Leveraging SE(3) Equivariance for Self-Supervised Category-Level Object Pose Estimation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Category-level object pose estimation aims to find 6D object pose of previously unseen object instances from known categories without access to object CAD models. In this work, we propose for the first time a self-supervised framework to learn category-level 6D object pose from a single 3D point cloud that can be either complete or partial. During training, our method assumes no ground-truth pose annotations, no CAD models, and no multi-view supervision. The key to our method is to disentangle shape and pose through an invariant shape reconstruction module and an equivariant pose estimation module, empowered by SE(3) equivariant point cloud networks. Our method can perform aligned reconstructions and therefore produce category-level reference frames without using any annotations. In addition, with equivariant pose embeddings, our approach achieves category-level pose estimation that is comparable to some fully supervised methods. Extensive experiments demonstrate the effectiveness of our approach on both complete and partial point clouds from the ModelNet40 benchmark and on real depth point clouds from the NOCS-REAL 275 dataset.

1 Introduction

Object pose estimation is a crucial computer vision task that is widely employed in robotics, human-object interaction, and augmented-reality applications. Previous work can be categorized broadly into two genres: classic instance-level 6D object pose estimation (*e.g.*, PoseCNN [23]), which assumes the availability of exact CAD models for all object instances; and category-level 6D object pose estimation (*e.g.*, [21]), which defines a category-level reference frame and is able to generalize to a category of object instances. The motivation underlying the category-level approach is to develop systems that can accommodate novel object instances that were unseen during training. This approach is therefore more applicable for general robotic vision systems (*e.g.*, of a home robot) that must face complex environments, such as those encountered in everyday life.

In order to achieve the benefits of generalizability, category-level 6D object pose estimation systems often require a significantly larger amount of annotated training data than classic instance-level systems. The reason for more training examples is the need to cover large intra-category shape variations among different object instances, in addition to the usual requirement for diversity in object poses, occlusion patterns, and environments. However, obtaining good 6D pose annotations is expensive and time-consuming. As a result, this line of research often relies heavily on synthetic training data. In order to scale up category-level object pose estimation systems to allow for more object categories, there is a strong demand for self-supervised methods that do not require pose annotations.

However, designing self-supervised methods for category-level pose estimation is very challenging. Although a few systems (*e.g.*, Self6D [20]) have recently been proposed for self-supervised instance-

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

level 6D object pose estimation, all of those systems have access to the CAD model of each instance, but without the pose annotations. These previous approaches are completely different from self-supervised category-level object pose estimation, for which no CAD model is available for any object instances during either training or testing. The lack of any CAD models and annotations present a formidable obstacle to the designer: *category-level pose is unspecified throughout the training*. To our knowledge, all of the existing systems for category-level pose estimation are fully-supervised, probably due to this issue.

In order to learn a category-level pose under self-supervision, we therefore require the network to develop the concept of a category-level reference frame during training. The NOCS [21] approach defines such a category-level reference frame (namely, Normalized Object Coordinate Space), in which all the shapes are aligned and dense canonical reconstruction of the visible points of the object of interest is performed. This procedure for dense canonical reconstruction is called NOCS regression. The relationship between the reference frame and canonical reconstruction inspires us to ask the following question: can a system learn to perform canonical reconstruction of object instances from the same object category, whose reconstruction space will naturally be a category-level reference frame? This canonical reconstruction presents two requirements: 1) all of the object reconstructions need to be aligned; and 2) the reconstruction should not change if the input object undergoes a change in pose. The second requirement suggests an SE(3)-invariant reconstruction of an observed object, which inspires us to consider point cloud processing networks that are SE(3)-invariant/equivariant.

Formally, given a set of point cloud transformations $T_A : \mathbb{R}^{N\times 3} \rightarrow \mathbb{R}^{N\times 3}$ for $A \in$ SE(3), a neural network $\phi : \mathbb{R}^{N\times 3} \rightarrow \mathcal{F}$ (where $\mathcal{F}$ is an arbitrary feature domain) is called equivariant if for each A there exists a equivariant transformation $S_A : \mathcal{F} \rightarrow \mathcal{F}$ so that:

$$S_A[\phi(X)] = \phi(T_A[X]) \text{ for } \forall\ A \in \text{SE(3)}, X \in \mathbb{R}^{N\times 3}$$

When S_A is an identity mapping, this network is also called invariant under the transformations.

Note that using an SE(3)-invariant network alone for shape reconstruction is not sufficient, because ground truth reconstructions are not available. The only thing available to the system is the input point cloud, which can be either a complete or a partial representation of an object, with a 6D object pose P that is to be estimated. If we can jointly estimate this object pose P, then we can transform our reconstructions by the 6D transformation in P so that a consistency loss can be formed between the proposed reconstruction and our input observation. In contrast to the pose invariance pursued by shape reconstruction systems, here this 6D pose estimation should naturally be equivariant with any SE(3) transformation applied to the input, which means that the estimated pose should undergo the same 6D transformation when the input object changes its 6D pose. The observations above lead us to disentangle shape and pose by leveraging SE(3) invariance and equivariance, respectively, and to construct a reconstruction-based self-supervision approach in the camera space. Our experiments show that this disentanglement can successfully produce the concept of category-level reference frame in an unsupervised manner for both complete and partial object observations, and with canonical reconstructions that are highly aligned (except for certain symmetry ambiguities). Our method achieves very accurate 6D pose estimation on synthetic point clouds that have been generated using models from ModelNet [22]. The system also works well on the real-world NOCS-REAL 275 dataset [21].

The contributions of this paper can be summarized as follows:
1. We propose a general self-supervised framework that achieves category-level 6D object pose estimation without the need for CAD models and annotations.
2. The concept of a category-level reference frame emerges naturally during self-supervised training, resulting in category-aligned reconstruction and enabling category-level 6D pose inference.
3. Our proposed method achieves high accuracy on both synthetic and real-world category-level pose estimation benchmarks.

2 Related Work

Supervised Category-Level Object Pose Estimation As a pioneering work, Wang et al. [21] proposed Normalized Object Coordinate Space (NOCS) as a category-specific canonical reference frame, so that the category-level pose of an unseen object can be defined as the transformation from its NOCS. Several followup works have improved NOCS by considering articulated objects [10], by

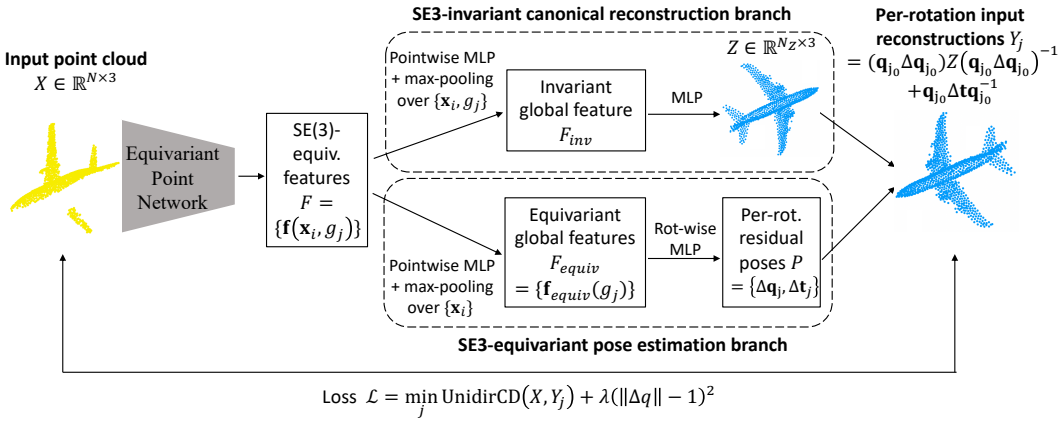


Figure 1: **Self-supervised category-level pose estimation pipeline.** Taking an input point cloud X (partial or complete), our network can construct invariant canonical shape Z and estimate an equivariant pose P, which together can form a self-supervised consistency loss to the input observation.

incorporating object pose tracking [19], by leveraging analysis-by-synthesis and shape generative models [6, 4], or by exploiting learnable deformation [18]. However, all of these approaches adopt fully-supervised training paradigms and assume that object poses are known at training time, which is too strong as a valid assumption in reality given the difficulty of collecting object pose annotations.

Self-supervised Pose Estimation To avoid the need for pose annotations in fully-supervised learning paradigms, several researchers have studied how to leverage self-supervision for instance-level object pose estimation. Self6D [20] and DTPE [14] obtain object poses by minimizing the difference between rendered depth (or RGBD) images and real observations in a self-supervised manner. AAE [15] trains a pose-augmented autoencoder on synthetic images, which can then be applied to pose estimation for real images. However, these methods still assume that a corresponding CAD model is available for the target object for which pose needs to be estimated. For category-level pose estimation without any CAD models, these approaches can not be applied. The key problem becomes to emerge a consistent category-level reference frame, which is known as *shape co-alignment* and studied in [11, 1, 3]. These methods usually assume clean meshes as inputs, or they leverage strong priors such as symmetry, making them unsuitable for more challenging shapes or even partial observations exhibiting high geometrical or topological variations. Moreover, these methods do not connect shape co-alignment with category-level 6D pose estimation. QEC [25] has a problem scope closest to ours. However, instead of aiming for self-supervised category-level 6D pose estimation for both complete and partial observations, this system only deals with 3D poses (3D rotation) of complete shapes. Also, QEC does not evaluate the predicted category-level 3D pose, and there is no guarantee regarding the pose quality. For all of these reasons, our work is the first to target self-supervised category-level object 6D pose estimation. This paper demonstrates significant potential of our method, using only a depth point cloud as the input.

Equivariant Point Cloud Neural Networks Recently, achieving SE(3) invariance and equivariance in point cloud processing has attracted a lot of attention. Early works [24, 12] focused more on achieving invariance, while most recent works are exploring equivariance, including Tensor Field Network [17] and SE3-transformer[9] leveraging spherical harmonics, quaternion equivariant capsule network[25] based on the idea of capsules, and, most recently, vector neuron[7]. We highlight that EPN[5] is only one network whose equivariance is not at the cost of network capacity.

3 Method

In section 3.1, we will first revisit Equivariant Point Networks [5] and focus on explaining its SE(3)-equivariance. In section 3.2, we will describe how we disentangle shape and pose, jointly estimate them, and form a self-supervised learning pipeline. In section 3.3, we will introduce how category-level pose is inferred and evaluated at test time.

3.1 Revisiting Equivariant Point Networks

To achieve shape and pose disentanglement, our framework leverages a state-of-the-art SE(3) equivariant network known as Equivariant Point Network (EPN) [5]. Taking a point cloud X as input, EPN achieves equivariance regarding both 3D translation $\mathbf{t} \in \mathbb{R}^3$ and 3D rotation $g \in$SO(3) and can generate per-point localized equivariant features $\mathbf{f}(\mathbf{x})$ for points $\mathbf{x} \in X$. We briefly review its translation and rotation equivariance here.

Translation Equivariance Considering 3D translation $T_{\mathbf{t}}[X] = X + \mathbf{t}$ that moves X to X', a translation equivariant transformation can be defined as $S_{\mathbf{t}} = I$ (identity mapping). This definition of translation equivariance is widely used in other SE(3)-equivariant networks, *e.g.*, Tensor Field Network [17] and SE(3)-transformer [9]. Under this equivariance, $\mathbf{f}'(\mathbf{x}') = \mathbf{f}'(\mathbf{x} + \mathbf{t}) = S_{\mathbf{t}}[\mathbf{f}(\mathbf{x})] = \mathbf{f}(\mathbf{x})$, which means the feature $\mathbf{f}$ of one point $\mathbf{x}$ does not change under the translation of the whole point cloud. The simplest way to produce this translation equivariance is to avoid directly processing the absolute point coordinates but instead use relative point offsets. EPN sets up local frames around each input point $\mathbf{x}$ and uses only the relative point offsets $\tilde{\mathbf{x}}_i = \mathbf{x}_i - \mathbf{x}$ for $\mathbf{x}_i \in \mathcal{N}(\mathbf{x})$ during feature extraction. These point offsets can still depict the geometry but do not change regarding the input translation.

Although any point network that uses only relative point offsets will be translation equivariant under this definition, EPN specifically chooses to use KPConv [16] as its convolution operator and performs convolution at each point $\mathbf{x}$ within its ball neighborhood $\mathcal{N}_{\mathbf{x}}$. More specifically, $(\mathbf{f} * h)(\mathbf{x}) = \sum_{\tilde{\mathbf{x}}_i + \mathbf{x} \in \mathcal{N}_{\mathbf{x}}} \mathbf{f}(\tilde{\mathbf{x}}_i) h(-\tilde{\mathbf{x}}_i)$, where $h(\tilde{\mathbf{x}})$ is a continuously parameterized function in $\mathcal{N}_{\mathbf{x}}$.

Rotation Equivariance Although a translation equivariant transformation $S_{\mathbf{t}}$ can be simply defined as the identity mapping, a non-trival (non-invariant) definition of rotation equivariance is very important, for the sake of rotation estimation. The key idea behind the rotation equivariance of EPN is to introduce a feature vector field F on SO(3) manifold. EPN first proposes a kernel-rotated version of KPConv, which rotates the convolution kernel h by a rotation $g \in$ SO(3); *e.g.*, its rotated kernel function is $h_g(\tilde{\mathbf{x}}_i) = h(T_{g^{-1}}[\tilde{\mathbf{x}}_i])$, where $(g)^{-1}$ is the inverse rotation of g. Using the rotated kernel h_g to perform convolution on the input point cloud will result in a new set of per-point features $\mathbf{f}(\mathbf{x}, g)$, equivalent to rotating the input X by g^{-1}. Assume for any $g \in$ SO(3), EPN can compute per-point features $\mathbf{f}(\mathbf{x}, g) \in \mathbb{R}^C$ using the corresponding rotated KPConv kernel h_g. Then the features for all g and all $\mathbf{x}$ form a per-point vector field $F = \phi(X) = \{\mathbf{f}(\mathbf{x}, g) \in \mathbb{R}^C \mid g \in \text{SO(3)}, \mathbf{x} \in X\}$ for points $\mathbf{x} \in X$ and all gs on SO(3) manifold. Note that the feature would be invariant if we simultaneously rotate X and the kernel by the same rotation $g_0 \in SO(3)$, *e.g.*, $\mathbf{f}(\mathbf{x}, g) = \mathbf{f}(T_{g_0}[\mathbf{x}], g_0 \circ g)$, where $T_{g_0}[\mathbf{x}] = \mathbf{q}_{g_0} \mathbf{x} \mathbf{q}_{(g_0)^{-1}}$ represents rotating $\mathbf{x}$ using g_0. We therefore can define a rotation equivariant transformation

$$S_{g_0}[\mathbf{f}(\mathbf{x}, g)] = \mathbf{f}(\mathbf{x}, (g_0)^{-1} \circ g),$$

where $(g_0)^{-1}$ is the inverse element of g_0. We then can prove that $F = \phi(X)$ is rotation equivariant, since $S_{g_0}[\phi(X)] = \{\mathbf{f}(\mathbf{x}, g_0^{-1} \circ g)\} = \{\mathbf{f}(T_{g_0}[\mathbf{x}], g)\} = \phi(T_{g_0}[X])$. In practice, it is infeasible to compute F for all rotations on SO(3), and EPN therefore leverages a discrete approximation and achieves equivariance regarding a finite subgroup of SO(3). Concretely, EPN discretizes the SO(3) rotation group into the icosahedron rotation group G_g, which contains 60 different rotations originating from the 60 rotational symmetries of a regular icosahedron and computes a rotated KPConv for each $g \in G_g$. This KPConv-based multi-rotation convolution is named PointConv, effectively generating a per-point 2D feature $\mathbf{f} \in \mathbb{R}^{C \times |G|}$ for all $\mathbf{x} \in X$. When the input point cloud X is rotated by some $g \in G_g$, the rotation equivariance guarantees that the $\mathbf{f}$s will be permuted in its G dimension according to g, *e.g.*, its jth feature $\mathbf{f}_j \in \mathbb{R}^C$ will become the jth feature of $\mathbf{f}'$, satisfying $g_{j'} = g \circ g_j$. To further increase the expressivity of the network and to allow information exchange between different coordinates of F, EPN further introduces a GroupConv operation, which passes messages across different gs of $\mathbf{f}(\mathbf{x}, g)$. This message passing leverages an rotation-invariant graph convolutional kernel and therefore would not break the equivariant structure of F. For more detailed information, please refer to EPN [5].

SE(3)-equivariance By combining the translation and rotation equivariance, we can define a rotation- and translation-equivariant transformation $S_A = S_{\mathbf{t}} \circ S_g$, so that EPN is strictly equivariant to all SE(3) transformations in $G_A = \{T_A \mid T_A[X] = T_{\mathbf{t}} \circ T_g[X], \text{for } \forall \mathbf{t} \in \mathbb{R}^3, g \in G_g\}$. Note that G_A is a subgroup of SE(3); we will discuss in section 3.2 how this subgroup approximation effects our task when the input undergoes rotations that are not in G_g.

3.2 Self-Supervised Joint Shape Reconstruction and 6D Pose Estimation

For an input point cloud $X \in \mathbb{R}^{N\times 3}$, we want to learn a network that jointly reconstructs an invariant canonical shape point cloud $Z \in \mathbb{R}^{N_Z\times 3}$ and estimates a 6D object pose $P = (\mathbf{q}, \mathbf{t})$, where N and N_Z are the number of points in X and Z, respectively, and we choose to use quaternion $\mathbf{q}$ as our rotation representation. Here invariant canonical shape reconstruction means the reconstruction should be invariant under all SE(3) transformations of the input point cloud, or in other words, invariant with 6D object pose. Furthermore, we want the reconstruction to be automatically aligned across different object instances under the same object category so that the reconstruction is under a category-level canonical reference frame. For equivariant 6D pose estimation, we want the estimated pose to change by the same SE(3) transformation A applied to the input object point cloud, *e.g.*, $P(T_A[X]) = T_A[P(X)]$ for $\forall A \in$ SE(3) (approximately). Combining these two goals, we can form a self-supervised loop by transforming the invariant canonical reconstruction Z by the 6D transformation T_P involved in estimated P into a posed reconstruction $Y = T_P[Z]$ and enforcing consistency between input point cloud X and this posed reconstruction Y. We will show that, although this self-supervised loop can be formed without leveraging SE(3) equivariance, it is crucial to use an SE(3)-equivariant network for creating a category-level canonical reference, which enables category-level 6D pose estimation.

In this work, we leverage EPN as our SE(3)-equivariant backbone. Our EPN backbone network ϕ alternately performs PointConv and GroupConv in each convolution block while downsampling the points between two convolution blocks. Given an input object point cloud X (partial or complete), we use the EPN backbone ϕ to extract per-point per-rotation SE(3)-equivariant features $F \in \mathbb{R}^{n\times C\times |G_g|} = \phi(X) = \{\mathbf{f}(\mathbf{x}_i, g_j)\}$, where $|\{\mathbf{x}_i\}| = n$ and $g_j \in G_g$. We then feed the features to the following canonical shape reconstruction and 6D pose estimation branches.

SE(3)-invariant Canonical Shape Reconstruction Branch The task of this branch is to learn a SE(3)-invariant canonical shape reconstruction function $\rho : F \in \mathbb{R}^{n\times C\times |G_g|} \rightarrow Z \in \mathbb{R}^{N_Z\times 3}$. We want Z to be invariant under arbitrary SE(3) transformations, *e.g.*, $Z(\rho(\phi(X))) = Z(\rho(\phi(T_A[X])))$ for $\forall A \in$ SE(3). As a relaxation, we instead require Z to be invariant under arbitrary transformation $A \in G_A$. To do so, we only need this learned function ρ to be invariant under all transformations S_A on F, *e.g.*, $Z = \rho(F) = \rho(S_A[F])$ for $\forall A \in$ SE(3), thanks to the equivariance of F.

To implement this invariant function ρ, we perform an elementwise $\text{MLP}_\rho : \mathbf{f} \in \mathbb{R}^C \rightarrow \mathbf{f}_\rho \in \mathbb{R}^{C_\rho}$ to $F = \{\mathbf{f}(\mathbf{x}_i, g_j)\}$ that only operate on each $\mathbf{f}(\mathbf{x}_i, g_j)$ individually, yielding a new set of features $F_\rho = \{\mathbf{f}_\rho(\mathbf{x}_i, g_j) \mid \mathbf{f}_\rho(\mathbf{x}_i, g_j) = \text{MLP}_\rho(\mathbf{f}(\mathbf{x}_i, g_j)) \text{ for } \forall \mathbf{x}_i \in X', g_j \in G\}$. To obtain a global SE(3)-invariant feature F_{inv}, we propose to perform max-pooling over $\{\mathbf{x}_i\}$, as done in PointNet[13]. Note that, for $\forall A \in G_A$, S_A is equivalent to a permutation in the j dimension of $F(\mathbf{x}_i, g_j)$ as well as that of $F_\rho(\mathbf{x}_i, g_j)$, therefore we only need F_{inv} to be permutation-invariant to g_j. This suggests another max-pooling over $\{g_j\}$. Formally, $F_{\text{inv}} = \text{max-pooling}_{i,j}(\{F_\rho(\mathbf{x}_i, g_j) \in \mathbb{R}^{C_\rho}\})$. Finally, we use a MLP-based point generator MLP_Z to transform F_{inv} into $\mathbb{R}^{3N_Z}$ and reshape into a point cloud $Z \in \mathbb{R}^{N_Z\times 3}$. To facilitate a canonical reconstruction, we choose to use $u() = \text{Sigmoid}() - 0.5$ as the final activation so that the reconstruction can be as zero-centered as possible.

Although Z is only strictly invariant when $g \in G_g$, for a rotation g_x not in G_g, we can decompose it into a main component $g_{x_0} = \min_{g_j \in G_g} \text{dist}(g_j, g_x)$ and a small residual rotation Δg_x, so that $g_x = \Delta g_{x_0} \circ g_x$. We can then see that $Z' = \rho(\phi(T_{g_x}[X])) = \rho(\phi(T_{g_{x_0}}[T_{\Delta g_x}[X]])) = \rho(\phi(T_{\Delta g_x}[X]))$. Given that most neural networks are continuous to its input, then $Z' = \rho(\phi(T_{\Delta g_x}[X])) \approx Z = \rho(\phi(X))$ when Δg_x is a sufficiently small rotation so that $T_{\Delta g_x}[X]) \approx X$. Or, in other words, to learn an invariant shape reconstruction, our network now only needs to be invariant to rotations in the quotient group $\text{SO}(3)/G_g$, which renders a much easier learning problem.

This SE(3)-invariance brings important consequences to the reconstructed point cloud Z: 1) if X represents a complete object point cloud, then the reconstructed Z will remain the same for this object under all 6D object poses; 2) for object instances with similar shapes from the same category, the network will lean towards emerging category-level aligned reconstructions, which may significantly simplify the task of the point cloud generation and resemble what happens in Quotient Network[11];

Note that when the input point cloud X only depicts a partial object, e.g. in a depth point clouds, a pose change in the object is not equivalent to a SE(3) transformation to the point cloud, due to the

change in point visibility. We find in our experiments that the emerging category-level alignment still holds but there is no guarantee that the reconstructed point clouds will be complete.

SE(3)-equivariant 6D Pose Estimation Branch The task of this branch is to learn a pose estimation function π from F to an equivariant 6D object pose P. By saying equivariance, we mean that the estimated pose of $X' = T_A[X]$ will under the same 6D transformation T_A that applies to the pose of the input point cloud X, *e.g.*, $\pi(\phi(X')) = \pi(\phi(T_A[X])) = T_A[\pi(\phi(X))] = T_A[P]$.

Inspired by the rotation estimation network introduced in EPN, we propose to predict $|G_g|$ pose hypothesis containing one 6D pose P_j for each $g_j \in G_g$. We therefore propose to learn a function $\pi : F \in \mathbb{R}^{n \times C \times |G_g|} \rightarrow \{P_j\} \in \mathbb{R}^{(3+4) \times |G_g|}$, where 3 is for translation, and 4 for quaterion as our rotation representation. Here we parameterize each 6D pose $P_j = (\mathbf{q}_j, \mathbf{t}_j)$ using four components $(\mathbf{q}_{j_0}, \Delta\mathbf{q}_j, \mathbf{t}_0, \Delta\mathbf{t})$, as explained in the following: $\mathbf{q}_{j_0}$ represents a major discrete quaternion corresponding to $g_j \in G_g$ and satisfies $g_j = \min_{q \in G_g} \text{dist}(g, g_j)$; $\Delta\mathbf{q}_j$ represent the residual quaternion, which can be composed with $\mathbf{q}_j$ to form the full quaternion $\mathbf{q}_j = \mathbf{q}_{j_0}\Delta\mathbf{q}_j$; $\mathbf{t}_0 = \bar{X}$ is the weight center of the input point cloud X and it is equivariant to the translation of X and remains invariant for different gs; and $\Delta\mathbf{t}_j$ represents the residual translation under rotation g_j. This pose parameterization will result in $\mathbf{q}_j = \mathbf{q}_{j_0}\Delta\mathbf{q}_j$ and $\mathbf{t}_j = \mathbf{q}_{j_0}\Delta\mathbf{t}\mathbf{q}_{j_0}^{-1} + \mathbf{t}_0$, which will pose a canonical object point cloud Z into $Z_j^{'} = \mathbf{q}_j Z \mathbf{q}_j{}^{-1} + \mathbf{t}_j$.

To achieve pose equivariance, we adopt a similar strategy to our invariant reconstruction branch but without max-pooling over g_j. We learn an elementwise $\text{MLP}_\pi : \mathbb{R}^C \rightarrow \mathbb{R}^{C_\pi}$ to process each $\mathbf{f}(\mathbf{x}_i, g_j) \in F$ individually, which transforms F into F_π. We then perform max-pooling on F_π over $\{\mathbf{x}_i\}$, yielding an equivariant global feature $F_{\text{equiv}} \in \mathbb{R}^{C_\pi \times |G_g|} = \{\mathbf{f}_{\text{equiv}}(g_j) \in \mathbb{R}^{C_\pi} \mid g_j \in G_g\}$. We further learn a per-rotation $\text{MLP}_P : \mathbb{R}^{C_\pi} \rightarrow \mathbb{R}^7$ to transform each $\mathbf{f}_{\text{equiv}}(g_j)$ to a residual quaternion $\Delta\mathbf{q}_j$ and a residual translation $\Delta\mathbf{t}_j$. This pose estimation branch is equivariant with any SE(3) transformation $A \in G_A$. Similar to our argument for the invariant reconstruction, this subgroup equivariance will significantly ease the learning of pose estimation equivariant with arbitrary SE(3) transformations.

This 6D pose estimation branch has several important differences to the rotation estimation branch used in EPN. First, due to the self-supervised nature of this work, we don't know which rotation g_j is the closet one to the ground truth object rotation and we therefore choose not to predict a probability distribution over g_j. Second, to enable 6D pose estimation, we additionally estimate a residual translation $\Delta\mathbf{t}$. Finally, EPN doesn't constrain the range of $\Delta\mathbf{q}_j$ since it knows which g_j is the best main rotation component and can supervised the residual accordingly. Here, to avoid an overparameterization of rotation, we constrain the residual quaternion $\Delta\mathbf{q}_j$ to be a small quaternion that can only compensate for the rotation discretization introduced by G_g but shouldn't go beyond the zone of each $\mathbf{q}_{j_0}$. Note that the relationship between a quaternion $\mathbf{q} = (q_w, q_x, q_y, q_z)$ and an axis-angle (ω, θ) allows us to constrain the rotation angle of one quaternion easily: $q_w = cos(\theta/2)$, where $theta$ is the rotation angle. Given that the rotation angle between any two nearest neighbouring elements from Icosahedron rotational group G is $\pi/5$, we propose to set $\theta(\Delta\mathbf{q}) < \pi/5$ to cover the whole $SO(3)$ or S^3 in the quaternion space while avoding being too overparameterized. This set a constraint to q_w of $\Delta\mathbf{q}$ to be $q_w \geq \cos(\pi/10)$. Also, for a unit quaternion, $q_w \leq 1$. We can implement these two constraints via a customized activation function: $\cos(\pi/10) + (1 - \cos(\pi/10)) * \text{Sigmoid}()$.

Self-supervised Loss Given the canonical reconstructed point cloud Z and a set of 6D pose hypotheses $\{P_j = (\mathbf{q}_j, \mathbf{t}_j)\}$, we can pose Z using each P_j, rendering posed reconstructions $Y_j = \mathbf{q}_j Z(\mathbf{q}_j)^{-1} + \mathbf{t}_j$. We then can enforce a min-of-N loss between the input observation X and the posed point clouds:

$$\mathcal{L}_{rec} = \min_{j \in \{1, \ldots, |G_g|\}} d(X, Y_j),$$

where $d : \mathbb{R}^{N_X \times 3} \times \mathbb{R}^{N_Y \times 3} \rightarrow \mathbb{R}$ represents a point cloud distance function. For complete object observation, we can choose d to be bidirectional Chamfer distance[8] $d_{CD}(X, Y) = \frac{1}{N_X} \sum_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y} ||\mathbf{x} - \mathbf{y}||^2 + \frac{1}{N_Y} \sum_{\mathbf{y} \in Y} \min_{\mathbf{x} \in X} ||\mathbf{x} - \mathbf{y}||^2$. For partial point cloud observation, due to the incompleteness of X, we can only enforce a unidirectional Chamfer distance $d_{\text{UnidirCD}}(X, Y) = \frac{1}{N_X} \sum_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y} ||\mathbf{x} - \mathbf{y}||^2$, which only enforces each point in the partial point cloud X can find a closed point in Y.

Note that the angle constraint of Δq implemented by constraining the range of its q_w only makes sense when Δq is a unit quaternion. We therefore additionally enforce a regularization loss $L_{reg} = (||\Delta q|| - 1)^2$. The total loss $\mathcal{L} = \mathcal{L}_{rec} + \lambda \mathcal{L}_{reg}$, where λ is a regularization weight.

3.3 Category-Level 6D Object Pose Inference and Evaluation

Category-Level 6D Object Pose Inference Given an object point cloud X (partial or complete), we feed it to our network and generate a reconstruction Z, a set of pose hypotheses $\{P_j\}$, and the correspoinding posed reconstruction $\{Y_j\}$. We can then obtain its category-level 6D object pose estimation $\hat{P} = (\mathbf{q}, \mathbf{t})$ via selecting the best pose hypothesis $\hat{P} = (\mathbf{q}_{j^*}, \mathbf{t}_{j^*})$ that satisfies $j^* = \underset{j \in \{1,\ldots,|G_g|\}}{\operatorname{argmin}} \; d(X, Y_j)$. This best pose hypothesis is the 6D object pose of the observed object instance in the reference frame of reconstruction Z.

Evaluating Category-Level 6D Object Pose To quantitatively evaluate the performance of our self-supervised category-level 6D object pose estimation algorithm on test point cloud data $\{X_k\}$ from a given category, we need to have the ground truth category-level pose annotations $P_k^* = (\mathbf{q}_k^*, \mathbf{t}_k^*)$ for all test data X_k, which uniquely defines a category-level pose reference frame R^*.

Since our estimated poses $\hat{P}_k$ are in the reference frames R_k of reconstruction Z_k, there can be center shift and orientation differences between R_k and R or even among different R_ks. We therefore need to find a consistent reference frame $\hat{R}$ and register it to R^*. We propose the following registration algorithms.

For each test data X_k, we transform it using the inverse transformation of P_k^*, yielding a canonicalized object point cloud $Z_k^* = (\mathbf{q}_k^*)^{-1} X_k \mathbf{q}_k^* - \mathbf{t}_k^*$. We then feed Z_k^* to our network for pose estimation and obtain a pose misalignment $(\tilde{\mathbf{q}}_k, \tilde{\mathbf{t}}_k)$. We propose to use a RANSAC-based method to find a majority consensus of $\tilde{\mathbf{q}}$ and $\tilde{\mathbf{t}}$. Then, for all test data, our final pose prediction can be computed as $(\hat{\mathbf{q}}\tilde{\mathbf{q}}^{-1}, \hat{\mathbf{t}} - \hat{\mathbf{q}}\tilde{\mathbf{q}}^{-1}\tilde{\mathbf{t}}\tilde{\mathbf{q}}\hat{\mathbf{q}}^{-1})$ using our network raw pose prediction $(\hat{\mathbf{q}}, \hat{\mathbf{t}})$.

4 Experiments

Datasets We evalute our method on three different datasets: (1) Complete shapes. We first test our method on complete shapes from ModelNet40 [22] with five categories including airplane, car, chair, sofa and bottle. Random rotation is applied to each input data during training, and 5 different rotations are randomly sampled for each test instance. We follow the original train/test split from the ModelNet40 dataset. (2) Partial observation. We evaluate our method on partial shape observations by rendering the depth images from the complete object instances described above. For each category, we follow a Gaussian distribution to randomly sample view points and view directions. For each category, we generate 30K training images with 6K testing images. The dataset will be made public. (3) Real-world depths. To evaluate the algorithm on realworld multi-object environment, we evaluate our algorithm on the NOCS-REAL 275 dataset [21] . The NOCS-REAL 275 dataset contains a training set with 275K discrete frames of synthetic data generated with 1085 models of instances of the classes from ShapeNet [2]. It also has 7 real videos capturing 3 different instances of objects for each category. The testing set has 3,200 frames collected from 6 real videos containing novel instances. We assume that the ground truth object mask is available in our experiments.

Metrics We perform all the pose evaluation on held-out instances per category. For each category, we report rotational error R_{err}, and translational error T_{err} in the form of mean, and median values, together with 5°5cm accuracy and error percentile curves. For symmetric objects like bottles, we use the angular error on up axis direction.

Baselines. We compare against two supervised methods: EPN and KPConv, which serve as performance oracles. We utilize a tuned implementation of ICP algorithm as traditional unsupervised methods, where we use the same set of 60 rotations from icosahedron group for rotation initializations. Since we assume that we don't have GT complete shapes for testing data, for ICP we randomly select an example shape, and iteratively register the observation to the example shape. There are no existing learning-based unsupervised methods that fit our setting, especially on partial shapes with both 3D rotation and 3D translation, we leave the discussion and detailed comparison with them to supplementary.

Table 1: **Angular error on complete point cloud.** We report mean and median rotation error, and accuracy for 3D pose estimation on rotated ModelNet40 Dataset.

Mean(°) ↓ / Med.(°) ↓ / Acc. ↑	Airplane	Car	Chair	Sofa	Bottle
EPN(supervised)	**3.35 / 1.12 / 0.96**	**9.48 / 1.85 / 0.95**	8.56 / **3.87 / 0.68**	**4.76 / 1.56 / 0.97**	49.72 / 43.19 / 0.03
KPConv(supervised)	14.85 / 10.78 / 0.12	38.16 / 19.26 / 0.06	20.39 / 12.01 / 0.06	128.62 / 134.40 / 0.00	**1.33 / 1.36 / 1.00**
ICP + template shape	8.11 / 1.22 / 0.90	22.76 / 2.94 / 0.69	88.92 / 96.28 /0.10	39.00 / 9.69 / 0.28	76.11 / 54.42 0.04
Ours	23.09 / 1.66 / 0.87	17.24/ 2.13 / 0.89	**7.05** / 4.55 / 0.56	8.87 / 3.22 / 0.68	4.42 / 0.83 / 0.98

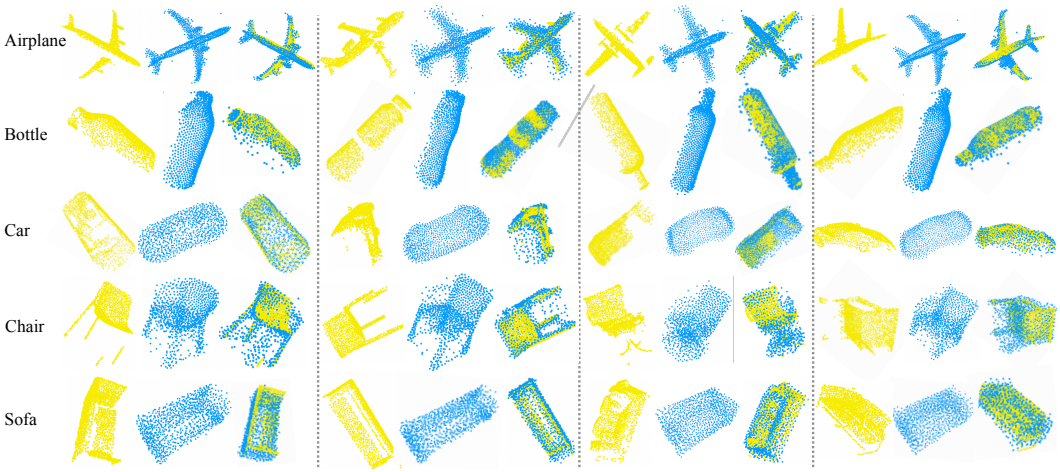


Figure 2: Visualization on the ModelNet partial shapes. Here we show four test instances for each categories: left: input X, middle: canonical reconstruction Z, and right: posed reconstruction Y. Note that Zs have been calibrated.

3D pose estimation on complete shape. As shown in Table 1, while our method is fully self-supervised, it could achieve comparable results to supervised method using SOTA point cloud learning models like EPN and KPConv. On certain categories, our method could even surpass the supervised oracles such as the mean rotational error on chair, and median rotational error on bottle. EPN gives the best supervised performance on all the categories except for bottle, mainly due to its limitation when trying to classify the correct rotational mode under rotational ambiguity. Without the equivariance property on full SO(3) space, KPConv struggles to give an accurate rotation estimation, and our self-supervised pipeline totally beats it on all 4 asymmetric categories, and keep presenting strong performance on symmetric bottle. For simple shapes like airplane and car, ICP algorithm could achieve a reasonable estimation after iterative alignment, but fails at chair and sofa category. It also doesn't work well for the bottle case as the given example shape won't match well with all different bottle instances.

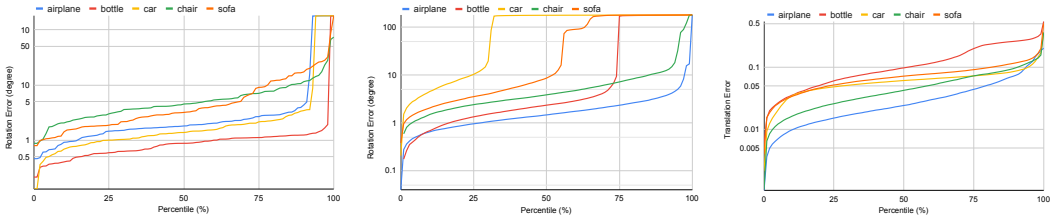
6D pose estimation on partial shapes.

When the input observation is partial shape, there will be more ambiguities in determining object canonical pose. Despite this challenge, we observe that our method is still able to infer well-aligned canonical shapes across different instances, as shown in Figure 2. For some object categories the method is able to infer a complete object shape by training on different view points. However, for bottle, the algorithm only considers the object shape as a 'half' bottle due to the symmetry – partial bottle shape can already cover all possible view points. Quantitatively, results in Table 2 shows that our method achieves the best accuracy on both partial airplane and chair data. It also handles the symmetric object like bottle well, while the two supervised methods suffer from performance degradation due to the increased difficulty.

When there are strongly occluded shape and pose ambiguity, the performance of our model tends to drift. As shown in Figure 3, the rotation error of both sofa and car category has a lot of 180-degree-flip cases, which means the model couldn't distinguish dominant direction like front and back. The error of rotation estimation also contributes to the translational error for car and sofa. For the bottles, we also observe multiple cases where the reconstructed shape is upside down and an considerable offset to the dominant object center which results in the poor performance for translation estimation.

Table 2: **6D pose evaluation on partial depth point cloud.** We show a full evaluation of category-level 6D pose estimation on our ModelNet40 depth dataset using our methods. We further show performance of supervised methods for better comparison, together with an ICP baseline.

$R_{err} \downarrow$ / $T_{err} \downarrow$ / Acc. $\uparrow$	Airplane	Car	Chair	Sofa	Bottle
EPN(supervised)	4.43 / 0.09 / 0.10	93.18 / 0.27 / 0.00	3.23 / 0.09 / 0.13	3.01 / 0.08 / 0.15	15.36 / 0.11 / 0.03
KPConv(supervised)	7.37 / 0.08 / 0.02	113.37/ 0.12 / 0.00	14.58 / 0.07 / 0.02	16.90 / 0.06 / 0.02	**1.97** / **0.09** / **0.23**
ICP + template shape	1.54/0.08/0.05	14.97/0.04/0.34	95.01/0.06/0.03	114.97/0.07/0.04	83.71/0.06/0.04
Ours	**1.46** / **0.02** / **0.76**	177.29 / **0.06** / **0.10**	3.84 / **0.04** / **0.44**	8.60 / 0.07 / 0.09	2.33 / 0.10 / 0.19



(a) complete shape rotation error (b) partial shape rotation error (c) partial shape translational error

Figure 3: **Error percentiles for complete and partial shapes from ModelNet40.**

6D pose estimation on real category-level data To test our method under a more realistic setting and explore the limits of its performance, we examine our method on the challenging real world depth data from NOCS-REAL275[21] dataset, where sensor noises and heavy occlusions in the real world cluttered scene further complicates the problem. We consider five different experiment settings: 1) perform supervised pretraining on CAMERA[21] dataset, similar to EPN; 2) self-supervisedly finetune the pretrained weight of 1) using the small train split of NOCS; 3) self-supervisedly pretrain on CAMERA; and finally 4) self-supervisedly finetune the pretrained weight of 3) on NOCS-train, together with 5) for ICP algorithm. As shown in Table 3, our fully unsupervised method beats ICP on almost all categories and performs quite well on bottle, bowl, and can, without using any labelled or real data. Self-supervised finetuning on real data further improves its performance. For supervised pretrained model,

Table 3: **6D pose evaluation on NOCS-REAL 275 dataset.**

$R_{err} \downarrow$ / $T_{err} \downarrow$ / Acc. $\uparrow$	Camera	Laptop	Mug	Bottle	Bowl	Can
1)	39.49 / **0.05** / 0.06	**4.06** / 0.03/ 0.55	**4.75** / 0.04 / **0.53**	3.01 / **0.03** / 0.78	**3.45/ 0.04** / **0.73**	2.41/ **0.04** / 0.86
2)	**15.26** / 0.08 / **0.07**	8.10 / **0.02** / 0.39	5.03 / **0.03** / 0.49	**2.60** / 0.07 / **0.86**	5.14 / 0.04 / 0.48	2.52 /0.13 / **0.87**
3)	167.75 / 0.06 / 0.0	13.64 / 0.06 / 0.01	126.62 / 0.22 / 0.0	4.53/0.05/0.57	7.52 / 0.04 / 0.22	3.88 / 0.13 / 0.68
4)	79.18/0.07/0.00	5.32/0.02/48.38	134.29/0.12/0.04	3.51/0.05/0.72	5.94/0.08/0.38	2.68/0.16/0.87
5)	115.75 / 0.11 / 0	22.13 / 0.12 / **0.72**	127.31 / 0.07 / 0.35	22.13 / 0.12 / 0.72	9.93 / 0.06 / 0.20	**2.27** / 0.08 / 0.89

5 Conclusion

This paper has presented a novel self-supervised network that is capable of category-level 6D object pose estimation. The system operates without CAD models or annotations, and it achieves impressive accuracy on both synthetic and real-world category-level pose estimation benchmarks. This work is the first to demonstrate the power of SE(3)-equivariance point cloud networks for 6D pose estimation, especially on partial object point clouds. As limitations, the performance of SE(3) equivariant networks may degrade when objects exhibit symmetries, or viewpoint distribution is highly biased, or occlusions are always heavy.

References

[1] Melinos Averkiou, Vladimir G Kim, and Niloy J Mitra. Autocorrelation descriptor for efficient co-alignment of 3D shape collections. In *Computer Graphics Forum*, volume 35, pages 261–271. Wiley Online Library, 2016.

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

[3] Mohamed Chaouch and Anne Verroust-Blondet. Alignment of 3D models. *Graphical Models*, 71(2):63–76, 2009.

[4] Dengsheng Chen, Jun Li, Zheng Wang, and Kai Xu. Learning canonical shape space for category-level 6D object pose and size estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11973–11982, 2020.

[5] Haiwei Chen, Shichen Liu, Weikai Chen, and Hao Li. Equivariant point network for 3D point cloud analysis. 2021.

[6] Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. In *European Conference on Computer Vision*, pages 139–156. Springer, 2020.

[7] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas Guibas. Vector neurons: A general framework for SO(3)-equivariant networks. *arXiv preprint arXiv:2104.12229*, 2021.

[8] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[9] Fabian B Fuchs, Daniel E Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *arXiv preprint arXiv:2006.10503*, 2020.

[10] Xiaolong Li, He Wang, Li Yi, Leonidas J. Guibas, A. Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3706–3715, 2020.

[11] Eloi Mehr, André Lieutier, Fernando Sanchez Bermudez, Vincent Guitteny, Nicolas Thome, and Matthieu Cord. Manifold learning in quotient spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9165–9174, 2018.

[12] Adrien Poulenard, Marie-Julie Rakotosaona, Yann Ponty, and Maks Ovsjanikov. Effective rotation-invariant point cnn with spherical harmonics kernels. In *2019 International Conference on 3D Vision (3DV)*, pages 47–56. IEEE, 2019.

[13] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[14] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Domain transfer for 3D pose estimation from color images without manual annotations. In *Asian Conference on Computer Vision*, pages 69–84. Springer, 2018.

[15] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, and Rudolph Triebel. Augmented autoencoders: Implicit 3D orientation learning for 6D object detection. *International Journal of Computer Vision*, 128(3):714–729, 2020.

[16] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019.

[17] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

[18] Meng Tian, Marcelo H Ang, and Gim Hee Lee. Shape prior deformation for categorical 6D object pose and size estimation. In *European Conference on Computer Vision*, pages 530–546. Springer, 2020.

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

[20] Gu Wang, Fabian Manhardt, Jianzhun Shao, Xiangyang Ji, Nassir Navab, and Federico Tombari. Self6D: Self-supervised monocular 6D object pose estimation. In *European Conference on Computer Vision*, pages 108–125. Springer, 2020.

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

[22] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

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

[24] Yang You, Yujing Lou, Qi Liu, Yu-Wing Tai, Lizhuang Ma, Cewu Lu, and Weiming Wang. Pointwise rotation-invariant network with adaptive sampling and 3D spherical voxel convolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12717–12724, 2020.

[25] Yongheng Zhao, Tolga Birdal, Jan Eric Lenssen, Emanuele Menegatti, Leonidas Guibas, and Federico Tombari. Quaternion equivariant capsule networks for 3D point clouds. In *European Conference on Computer Vision*, pages 1–19. Springer, 2020.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] supplementary material
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] in the section of conclusion.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] supplementary material
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] supplementary material
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]