \in P^i} \min_{j \in P^j_{gt}} \|i - j\|_2^2 + \sum_{j \in P^j_{gt}} \min_{i \in P^i i} \|i - j\|_2^2. \tag{14}$$

3.3.2. Correspondence Loss

The correspondence loss is employed to indirectly supervise the dense correspondence matrix. During training, the ground truth coordinates of the observed object's point cloud in the NOCS are known, and we aim to minimize the discrepancy between the NOCS coordinates of the deformed point cloud and the ground truth point cloud of the observed object. We measure the distance between the two point clouds using the distance between corresponding point coordinates. Additionally, in the refined prior-guided category-level 6D object pose estimation model, the predicted NOCS coordinates of the observed object's point cloud are given by $P_{\text{nocs}} = M \times P$. Therefore, we use the soft L1 loss to supervise M, and the correspondence loss is defined as follows:

$$L_c(x, x_{gt}) = \frac{1}{N_c} \begin{cases} 5(x - x_{gt})^2 & \text{if } |x - x_{gt}| \leq 0.1, \\ |x - x_{gt}| - 0.05 & \text{otherwise.} \end{cases} \tag{15}$$

where x represents the predicted coordinates, and x_{gt} represents the ground truth.

3.3.3. Regulation Loss

The regulation loss is used to directly supervise D and M. Since each row vector in M represents the correspondence between each point in the predicted point cloud P of the observed object and the points in P_{nocs}, we use the mean cross-entropy loss to regulate M. The mean cross-entropy loss is defined as follows:

$$L_{\text{entropy}} = \frac{1}{N_o} \sum_i \sum_j -M_{i,j} \log M_{i,j}. \tag{16}$$

Considering that a specific object within a category should not undergo significant deformation relative to the category prior, we also need to regularize D. The regularization loss is defined as follows:

$$L_{\text{def}} = \frac{1}{N_r} \sum_{d_i \in D} \|d_i\|^2. \tag{17}$$

3.3.4. Overall Loss

Thus, the loss for the first stage of the network can be defined as follows:

$$L_{d1} = \lambda_1 L_{r1} + \lambda_2 L_{c1} + \lambda_3 L_{\text{def1}} + \lambda_4 L_{\text{reg1}}, \tag{18}$$

where L_{r1}, L_{c1}, L_{def1}, and L_{reg1} represent the reconstruction loss, correspondence loss, and two regularization losses in the first stage, respectively.

Similarly, the loss for the second stage of the network is defined as

$$L_{d2} = \lambda_1 L_{r2} + \lambda_2 L_{c2} + \lambda_3 L_{\text{def2}} + \lambda_4 L_{\text{reg2}}, \tag{19}$$

where L_{r2}, L_{c2}, L_{def2}, and L_{reg2} represent the reconstruction loss, correspondence loss, and two regularization losses in the second stage, respectively.

Then, the overall loss can be defined as

$$L_d = k_1 \times L_{d1} + k_2 \times L_{d2}. \tag{20}$$

3.4. 6D Pose Parameter Calculation

Through the entire model, we obtain the deformation matrix D for the category prior and the dense correspondence matrix M with the NOCS coordinates. Consequently, the coordinates of the observed object in the NOCS can be computed as $P_{\text{nocs}} = M \times (P_p + D)$.

Having obtained the coordinates in the NOCS, we enhance the robustness of the results by applying the RANSAC algorithm to eliminate noise. The Umeyama algorithms are then employed to recover the 6D pose of the observed object from the NOCS, which includes translation and rotation. The object's size is calculated using the sum of the variances of the points after mean subtraction in the NOCS coordinates, along with the singular values obtained from the singular value decomposition (SVD) of the covariance matrix:

$$\text{size} = \frac{1}{\text{varP}} \times \sum V, \tag{21}$$

where varP is the sum of the variances of the points after mean subtraction in the source point cloud, and V is the singular value vector obtained from the singular value decomposition of the covariance matrix.

4. Results

We trained our model on the CAMERA25 and REAL275 datasets. Subsequently, we performed grasping experiments using a UR3 robotic arm in the PyBullet simulation environment to validate the effectiveness of the model.

4.1. Preprocessing and Implementation Detail

We utilized Mask R-CNN, proposed by [32], to perform semantic segmentation on the scene images. After segmentation, we randomly selected 1024 points from the segmented instance image as input to the model's RGB image. PSPNet, proposed by [33], was employed to extract features from the RGB image. To ensure consistency with previous work and enable a fair comparison, the backbone of PSPNet was set to ResNet-18, and in the pyramid pooling module, the feature maps were pooled at scales of 1, 2, 3, and 6. Both the instance point cloud and the category prior point cloud contained 1024 points. We used PointNet++, proposed by [34], to extract features from both the instance point cloud and the category prior point cloud.

The PointNet++ network we employed had four layers, with the radii of the local regions used for feature extraction in these four layers set to $[0.01, 0.02]$, $[0.02, 0.04]$, $[0.04, 0.08]$, and $[0.08, 0.16]$, respectively. Each layer's two parameters represented two different radii, which divided the local regions of the point cloud into different scales, thereby enabling the model to capture multi-scale features. The numbers of points sampled at each layer were 512, 256, 128, and 64. The numbers of neighboring points sampled within the local regions at each layer were $[16, 32]$, $[16, 32]$, $[16, 32]$, and $[16, 32]$ respectively. The number of layers in the Multi-layer Perceptron (MLP) used at each scale in each layer and the number of neurons per layer were set to [[16, 16, 32], [32, 32, 64]], [[64, 64, 128], [64, 96, 128]], [[128, 196, 256], [128, 196, 256]], and [[256, 256, 512], [256, 384, 512]].

The training and evaluation were both performed on an 11th Gen Intel Core i9-11900K @ 3.50 GHz CPU with a single Nvidia GeForce RTX 3080 graphics card.

4.2. Dataset

The CAMERA25 and REAL275 datasets were introduced by Wang et al. in [7]. The objects within these datasets belong to six common categories: bottle, bowl, camera, can, laptop, and mug. Each category contains multiple object instances with varying shapes, colors, and sizes. The CAMERA25 dataset comprises a total of 309 object instances and 300,000 RGB-D images. The REAL275 dataset includes 8K RGB-D frames, of which 4300 are used for training, 950 for validation, and 2750 for testing. It also features 18 real-world scenes and 42 unique object instances, with 7 scenes allocated for training, 5 for validation, and 6 for testing.

4.3. Evaluation Metrics

We utilized 3D Intersection over Union (IoU), translation, and rotation errors as metrics to evaluate the accuracy of the model from different perspectives.

Three-dimensional IoU: Three-dimensional IoU measures the discrepancy between the predicted size and the ground truth, represented as the ratio of the intersection to the union of the predicted object's three-dimensional bounding box and the ground truth. The formula is defined as follows:

$$\text{IOU} = \frac{B_p \cap B_g}{B_p \cup B_g} \times 100\%, \tag{22}$$

where B_p and B_g represents the predicted 3D bounding box and the ground truth of the 3D bounding box.

We employed two types of 3D IoU metrics: 3D25 and 3D50. These indicate that the 3D IoU is greater than 25% and 50%, respectively, as the evaluation criteria.

Translation and Rotation Error: The translation and rotation errors measure the discrepancy between the predicted and ground truth values in terms of translation and rotation. These errors represent the difference between the predicted translation and rotation of the target object and the ground truth. We used four specific translation and rotation error metrics: 5°2 cm, 5°5 cm, 10°2 cm, and 10°5 cm, where each denotes a rotation error of 5° with a translation error of 2 cm, a rotation error of 5° with a translation error of 5 cm, a rotation error of 10° with a translation error of 2 cm, and a rotation error of 10° with a translation error of 5 cm, respectively.

4.4. Comparison with State-of-the-Art Methods

Tables 1 and 2 present the comparison of the mean Average Precision (mAP) of the method proposed in this paper with the previous state-of-the-art methods [7,8,28–31]. On the CAMERA25 dataset, our experimental results for 3D50, 3D75, 5°2 cm, 5°5 cm, 10°2 cm, and 10°5 cm are 93.8%, 89.2%, 70.2%, 72.6%, 86.7%, and 90.4%, respectively. Notably, our method achieves the highest mAP in the 3D75, 10°2 cm, and 10°5 cm metrics, exceeding the previous best mAP by 0.2%, 4.0%, and 0.9%, respectively. On the REAL275 dataset, our experimental results for 3D50, 3D75, 5°2 cm, 5°5 cm, 10°2 cm, and 10°5 cm are 82.1%, 66.1%, 38.7%, 40.3%, 65.1%, and 79.7%, respectively. Our method achieves the highest mAP in the 3D50, 10°2 cm, and 10°5 cm metrics on this dataset as well, surpassing the previous best mAP by 2.0%, 1.6%, and 0.5%, respectively.

Table 1. Comparison with state-of-the-art methods on CAMERA25 dataset. The bolded parameters indicate the optimal values for each metric.

Method	3D50	3D75	5°2 cm	5°5 cm	10°2 cm	10°5 cm
NOCS [7]	83.9	69.5	32.3	40.9	48.2	64.6
SAR-Net [31]	86.8	79.0	66.7	70.9	75.3	80.3
SPD [8]	93.2	83.1	54.3	59.0	73.3	81.5
CR-Net [30]	**93.8**	88.0	72.0	76.4	81.0	87.7
RPB-Pose [29]	93.1	89.0	**73.5**	**79.6**	82.1	89.5
SGPA [28]	93.2	88.1	70.7	74.5	82.7	88.4
Ours	93.4	**89.2**	70.2	72.6	**86.7**	**90.4**

Figure 8 illustrates a comparison between our method and the results obtained by SGPA in two scenarios. The red boxes represent the model's predicted results, while the green boxes represent the ground truth bounding boxes. The top two scenarios show the predictions made by SGPA, while the bottom two scenarios display the predictions made by our method.

Table 2. Comparison with state-of-the-art methods on REAL275 dataset. The bolded parameters indicate the optimal values for each metric.

Method	3D50	3D75	5°2 cm	5°5 cm	10°2 cm	10°5 cm
NOCS [7]	78.0	30.1	7.2	10.0	13.8	25.2
SAR-Net [31]	79.3	62.4	31.6	42.3	50.3	68.3
SPD [8]	77.3	53.2	19.3	21.4	43.2	54.1
CR-Net [30]	79.3	55.9	27.8	34.3	47.2	60.8
RPB-Pose [29]	-	**67.8**	**38.2**	**48.2**	63.1	79.2
SGPA [28]	80.1	61.9	35.9	39.6	61.3	70.7
Ours	**82.1**	66.1	36.7	40.3	**64.7**	**79.7**

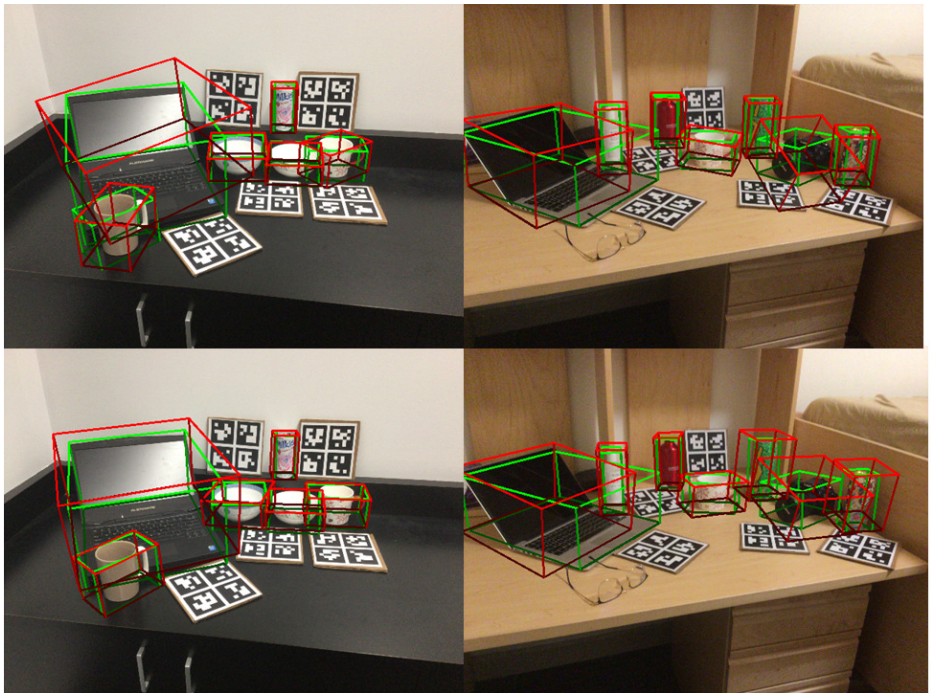

Figure 8. Illustration of the comparison between the detection results of our proposed method and those of SGPA. The top two images depict the results from SGPA, while the bottom two images show the predictions made by our method. The red boxes represent the algorithm's predicted results, and the green boxes indicate the ground truth. It is evident that our proposed algorithm performs better than SGPA in predicting rotation and size.

The experimental results on both datasets indicate that our method achieves superior results in the 10°2 cm and 10°5 cm metrics. Additionally, the visual validation results further demonstrate that our method outperforms SGPA in predicting rotation and scale, thereby proving the effectiveness of our approach.

4.5. Ablation Studies

Table 3 presents the results on the CAMERA25 dataset after incorporating the Feature Fusion Module and Refine Module into the network proposed in this paper. The results indicate that while the Feature Fusion Module provides a certain degree of improvement to the model, it notably enhances performance on the 5°5 cm metric, increasing the mAP by 2.3%. Furthermore, after adding the Refine Module, significant improvements are observed across all evaluation metrics, with increases of 0.7%, 2.4%, 4.3%, 4.1%, 2.3%, and 2.6% in the 3D50, 3D75, 5°2 cm, 5°5 cm, 10°2 cm, and 10°5 cm metrics, respectively. This demonstrates that each module contributes to the overall effectiveness of the model.

Table 3. Ablation study of the feature fusion module and refine module on CAMERA25 dataset. ✓indicates inclusion, and - indicates exclusion.

	Feature Fusion Module	Refine Module	CAMERA25					
			3D50	3D75	5°2 cm	5°5 cm	10°2 cm	10°5 cm
1	-	-	92.1	86.5	66.0	66.2	84.9	87.7
2	✓	-	92.7	86.8	65.9	68.5	84.4	87.8
3	✓	✓	93.4	89.2	70.2	72.6	86.7	90.4

4.6. Simulation of Robotic Grasping

To further validate the effectiveness of the model, we applied the model proposed in this paper to robotic arm-grasping experiments, conducting simulations through PyBullet. The experiments utilized a UR3 robotic arm equipped with a Robotiq 85 gripper. The models used in the simulation were bottle, bowl, camera, can, laptop, and mug, with three instances of each model. Some of these models are shown in Figure 9. Two cameras were used, a Realsense 415 positioned above the robotic arm, and a Realsense 435i positioned directly in front of the robotic arm, both fixed and not moving with the arm. The simulation involved 50 scenes, each loaded with three objects, with random object positions and rotations. Each object category was subjected to 25 grasping attempts. Figure 9 shows some models we used during the simulation. Figure 10 shows some of the scenes used in the simulation.

The 6D pose estimation of symmetrical objects is a significant challenge in 6D pose estimation. This is because a symmetrical object may have multiple correct rotation predictions, while there is only one ground truth. If the algorithm does not effectively address the mismatch between the correct predictions and the ground truth, it can lead to difficulties in training convergence and result in inaccurate final rotation predictions. Figure 11 illustrates the grasping process of a symmetrical object (bottle) in two different scenarios with varying positions and rotations, demonstrating that our algorithm effectively handles the challenges associated with predicting the rotation of symmetrical objects.

The Low-rank Transformer used in SGPA increases inference time. Hence, we chose another relatively lightweight model, SPD, for comparison with our method. Table 4 presents the detection success rates and grasping success rates for each object category, comparing these results with those obtained when implementing the SPD algorithm.

To ensure a fair comparison, we implemented the SPD algorithm using the same parameters and structures for the shared components. Specifically, we utilized Mask R-CNN for segmenting the target objects. For semantic feature extraction from RGB images, we employed a PSPNet with a ResNet-18 backbone. The input priors were also generated using the information obtained from our trained encoder, with the prior point cloud consisting of 1024 points.

Figure 9. Illustration of part of the models we used for simulation.

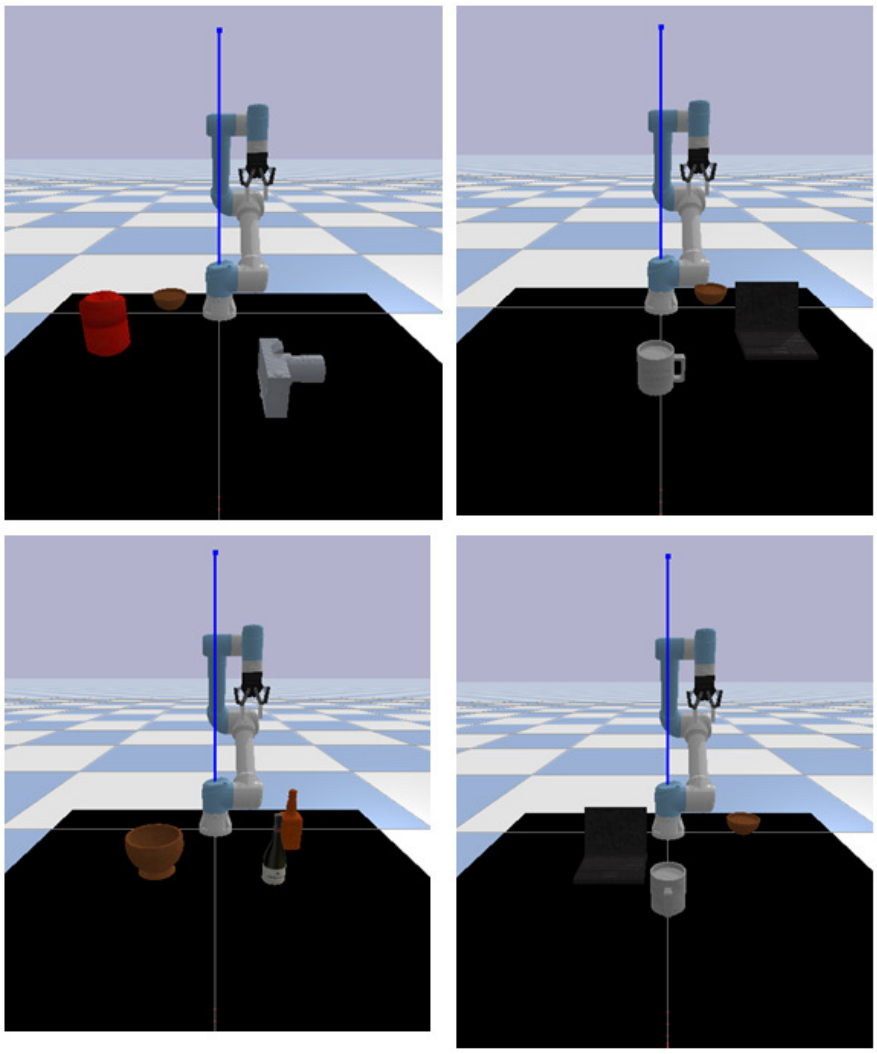

Figure 10. Illustration of part of scenes we used for simulation.

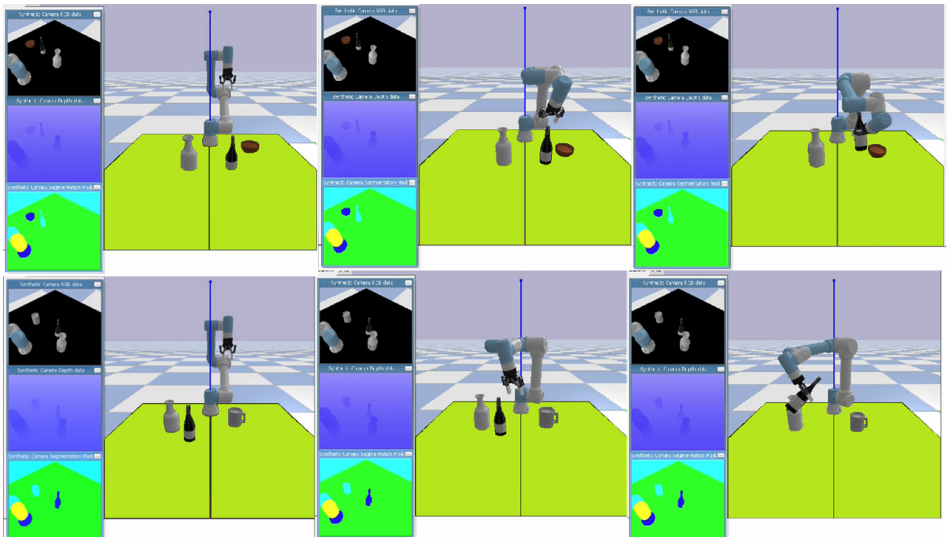

Figure 11. Illustration of the process of robotic grasping. Three subfigures represent the grasping process of the bottle in two different scenarios. From left to right, they show detection, grasping, and removal.

Table 4. Comparison of grasping and detection success rates with SPD.

		Bowl	Bottle	Can	Camera	Laptop	Mug
Detection Success Rate	SPD [8]	23/25	24/25	24/25	15/25	20/25	21/25
	ours	23/25	25/25	24/25	20/25	23/25	22/25
Grasp Success Rate	SPD [8]	22/25	24/25	23/25	13/25	20/25	20/25
	ours	20/25	24/25	23/25	19/25	22/25	22/25

The experimental results indicate that our proposed model can effectively identify target objects, achieving a detection success rate of 91.3%, which is an improvement of 6.6% over SPD. As seen in Table 3, both models have relatively low detection success rates for the camera category, primarily because the camera models selected for the simulation differ significantly in shape from the category prior point cloud. This highlights that intra-class variation remains a bottleneck problem for category-level 6D pose estimation. However, our algorithm demonstrates a better capability to address the intra-class variation issue compared to SPD, thereby proving the rationale and effectiveness of our approach.

5. Discussion

This paper presents a refined prior-guided category-level 6D object pose estimation method and applies it to robotic arm grasping. The model takes RGB images, depth images, and category priors as inputs, and outputs a deformation matrix and a dense correspondence matrix for the point cloud. Subsequently, the instance's NOCS coordinates are obtained, and the instance's 6D pose, including translation, rotation, and 3D size, is computed using the Umeyama algorithm and RANSAC. We address the intra-class variation problem by introducing an explicit deformation point cloud. Furthermore, this paper proposes a novel attention module and refine modules for point cloud deformation and dense correspondence to improve prediction accuracy. Extensive experiments conducted on the widely used CAMERA25 and REAL275 datasets demonstrate that our method outperforms existing approaches. Finally, we implement the model on a robotic arm, and grasping simulation experiments further validate the model's effectiveness and practicality.

A key advantage of this model is its ability to generalize to unseen objects, enabling pose estimation for all objects within a category rather than being limited to objects with known, precise 3D CAD models. This makes it broadly applicable to augmented reality, virtual reality, 3D reconstruction, and robotic arm grasping. The model's versatility is particularly important for robotic arm grasping, as the robot must handle a wide variety of objects in different environments. Our model assists the robotic arm in accurately identifying and understanding the precise location and orientation of objects, thereby enhancing its operational flexibility in complex environments. The simulation results indicate that our method can improve recognition accuracy, which in turn increases the success rate of robotic arm grasping.

Analyzing the cases of failed simulated grasps by the robotic arm, we found that many of them were caused by the target objects being heavily occluded. Looking ahead, while our method can already perform pose estimation among different instances within a category, its robustness under extreme conditions (such as partial occlusion, significant lighting changes, or reflective object surfaces) still needs improvement. Additionally, increasing the model's capacity can enhance its performance, but it may also reduce computational efficiency and real-time capability. Moreover, when the robotic arm needs to grasp fragile objects, the algorithm can only predict the 6D bounding box of the target object, without the ability to avoid specific parts of the object. To address the issue of partial occlusion, we may explore three directions: segmentation, feature extraction, and model capacity. Since our model currently uses Mask R-CNN for segmenting target objects and PSP-Net for feature extraction, future work could involve employing MAE (Masked Autoencoder) or SAM (Segment Anything Model) for more precise segmentation and feature extraction.

For the model itself, the incorporation of depth images and point cloud information can alleviate, to some extent, the severe impact on prediction results caused by occlusion and lighting condition variations when only RGB images are used. In future work, we will further refine the model structure to ensure more accurate deformation of the prior when adapting to partially missing instances. Furthermore, our future work will also focus on lightweight model design and the recognition of specific parts of the target object. The aim is to maintain the accuracy of the results while improving inference speed to ensure more effective grasping, thereby making the model more suitable for industrial applications.

6. Conclusions

In conclusion, this paper presents a refined prior-guided category-level 6D object pose estimation model and validates its effectiveness through simulation experiments in robotic arm grasping. The key contributions of this work include the introduction of a refine module and a novel attention mechanism aimed at addressing bottleneck issues in the field. Extensive experiments conducted on widely recognized benchmarks demonstrate that our method is superior to many existing approaches. Additionally, through robotic arm simulation experiments, we have shown that our method can significantly improve the detection success rate of target objects, thereby enhancing the grasping success rate of the robotic arm. Our approach effectively increases the flexibility of robotic arms in complex environments, as it enables the prediction of the pose of unknown objects without prior knowledge of their CAD models, which has important implications and applications in both industrial and everyday robotic tasks.

Author Contributions: Conceptualization, H.S. (Huimin Sun); methodology, H.S. (Huimin Sun); software, H.S. (Huimin Sun) and Y.Z.; validation, H.S. (Huimin Sun), Y.Z. and H.S. (Honglin Sun); formal analysis, Y.Z.; investigation, H.S. (Honglin Sun); resources, K.H.; data curation, H.S. (Huimin Sun) and Y.Z.; writing—original draft preparation, H.S.; writing—review and editing, Y.Z.; visualization, H.S. (Huimin Sun) and H.S. (Honglin Sun); supervision, K.H.; project administration, K.H.; funding acquisition, K.H. All authors have read and agreed to the published version of the manuscript.

Funding: This study was conducted with the support of Future Robotics Organization, Waseda University, and as a part of the humanoid project at the Humanoid Robotics Institute, Waseda University. This work was supported by JSPS KAKENHI Grant Numbers JP20H04267, JP21H05055. This work was also supported by a Waseda University Grant for Special Research Projects (Project number: 2024C-188). This work was also supported by JST SPRING, Grant Number JPMJSP2128.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kumra, S.; Joshi, S.; Sahin, F. Antipodal robotic grasping using generative residual convolutional neural network. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 9626–9633.
2. Morrison, D.; Corke, P.; Leitner, J. Learning robust, real-time, reactive robotic grasping. *Int. J. Robot. Res.* **2020**, *39*, 183–201. [CrossRef]
3. Sahin, C.; Garcia-Hernando, G.; Sock, J.; Kim, T.K. Instance-and category-level 6D object pose estimation. In *RGB-D Image Analysis and Processing*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 243–265.
4. Wang, C.; Xu, D.; Zhu, Y.; Martín-Martín, R.; Lu, C.; Fei-Fei, L.; Savarese, S. Densefusion: 6D object pose estimation by iterative dense fusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3343–3352.
5. Tremblay, J.; To, T.; Sundaralingam, B.; Xiang, Y.; Fox, D.; Birchfield, S. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv* **2018**, arXiv:1809.10790.

6. Fang, H.S.; Wang, C.; Gou, M.; Lu, C. Graspnet-1Billion: A large-scale benchmark for general object grasping. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Eecognition, Seattle, WA, USA, 13–19 June 2020; pp. 11444–11453.
7. Wang, H.; Sridhar, S.; Huang, J.; Valentin, J.; Song, S.; Guibas, L.J. 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, Long Beach, CA, USA, 15–20 June 2019; pp. 2642–2651.
8. Tian, M.; Ang, M.H.; Lee, G.H. Shape prior deformation for categorical 6d object pose and size estimation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020, Proceedings, Part XXI 16*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 530–546.
9. Park, K.; Mousavian, A.; Xiang, Y.; Fox, D. 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, Seattle, WA, USA, 13–19 June 2020; pp. 10710–10719.
10. Wang, G.; Manhardt, F.; Tombari, F.; Ji, X. GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 16611–16621.
11. Nie, T.; Ma, J.; Zhao, Y.; Fan, Z.; Wen, J.; Sun, M. Category-level 6D pose estimation using geometry-guided instance-aware prior and multi-stage reconstruction. *IEEE Robot. Autom. Lett.* **2023**, *8*, 2381–2388. [CrossRef]
12. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
13. Umeyama, S. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *13*, 376–380. [CrossRef]
14. Marullo, G.; Tanzi, L.; Piazzolla, P.; Vezzetti, E. 6D object position estimation from 2D images: A literature review. *Multimed. Tools Appl.* **2023**, *82*, 24605–24643. [CrossRef]
15. Muñoz, E.; Konishi, Y.; Beltran, C.; Murino, V.; Del Bue, A. Fast 6D pose from a single RGB image using Cascaded Forests Templates. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Republic of Korea, 9–14 October 2016; pp. 4062–4069.
16. Pavlakos, G.; Zhou, X.; Chan, A.; Derpanis, K.G.; Daniilidis, K. 6-DoF object pose from semantic keypoints. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 2011–2018.
17. Peng, S.; Liu, Y.; Huang, Q.; Zhou, X.; Bao, H. PVNet: Pixel-wise voting network for 6DoF pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4561–4570.
18. Zhao, W.; Zhang, S.; Guan, Z.; Zhao, W.; Peng, J.; Fan, J. Learning deep network for detecting 3D object keypoints and 6D poses. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 14134–14142.
19. Lepetit, V.; Moreno-Noguer, F.; Fua, P. EP*n*P: An accurate O(*n*) solution to the P*n*P problem. *Int. J. Comput. Vis.* **2009**, *81*, 155–166. [CrossRef]
20. Payet, N.; Todorovic, S. From contours to 3D object detection and pose estimation. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 983–990.
21. Sundermeyer, M.; Marton, Z.C.; Durner, M.; Triebel, R. Augmented autoencoders: Implicit 3D orientation learning for 6D object detection. *Int. J. Comput. Vis.* **2020**, *128*, 714–729. [CrossRef]
22. Liu, F.; Fang, P.; Yao, Z.; Fan, R.; Pan, Z.; Sheng, W.; Yang, H. Recovering 6D object pose from RGB indoor image based on two-stage detection network with multi-task loss. *Neurocomputing* **2019**, *337*, 15–23. [CrossRef]
23. He, Y.; Huang, H.; Fan, H.; Chen, Q.; Sun, J. FFB6D: A full flow bidirectional fusion network for 6d pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3003–3013.
24. Chen, W.; Jia, X.; Chang, H.J.; Duan, J.; Shen, L.; Leonardis, A. 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, Nashville, TN, USA, 20–25 June 2021; pp. 1581–1590.
25. Di, Y.; Zhang, R.; Lou, Z.; Manhardt, F.; Ji, X.; Navab, N.; Tombari, F. GPV-Pose: Category-level object pose estimation via geometry-guided point-wise voting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, New Orleans, LA, USA, 18–24 June 2022; pp. 6781–6791.
26. Lin, X.; Yang, W.; Gao, Y.; Zhang, T. Instance-adaptive and geometric-aware keypoint learning for category-level 6D object pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle WA, USA, 17–21 June 2024; pp. 21040–21049.
27. Chen, D.; Li, J.; Wang, Z.; Xu, K. 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, Seattle, WA, USA, 13–19 June 2020; pp. 11973–11982.
28. Chen, K.; Dou, Q. SGPA: Structure-guided prior adaptation for category-level 6d object pose estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2773–2782.
29. Zhang, R.; Di, Y.; Lou, Z.; Manhardt, F.; Tombari, F.; Ji, X. RBP-Pose: Residual bounding box projection for category-level pose estimation. In *Computer Vision—ECCV 2022*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 655–672.

30. Wang, J.; Chen, K.; Dou, Q. Category-level 6D object pose estimation via cascaded relation and recurrent reconstruction networks. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 4807–4814.
31. Lin, H.; Liu, Z.; Cheang, C.; Fu, Y.; Guo, G.; Xue, X. SAR-Net: Shape alignment and recovery network for category-level 6D object pose and size estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 6707–6717.
32. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
33. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid scene parsing network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2881–2890.
34. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
35. Vaswani, A. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.