ConDor: Self-Supervised Canonicalization of 3D Pose for Partial Shapes

Rahul Sajnani[1] Adrien Poulenard[2] Jivitesh Jain[1] Radhika Dua[3]
Leonidas J. Guibas[2] Srinath Sridhar[4]
[1]RRC, IIIT-Hyderabad [2]Stanford University [3]KAIST [4]Brown University
ivl.cs.brown.edu/ConDor

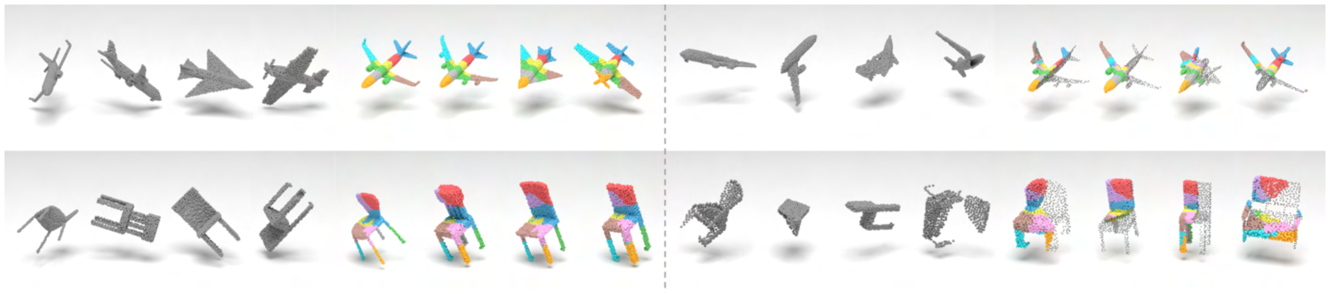

Figure 1. **ConDor** is a self-supervised method that learns to **C**an**on**icalize the 3**D or**ientation and position (3D pose) for full and partial shapes. (***left***) Our method takes un-canonicalized 3D point clouds (gray) from different categories as input and produces consistently canonicalized outputs (colored). (***right***) Our method can also operate on partial point clouds (missing part of shape shown only for visualization). In addition, ConDor can also learn consistent co-segmentation of shapes without supervision, visualized as colored parts.

Abstract

Progress in 3D object understanding has relied on manually "canonicalized" shape datasets that contain instances with consistent position and orientation (3D pose). This has made it hard to generalize these methods to in-the-wild shapes, e.g.*, from internet model collections or depth sensors.* ***ConDor*** *is a self-supervised method that learns to* ***C****an****on****icalize the 3****D or****ientation and position for full and partial 3D point clouds. We build on top of Tensor Field Networks (TFNs), a class of permutation- and rotation-equivariant, and translation-invariant 3D networks. During inference, our method takes an unseen full or partial 3D point cloud at an arbitrary pose and outputs an equivariant canonical pose. During training, this network uses self-supervision losses to learn the canonical pose from an un-canonicalized collection of full and partial 3D point clouds. ConDor can also learn to consistently co-segment object parts without any supervision. Extensive quantitative results on four new metrics show that our approach outperforms existing methods while enabling new applications such as operation on depth images and annotation transfer.*

1. Introduction

Humans have the ability to recognize 3D objects in a wide variety of positions and orientations (poses) [40], even if objects are occluded. We also seem to prefer certain *canonical views* [10], with evidence indicating that an object in a new pose is *mentally rotated* to a canonical pose [47] to aid recognition. Inspired by this, we aim to build scene understanding methods that reason about objects in different poses by learning to map them to a canonical pose without explicit supervision.

Given a 3D object shape, the goal of **instance-level canonicalization** is to find an *equivariant frame* of reference that is consistent relative to the geometry of the shape under different 3D poses. This problem can be solved if we have shape correspondences and a way to find a distinctive equivariant frame (*e.g.*, PCA). However, it becomes significantly harder if we want to operate on different 3D poses of different object instances that lack correspondences. This **category-level canonicalization** problem has received much less attention despite tremendous interest in category-level 3D object understanding [8, 11, 14, 25, 26, 31, 56]. Most methods rely on data augmentation [23], or manually annotated datasets [3, 56] containing instances that are consistently positioned and oriented within each category [44, 48, 52]. This has prevented broader application of these methods to un-canonicalized data sources, such as online model collections [1]. The problem is further exacerbated by the difficulty of canonicalizing partial shape observations (*e.g.*, from depth maps [36]), or symmetric objects that require an understanding of inter-instance part relationships. Recent work addresses these limitations using weakly-supervised [15, 38] or self-supervised learning [13, 29, 43, 46], but cannot handle partial 3D shapes, or is limited to canonicalizing only orientation.

We introduce **ConDor**, a method for self-supervised category-level **C**an**on**icalization of the 3**D** p**o**se of pa**r**tial shapes. It consists of a neural network that is trained on an un-canonicalized collection of 3D point clouds with inconsistent 3D poses. During inference, our method takes a full or partial 3D point cloud of an object at an arbitrary pose, and outputs a canonical rotation frame and translation vector. To enable operation on instances from different categories, we build upon Tensor Field Networks (TFNs) [49], a 3D point cloud architecture that is equivariant to 3D rotation and point permutation, and invariant to translation. To handle partial shapes, we use a two-branch (Siamese) network with training data that simulates partiality through shape slicing or camera projection. We introduce several losses to help our method learn to canonicalize 3D pose via self-supervision. A surprising feature of our method is the (optional) ability to learn consistent part co-segmentation [6] across instances without any supervision (see Figure 1).

Given only the recent interest, **standardized metrics** for evaluation of canonicalization methods have not yet emerged. We therefore propose four new metrics that are designed to evaluate the consistency of instance- and category-level canonicalization, as well as consistency with manually pre-canonicalized datasets. We extensively evaluate the performance of our method using these metrics by comparing with baselines and other methods [43, 46]. Quantitative and qualitative results on common shape categories show that we outperform existing methods and produce consistent pose canonicalizations for both full and partial 3D point clouds. We also demonstrate previously difficult **applications** enabled by our method such as operation on partial point clouds from depth maps, keypoint annotation transfer, and expanding the size of existing datasets. To sum up, our contributions include:

- A self-supervised method to canonicalize the 3D pose of full point clouds from a variety of object categories.
- A method that can also handle **partial** 3D point clouds.
- New metrics to evaluate canonicalization methods, extensive experiments, and new applications.

2. Related Work

Canonical object representations in human perception have been extensively studied as mental rotation [40, 47], shape constancy and equivalence [30], and canonical views [10]. We review related work that studies or uses canonicalization for machine perception of 3D scenes.

3D Scene Understanding: Invariance and equivariance to 3D pose in tasks such as shape classification, reconstruction and registration was initially achieved using hand-crafted features [17, 37, 39]. With machine learning, these features were replaced with learned features [51, 53, 59], but 3D data introduces challenges in learning invariant features [33]. Data augmentation by sampling the space of 3D poses for each object is one way to address this problem [23] but results in longer training and larger networks. Category-level object reconstruction methods have gained significant attention with representations ranging from voxel grids [8, 25], implicit surfaces [31], parametric surfaces [14, 22], point clouds [57], and depth images [61]. Almost all of these methods rely on manually pre-canonicalized datasets like ShapeNet [4] and ModelNet40 [56] to learn inductive biases for effective learning [48]. Neural networks have also been successfully used for supervised [27] and unsupervised [6] segmentation of object parts.

3D Neural Networks: Numerous neural networks have been proposed for processing 3D data represented as voxels [24, 34, 56], multiple views [45], point clouds [33, 35, 54] or meshes [16, 58]. For 3D point clouds, PointNet and related methods achieve point permutation equivariance and translation equivariance, but not rotation equivariance. Spherical CNNs [9] and Tensor Field Networks (TFNs) [32, 49] address this limitation. We use 3D point clouds as our shape representation and TFNs to achieve equivariance to permutation, translation, and rotation.

Supervised Canonicalization: Supervised canonicalization of shapes enables applications such as instance-level camera pose estimation [42] or human pose estimation [18, 41]. It can also be useful for *category-level* reasoning, for example 6 DoF pose estimation [50, 52]. However, these methods are limited to learning from data with ground truth canonicalization making it hard to generalize to real data.

Our method is most related to recent work on weakly supervised [15, 38], or self-supervised learning of canonicalization of semantic keypoints [29] and point clouds [43,46]. Unlike these methods, we can canonicalize both orientation and translation for partial shapes.

3. Background

3D Pose Canonicalization: The 3D pose of an object refers to its 3D position and orientation in space specified by an intrinsic object-centric reference frame (defined by an origin and orthonormal rotation). Having a consistent intrinsic frame across different shapes is critical in many problems [5, 12, 32, 62]. We denote such a consistent intrinsic frame as a **canonical frame**. This frame transforms together with the object, *i.e.*, it is equivariant. The object pose is constant relative to the canonical frame – we call this our **canonical pose**.

In **instance-level 3D pose canonicalization**, our goal is to find a consistent canonical frame across different poses of the same object instance (Figure 2, top). In **category-level 3D pose canonicalization**, we want a canonical frame that is consistent with respect to the geometry and local shape across different object instances (Figure 2, bottom). Any equivariant frame that is consistent across shapes is a valid

Figure 2. A canonical frame visualized for (***top***) the same instance in different 3D poses, and (***bottom***) different instances in different 3D poses. Partial shapes with amodal frame shown in last column.

canonical frame – this allows us to compare canonicalization with manually-labeled ground truth (see Section 6.1). In addition to full shapes, we also consider partial shape canonicalization for which we define an *amodal* canonical frame as shown in Figure 2.

Tensor Field Networks: Our method estimates a canonical frame for 3D shapes represented as point clouds. For this task, we use Tensor Field Networks [49] (TFNs), a 3D architecture that is equivariant to point permutation and rotation, and invariant to translation. Given a point cloud $X \in \mathbb{R}^{3 \times K}$ and a integer (aka type) $\ell \in \mathbb{N}$, a TFN can produce global (type ℓ) feature vectors of dimension $2\ell + 1$ stacked in a matrix $F^\ell \in \mathbb{R}^{(2\ell+1) \times C}$, where C is user-defined number of channels. $F^\ell_{:,j}(X)$ satisfies the equivariance property $F^\ell_{:,j}(RX) = D^\ell(R) F^\ell_{:,j}(X)$, where $D^\ell : \mathrm{SO}(3) \to \mathrm{SO}(2\ell + 1)$ is the so-called Wigner matrix (of type ℓ) [7, 20, 21]. Please see [2, 32, 49, 55] for details.

4. Method

Given a point cloud $X \in \mathbb{R}^{3 \times K}$ denoting a full or partial shape from a set of non-aligned shapes, our goal is to estimate its rotation $\mathcal{R}(X)$ (canonical frame) sending X to a canonical pose. For a partial shape $Y \subset X$ we also learn a translation $\mathcal{T}(Y)$ aligning Y with X in the canonical frame. We achieve this by training a neural network on 3D shapes in a self-supervised manner (see Figure 3).

4.1. Learning to Canonicalize Rotation

We first discuss the case of canonicalizing 3D rotation for full shapes. Given a point cloud X, our approach estimates a rotation-invariant point cloud X^c, and an equivariant rotation E that rotates X^c to X. Note that for full shapes, translation can be canonicalized using mean centering [29], but this does not hold for partial shapes.

Rotation Invariant Point Cloud/Embedding: To estimate a rotation-invariant point cloud, we build on top of a permutation-, rotation-equivariant and translation-invariant neural network architecture: Tensor Field Networks (TFNs) [49] with equivariant non-linearities for TFNs [32]. Given X, we use a TFN [32] to produce global **equivariant features** F^ℓ, with columns $F^\ell_{:,j}$ as described in Section 3.

The central observation of [32] is that the features F have the same rotation equivariance property as coefficients of spherical functions in the spherical harmonics basis, and can therefore be treated as such. We exploit this property by embedding the shape using the spherical harmonics basis and using the global TFN features F as coefficients of this embedding. Since the input to the spherical harmonics embedding and the coefficients rotate together with the input shape, they can be used to define a rotation and translation **invariant embedding** of the shape. Formally, let $Y^\ell(x) \in \mathbb{R}^{2\ell+1}$ be the vector of degree ℓ spherical harmonics which are homogeneous polynomials defined over $\mathbb{R}^3$. We define a rotation invariant embedding of the shape as the dot products

$$H^\ell_{ij} := \langle F^\ell_{:,j}, Y^\ell(X_i) \rangle, \tag{1}$$

where i is an index to a single point on the point cloud, and j is the channel index as in Section 3. Both sides of the dot product are rotated by the same Wigner rotation matrix when rotating the input pointcloud X making H invariant to rotations of X. The input point cloud is mean-centered to achieve invariance to translation. Note that we can use any functional basis of the form : $x \mapsto (\varphi^r(\|x\|) Y^\ell(x))_{r\ell}$, where $(\varphi^r)_r$ are real valued functions to define H.

We use the rotation invariant embedding corresponding to $\ell = 1$ (degree 1) to produce a 3D **invariant shape** through a linear layer on top of H. Note that degree 1 spherical harmonics are the x, y, z coordinates of the input point cloud since $Y^1(x) = x$. As we show in Appendix B, other choices for ℓ enable us to learn consistent co-segmentation without supervision. The 3D rotation invariant shape is given by:

$$X^c_i := \sum_j W_{:,j} H^1_{ij} = W (F^1)^\top X_i. \tag{2}$$

We obtain our canonical frame as described in Section 3 as $\mathcal{R}(X) = W(F^1)^\top$ where W is the learnable weights matrix of the linear layer.

Rotation Equivariant Embedding: Next, we seek to find an equivariant rotation that transforms X^c to X. In addition to the equivariant features F, our TFN also outputs a 3D equivariant frame E which we optimise to be a rotation matrix. E satisfies the equivariance relation $E(R.X) = RE(X)$ so that the point cloud $E(X)X^c$ is rotation equivariant. Note that we could have chosen $E(X) = \mathcal{R}(X)^\top$ but we instead choose to learn $E(X)$ independently as this approach generalizes to the case of non-linear embeddings

Figure 3. **ConDor**. (***left***) Our method learns to canonicalize rotation by estimating an equivariant pose $E(X)$ and an invariant point cloud X^c of an input shape X. A self-supervision loss ensures that the input and transformed canonical shapes match. (***right***) To handle translation in partial shapes, we train a two-branch (Siamese) architecture, one taking the full shape and the other taking an occluded (*e.g.*, via slicing) version of the full shape as input. Various losses ensure that the feature embeddings of the full and partial shapes match. We predict the amodal barycenter of the full shape $T(O(X))$ from the partial shape to canonicalize for position.

(*e.g.*, with values other than $\ell = 1$ in Equation (2)) which we use for unsupervised segmentation in Appendix B.

Using E, we can transform our 3D invariant embedding X^c back to the input equivariant embedding and compare it to the input point cloud. To handle situations with high occlusion and symmetric objects we estimate P equivariant rotations and choose the frame that minimizes the L^2 norm between corresponding points in the input and the predicted invariant shape.

4.2. Learning to Canonicalize Translation

Next, we discuss canonicalizing 3D translation for **partial point clouds**, *e.g.*, acquired from depth sensors or LiDAR. As noted, translation canonicalization for full shapes is achieved using mean centering [29]. Thus, our approach in Section 4.1 is sufficient for 3D pose canonicalization of full shapes. However, partial shapes can have different centroids depending on how the shape was occluded. To address this issue, we extend our approach to additionally find a **rotation-equivariant translation** $\mathcal{T} \in \mathbb{R}^3$ that estimates the difference between the barycenter of the full and partial shape from the mean-centered partial point cloud that translates it to align with the full shape in the input frame.

In practice, we operationalize the above idea in a two-branch Siamese architecture as shown in Figure 3. We slice the input point cloud to introduce synthetic occlusion. We penalize the network by ensuring semantic consistency between the full and the partial point cloud. Furthermore, our network predicts an amodal translation vector that captures the barycenter of the full shape from the partial input shape.

4.3. Unsupervised Co-segmentation

A surprising finding is that our method can be used for unsupervised part co-segmentation [6] of full and partial shapes with little modification. This result is enabled by finding the rotation invariant embedding H in Equation (1) corresponding to all $\ell > 0$ to produce a **non-linear invariant embedding**. To obtain a consistent rotation invariant part segmentation, we segment the input shape into N parts by learning an MLP on top of the rotation invariant embedding. The part label of each point in the input point cloud is given by $S_i := \mathrm{softmax}[\mathrm{MLP}(H)_i]$. Results visualized in the paper include these segmentations as colored labels. Please see the supplementary material for more details.

5. Self-Supervised Learning

5.1. Loss Functions

A key contribution of our work is to demonstrate that 3D pose canonicalization can be achieved through self-supervised learning as opposed to supervised learning from labeled datasets [4, 56]. We now list the loss functions that enable this. Additionally, we describe losses that prevent degenerate results, handle symmetric shapes, and enable unsupervised segmentation. We begin with full shapes.

Canonical Shape Loss: Our primary self-supervision signal comes from the canonical shape loss that tries to minimize the L^2 loss between the rotation invariant point cloud X^c transformed by the rotation equivariant rotation E with the input point cloud X. It is worth noting that X^c and X are in correspondence because our method is permutation equivariant and we extract point-wise embeddings. For each point i in a point cloud of size K, we define the canonical shape loss to be

$$\mathcal{L}_{canon} = \frac{1}{K} \sum_i \|EX_i^c - X_i\|_2. \tag{3}$$

We empirically observe that our estimation of E can be flipped 180° or X^c can become a degenerate shape when the object class has symmetry or heavy occlusions. To mitigate this issue, we estimate P equivariant rotations E_p and choose the one that minimizes the above loss.

Orthonormality Loss: The equivariant rotation E estimated by our method must be a valid rotation in $\mathrm{SO}(3)$, but this cannot be guaranteed by the TFN. We therefore add a loss to constrain E to be orthonormal by minimizing its difference to its closest orthonormal matrix. We achieve this using the SVD decomposition of $E = U\Sigma V^\top$ and enforcing unit eigenvalues with the loss

$$\mathcal{L}_{ortho} = \|UV^\top - E\|_2. \tag{4}$$

Separation Loss: When estimating P equivariant rotations E_p, our method could learn a degenerate solution where all E_p are similar. To avoid this problem, we introduce a separation loss that encourages the network to estimate different equivariant rotations as

$$\mathcal{L}_{sep} = -\frac{1}{9P}\sum_{i\neq j} ||E_i - E_j||_2 \,. \tag{5}$$

Restriction Loss: We next turn our attention to partial shapes. Similar to full shapes, we compute the canonical shape, orthonormality and separation losses. We assume that a partial shape is a result of a cropping operator $\mathcal{O}$ that acts on a full point cloud X to select points corresponding to a partial version $\mathcal{O}(X) \subseteq X$. In practice, our cropping operator is slicing or image projection (see Section 5.2). During training, we train two branches of our method, one with the full shape and the other with a partial shape generated using a random sampling of $\mathcal{O}$. We then enforce that the invariant embedding for partial shapes is a restriction of the invariant embedding of the full shape X using the loss

$$\mathcal{L}_{rest} = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \|\mathcal{O}[\hat{X}^c]_i - \widehat{\mathcal{O}[X]}^c_i\|_2^2, \tag{6}$$

where $\mathcal{S}$ is the set of valid indices of points in both X and $\mathcal{O}(X)$, and the hat indicates mean-centered point clouds. During inference, we do not require the full shape and can operate only with partial shapes. Empirically, we observe that our method generalizes to different cropping operations between training and inference (see Section 6.3)

Amodal Translation Loss: Finally, to align the mean-centered partial shape with the full shape, we estimate the barycenter of the full shape after the occlusion operation $\overline{\mathcal{O}[X]}$ from the partial shape only using a rotation-equivariant translation vector $\mathcal{T}(\widehat{\mathcal{O}[X]})$ by minimizing

$$\mathcal{L}_{amod} = \|\mathcal{T}(\widehat{\mathcal{O}[X]}) - \overline{\mathcal{O}(X)}\|_2^2. \tag{7}$$

Unsupervised Part Segmentation Losses: A surprising finding in our method is that we can segment objects into parts consistently across instances without any supervision (see Figure 1). This is enabled by interpreting higher degree invariant embedding H^ℓ as a feature for unsupervised segmentation. Our losses are based on the localization and equilibrium losses of [46]. We refer the reader to [46] and the supplementary document for details on these losses. Note that [46] need to perform segmentation to enable rotation canonicalization, while it is optional for us.

5.2. Network Architecture & Training

Our method is trained on a collection of un-canonicalized shapes $\mathcal{X}$, and partial shapes randomly generated using a suitable operator $\mathcal{O}$. We report two kinds of partiality: slicing and image projection (*i.e.*, depth maps). We borrow our TFN architecture from [32] and use the ReLU non-linearity in all layers. We use 1024 and 512 points for full and partial point cloud. Our method predicts 5 canonical frames for every category. Our models are trained for 45,000 iterations for each category with the Adam [19] optimizer with an initial learning rate of 6×10^{-4}. We set a step learning rate scheduler that decays our learning rate by a factor of 10^{-1} every 15,000 steps. Our models are trained on Linux with Nvidia Titan V GPUs – more details in the supplementary document.

6. Experiments

We present quantitative and qualitative results to compare our method with baselines and existing methods, justify design choices, and demonstrate applications.

Datasets: For full shapes, we use un-canonicalized shapes from ShapeNet (Core) [3] and ModelNet40 [56]. For ShapeNet, our data split [11, 46] has 31,747 train shapes, and 7,943 validation shapes where each shape is a 3D point cloud with 1024 points sampled using farthest point sampling. The shapes are from 13 classes: airplane, bench, cabinet, car, chair, monitor, lamp, speaker, firearm, couch, table, cellphone, and watercraft. For ModelNet40 [56], we use 40 categories with 12,311 shapes (2,468 test). For partial shapes, we either randomly slice shapes from the above datasets, or we use the more challenging ShapeNet-COCO dataset [44] that contains objects viewed from multiple camera angles and mimics occlusions from depth sensors. While all these datasets are already pre-canonicalized, we use this information **only for evaluation** – our method is trained on randomly transformed un-canonicalized shapes $X \in \mathcal{X}$ from these datasets.

6.1. Canonicalization Metrics

Most work on canonicalization evaluates performance indirectly on downstream tasks such as segmentation or registration [43, 46]. This makes it hard to disentangle canonicalization performance from task performance. We contribute four new metrics that measure different aspects of 3D pose canonicalization while disentangling performance from downstream tasks. The first three of these metrics evaluate rotation assuming mean-centering, while the last metric measures translation errors for partial shapes.

Instance-Level Consistency (IC): The IC metric is designed to evaluate how well a method performs for canonicalizing the 3D rotation of the *same shape instance*. For each shape in the dataset, we obtain another copy of it by

Table 1. Full shape canonicalization compared to a PCA baseline, Canonical Capsules (CaCa) [46] and Compass [43], and full (F) and full+partial (F+P) versions of our method. We outperform methods on most categories and metrics.

	bench	**cabinet**	**car**	**cellph.**	**chair**	**couch**	**firearm**	**lamp**	**monitor**	**plane**	**speaker**	**table**	**water.**	**avg.**	**multi**
Instance-Level Consistency (IC) ↓															
PCA	0.0573	0.0350	0.0477	0.0276	0.0974	0.0628	0.0324	0.0755	0.0480	0.0502	0.0491	0.0727	0.0400	0.0535	0.0535
CaCa [46]	0.0630	0.1567	0.0426	0.0823	0.0253	0.1479	0.0084	**0.0372**	0.0748	**0.0093**	0.1540	0.0787	0.0270	0.0698	0.0395
Compass [43]	0.1030	0.0816	0.0790	0.0664	0.0791	0.0766	0.0748	0.0495	0.0638	0.0610	0.0721	0.0641	0.0430	0.0703	0.0507
Ours (F)	**0.0225**	0.0346	**0.0191**	**0.0234**	**0.0221**	**0.0221**	**0.0081**	0.0454	0.0283	0.0163	0.0787	0.0523	0.0270	**0.0308**	0.0394
Ours (F+P)	0.0696	**0.0288**	0.0230	0.0263	0.0235	0.0222	0.0084	0.0403	**0.0242**	0.0144	**0.0678**	**0.0361**	**0.0236**	0.0314	**0.0329**
Category-Level Consistency (CC) ↓															
Ground truth	0.0980	0.1460	0.0578	0.0733	0.1191	0.0955	0.0536	0.2147	0.1088	0.0673	0.1709	0.1444	0.0915	0.1108	0.1108
PCA	**0.0976**	**0.1055**	0.0654	**0.0600**	0.1389	0.0937	0.0527	0.1802	**0.0970**	0.0731	**0.1397**	0.1479	**0.0816**	0.1026	0.1026
CaCa [46]	0.1134	0.1742	0.0730	0.1033	0.1220	0.1919	**0.0493**	0.1888	0.1186	0.0684	0.1840	0.1660	0.0883	0.1262	0.1132
Compass [43]	0.1654	0.1348	0.1077	0.0931	0.1522	0.1175	0.1258	0.1833	0.1266	0.1019	0.1579	0.1626	0.0942	0.1325	0.1283
Ours (F)	0.1043	0.1067	**0.0575**	0.0612	**0.1135**	**0.0869**	0.0525	**0.1754**	0.0988	**0.0681**	0.1504	0.1475	0.0851	**0.1006**	0.1035
Ours (F+P)	0.1250	0.1065	0.0581	0.0635	0.1145	0.0874	0.0500	0.1844	0.1001	0.0679	0.1477	**0.1432**	0.0912	0.1030	**0.1005**
Ground Truth Consistency (GC)↓															
PCA	0.0760	0.1047	**0.0208**	**0.0390**	0.1190	0.0799	0.0261	0.1366	0.0862	0.0460	0.1280	0.1267	0.0645	0.0810	**0.0810**
CaCa [46]	0.0761	**0.0688**	0.0529	0.0667	0.0943	0.1812	0.0330	0.1592	0.0897	0.0266	**0.0744**	0.1401	0.0683	0.0870	0.1060
Compass [43]	0.1599	0.1586	0.0892	0.0851	0.1504	0.1160	0.1214	0.1654	0.1231	0.0975	0.1552	0.1554	0.0804	0.1275	0.1247
Ours (F)	**0.0671**	0.1131	0.0257	0.0511	0.0526	0.0585	0.0359	0.1399	0.0674	**0.0255**	0.1505	0.0779	0.0746	0.0723	0.0902
Ours (F+P)	0.1115	0.1134	0.0230	0.0553	**0.0509**	**0.0537**	**0.0223**	**0.1274**	**0.0650**	0.0286	0.1456	**0.0738**	**0.0477**	**0.0706**	0.0843

applying a rotation from $\mathbf{R}$, a user-defined set of random rotations (we use 120 rotations). We then compute the 2-way Chamfer Distance (CD), to handle classes with symmetries such as tables, between the canonicalized versions of the shapes (with superscript ^{c}). We expect this to be as small as possible for better canonicalization. The average IC metric is given as:

$$\text{IC} := \frac{1}{|\mathcal{X}||\mathbf{R}|} \sum_{X_i \in \mathcal{X}} \sum_{R_j \in \mathbf{R}} \text{CD}[(R_j.X_i)^c, X^c].$$

Category-Level Consistency (CC): The CC metric is designed to evaluate the quality of 3D rotation canonicalization between *different shape instances*. For each shape X in the dataset, we pick N other shapes to form a set of comparison shapes $\mathcal{N}$. We then follow a similar approach as IC and compute the 2-way Chamfer Distance between each shape and its N possible comparison shapes. Intuitively, we expect this metric to be low if canonicalization is consistent across different instances. Ideally, we want to evaluate this metric for all possible comparison shapes, but to reduce computation time, we pick $N = 120$ random comparison shapes. The average CC metric is given as:

$$\text{CC} := \frac{1}{|\mathcal{X}|N} \sum_{X_i \in \mathcal{X}} \sum_{X_j \in \mathcal{N}} \text{CD}[X_i^c, X_j^c].$$

Ground Truth Consistency (GC): The GC metric is designed to compare estimated canonicalization with manual ground truth pre-canonicalization in datasets like ShapeNet and ModelNet40. For perfect canonicalization, the predicted canonical shape should be a constant rotation away from ground truth shape. Given the predicted canonicalizing frames $\mathcal{R}(X_j), \mathcal{R}(X_k)$ for aligned shapes $X_j, X_k \in \mathcal{X}$, we induce the same canonicalization on any other shape $X_i \in \mathcal{X}$ and compute the 2-way CD between them.

$$\text{GC} := \frac{1}{|\mathcal{X}|^3} \sum_{X_i, X_j, X_k \in \mathcal{X}} \text{CD}[\mathcal{R}(X_j).X_i, \mathcal{R}(X_k).X_i].$$

We note that manual canonicalization, which is based on human semantic understanding of shapes, does not necessarily match with this paper's notion of canonicalization which is founded on geometric similarity. Nonetheless, this metric provides a way to compare with human annotations.

Translation Error (TE): To measure error in translation for partial shapes, we compute the average L^2 norm between the estimated amodal translation and ground truth amodal translation – this has the same form as $\mathcal{L}_{amod}$ in Section 5.1. Note that we have the ground truth amodal translation for our datasets since partial shapes are generated from the full shapes using an occlusion function $\mathcal{O}$.

6.2. Comparisons

We report comparisons on canonicalizing both full and partial shapes. Only the rotation metrics from Section 6.1 are relevant for full shapes since we assume input shapes are mean-centered without translation differences [29]. We report the TE metric for partial shape canonicalization. Outside of these metrics, we also report indirect evaluations of canonicalization [43, 46] on classification.

Canonicalization Metrics: We compare our method with baselines and other methods using our new canonicalization metrics (Section 6.1). For this experiment, we follow previous work [11] and choose 13 categories from the ShapeNet, training one model per category as well as a joint model for all categories. We choose PCA as a baseline – for each shape we compute the top-3 principal components and use this as an equivariant frame for alignment across instances.

Table 2. Partial shape canonicalization compared to PCA and Compass*, our modification of [43]. We outperform other methods by a larger margin than in the full shapes setting.

	bench	cabinet	car	cellph.	chair	couch	firearm	lamp	monitor	plane	speaker	table	water.	avg.	multi
Ground Truth Consistency (GC)↓															
PCA	**0.0916**	0.1391	0.0727	0.0879	0.1337	0.0908	0.0371	0.1985	0.0804	0.0915	0.1479	0.1087	0.1021	0.1063	0.1063
Compass*	0.1917	0.1412	0.1020	0.1066	0.1476	0.1115	0.1538	0.1735	0.1194	0.1115	0.1617	0.1709	**0.0737**	0.1358	0.1423
Ours(F+P)	0.1416	**0.1182**	**0.0356**	**0.0685**	**0.0780**	**0.0593**	**0.0300**	**0.1501**	**0.0692**	**0.0360**	**0.1469**	**0.0662**	0.0739	**0.0826**	**0.1016**
Instance-Level Consistency (IC) ↓															
PCA	**0.1033**	0.1140	0.1149	0.0828	0.1475	0.1221	0.0517	0.1571	0.0867	0.1000	0.1182	0.1401	0.0756	0.1088	0.1088
Compass*	0.1900	0.0790	0.1183	0.0911	0.1280	0.1053	0.1440	0.1000	0.0836	0.1000	0.1134	0.1080	0.0487	0.1084	0.1247
Ours(F+P)	0.1432	**0.0501**	**0.0349**	**0.0442**	**0.0622**	**0.0478**	**0.0221**	**0.0891**	**0.0442**	**0.0265**	**0.1086**	**0.0739**	**0.0469**	**0.0611**	**0.0792**
Category-Level Consistency (CC) ↓															
PCA	**0.1269**	0.1500	0.1253	0.1081	0.1636	0.1367	0.0691	0.2312	0.1178	0.1124	0.1677	0.1769	0.1078	0.1380	0.1380
Compass*	0.2118	0.1300	0.1438	0.1215	0.1612	0.1280	0.1688	**0.1990**	0.1242	0.1255	0.1760	0.1719	**0.0919**	0.1503	0.1647
Ours (F+P)	0.1695	**0.1109**	**0.0632**	**0.0739**	**0.1270**	**0.0935**	**0.0546**	0.2048	**0.1042**	**0.0713**	**0.1666**	**0.1579**	0.0936	**0.1147**	**0.1234**

We compare with two methods for rotation canonicalization: Canonical Capsules (CaCa) [46] and Compass [43].

Results for full shape canonicalization are shown in Table 1. We evaluate two versions of our method on full shapes, one trained with only full shapes (F) and one trained on both full and partial shapes (F+P). For the IC metric, both our methods outperform other methods, including baselines, in almost all categories. PCA underperforms in the IC metric due to the frame ambiguity. Our method outperforms other canonicalization methods, but surprisingly, we find that PCA is very close. For the CC metric, canonicalized shapes of different geometry are compared with each other. PCA minimizes CC metric by aligning shapes using the principal directions, but does not result in the correct canonical frame as shown in Section 6.2 (see supplement for in-depth discussion). Qualitative results in Section 6.2 show that we perform significantly better than other methods. Finally, our method outperforms other methods on the GC metric indicating that it could be used to extend the size of existing datasets (see Section 6.4).

Next, we discuss results of partial shape canonicalization shown in Table 2. Since no other method exists for partial shape canonicalization, we modified the training setting of Compass to include slicing augmentation (using $\mathcal{O}$) to operate similar to our F+P method (Compass*). The training data and occlusion function are identical for all methods. Different from full shapes, we observe that our method significantly outperforms other methods on all three metrics indicating that our method's design is suited for handling partiality. We also compute the Translation Error (TE) metric averaged over all our single category models as **0.0291** while it is **0.0326** for our multi-category model. For comparison, all our shapes lie within a unit-diagonal cuboid [4].

3D Shape Classification: We measure 3D shape classification accuracy as an indirect metric of canonicalization following [43]. We train models with un-canonicalized shapes from all 13 categories. We augment the PCA baseline, CaCa, Compass and our full shape models with PointNet [33] which performs classification on canonicalized outputs. We observe that our method (**74.6%**) outperforms other methods on classification accuracy: PCA (64.9%), CaCa (72.5%), and Compass (72.2%). Please see the supplementary document for comparison on registration.

Registration: We measure the registration accuracy of our method for categories (airplanes, chairs, multi) on full shapes in table 5. Our method does not perform well in this task as we predict a frame $E \in O(3)$ which can have reflection symmetries resulting in high RMSE, but low CD.

Table 3. **Registration** – Distance in terms of root mean-square error (RMSE) and Chamfer distance between registered and ground-truth points on the ShapeNet (core) dataset for full shapes only.

	RMSE↓			**Chamfer (CD)**↓		
Method	**Airplane**	**Chair**	**Multi**	**Airplane**	**Chair**	**Multi**
PCA	0.616	0.695	0.715	0.050	0.097	0.054
Deep Closest Points [53]	0.318	0.160	0.131	-	-	-
Deep GMR [60]	0.079	0.082	0.077	-	-	-
CaCa [46]	**0.024**	**0.027**	**0.070**	**0.009**	0.026	0.040
Compass [43]	0.361	0.369	0.487	0.061	0.079	0.051
Ours (F)	0.254	0.314	0.496	0.015	0.026	0.040
Ours (F + P)	0.201	0.280	0.404	0.014	**0.023**	**0.033**

6.3. Ablations

We justify the following key design choices: the effect of increasing amounts of occlusion/partiality, loss functions (Section 5.1), and the benefit of multiple frames.

Degree of Occlusion/Partiality: We examine the ability of our model to handle varying amounts of occlusion/partiality for the car category. Our occlusion function $\mathcal{O}$ occludes shapes to only keep a fraction of the original shape between 25% and 75% (*i.e.*, 25% is more occluded than 75%). The average over all metrics indicates that our method performs optimally when trained at 50% occlusion (25%: 0.0594, 50%: **0.0580**, 75%: 0.0886).

Loss Functions: We evaluate our F+P model on both full and partial shapes trained with all losses, without the separation loss $\mathcal{L}_{sep}$, and without the restriction loss $\mathcal{L}_{rest}$. We observe that using $\mathcal{L}_{sep}$ and $\mathcal{L}_{rest}$ performs optimally with the least average error **0.0696** across all canonicalization

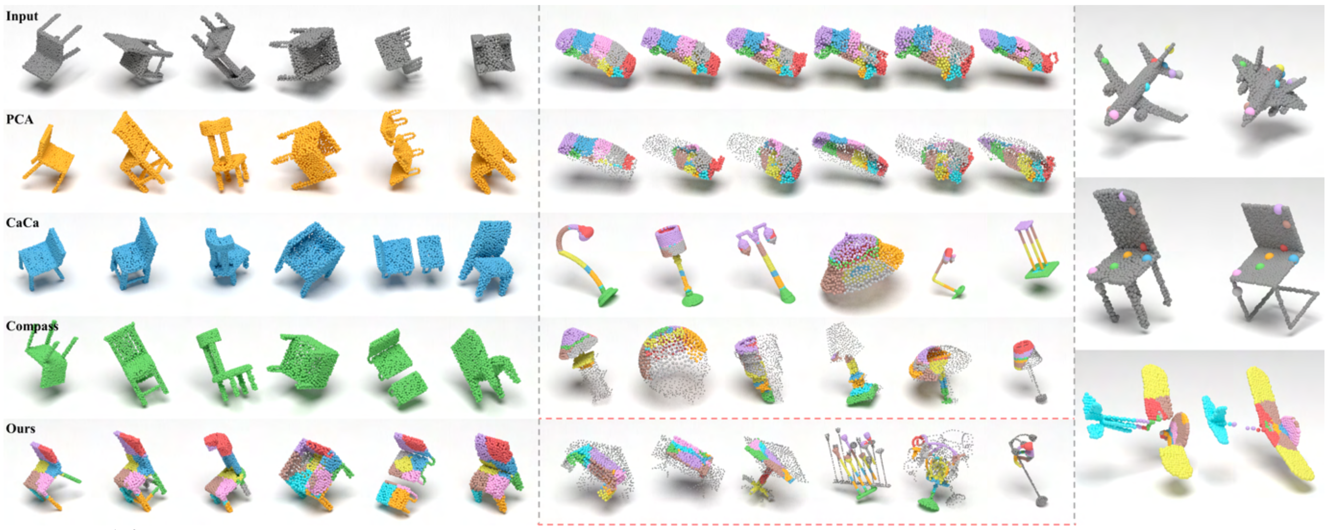

Figure 4. (***left***) Qualitative comparison with other methods on 6 *randomly chosen* full shapes. (***center***) More qualitative results from our method on challenging full/partial car shapes and a variety of full/partial lamp shapes (missing parts only shown for visualization). The last row (red border) shows failure cases caused due to incorrect canonical translation for partial shapes, or symmetric shapes. (***right***) Rows 1–2: Application of our method in transferring sparse keypoints from one shape to another. Row 3: Canonicalization of two depth maps from the ShapeNetCoco [44] dataset showing consistency in canonicalized shapes. All results rendered using Mitsuba 2 [28].

metrics over three categories (airplanes, tables, chairs).

Multi-Frame Prediction: We ablate on the number of canonical frames $(1, 3, 5)$ predicted by our method to measure its effectiveness on symmetric categories. We evaluate on two symmetric categories, table, and lamp, and observe (Table 4) that 3 and 5 frames perform better in most cases.

Table 4. Our method handles symmetric categories like lamp and table by estimating multiple canonical frames.

Category	lamp			table		
Frames	1	3	5	1	3	5
GC (full) ↓	0.1400	0.1370	**0.1274**	0.0749	**0.0693**	0.0738
IC (full) ↓	0.0686	0.0635	**0.0403**	0.0607	0.0564	**0.0361**
CC (full) ↓	0.1869	0.1887	**0.1844**	0.1595	0.1569	**0.1432**
GC (partial) ↓	0.1782	0.1711	**0.1501**	0.0681	**0.0635**	0.0662
IC (partial) ↓	0.1376	0.1319	**0.0891**	0.0923	0.0936	**0.0739**
CC (partial) ↓	0.2230	0.2226	**0.2048**	0.1705	0.1722	**0.1579**

6.4. Applications

ConDor enables applications that were previously difficult, particularly for category-level object understanding. First, since our method operates on partial shapes, we can canonicalize objects in **depth images**. To validate this, we use depth maps from the ShapeNetCOCO dataset [44] and canonicalize partial point clouds from the depth maps. Section 6.2 (right, row 3) shows an example of depth map canonicalization (see supplementary). Second, since our method outperforms other methods, we believe it can be used to expand existing canonical datasets with un-canonicalized shapes from the internet – we show examples of expanding the ShapeNet in the supplementary document. Finally, we show that ConDor can be used to transfer sparse keypoint annotations between shape instances. We utilize the unsupervised part segmentation learned using our method to solve this task (see supplementary). Section 6.2 (right, rows 1–2) shows results of transferring keypoint annotations from one shape to another.

7. Conclusion

We introduced ConDor, a self-supervised method to canonicalize the 3D pose of full and partial 3D shapes. Our method uses TFNs and self-supervision losses to learn to canonicalize pose from an un-canonicalized shape collection. Additionally, we can learn to consistently co-segment object parts without supervision. We reported detailed experiments using four new metrics, and new applications.

Limitations & Future Work: Despite the high quality of our results, we encounter failures (see Section 6.2), primarily with symmetric or objects with fine details (lamps) where the canonical frame is incorrect. We also observed that PCA often performs very well, and sometimes outperforms methods on full shapes (we do significantly better on partial shapes). Our method occasionally generates flipped canonicalized shapes along the axis of symmetry due to the prediction of an $O(3)$ frame. Our work can be extended to canonicalize purely from partial shapes and perform scale canonicalization.

Acknowledgments: This work was supported by AFOSR grant FA9550-21-1-0214, a Google Research Scholar Award, a Vannevar Bush Faculty Fellowship, ARL grant W911NF2120104, and gifts from the Adobe and Autodesk corporations. We thank the reviewers for their valuable comments.

References

[1] Unity asset store - the best assets for game making. https://assetstore.unity.com/. (Accessed on 11/08/2021). 1

[2] Brandon Anderson, Truong-Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. *arXiv preprint arXiv:1906.04015*, 2019. 3

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 5, 13

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 2, 4, 7

[5] Chao Chen, Guanbin Li, Ruijia Xu, Tianshui Chen, Meng Wang, and Liang Lin. Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4994–5002, 2019. 2

[6] Zhiqin Chen, Kangxue Yin, Matthew Fisher, Siddhartha Chaudhuri, and Hao Zhang. Bae-net: branched autoencoder for shape co-segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8490–8499, 2019. 2, 4

[7] Gregory S Chirikjian, Alexander B Kyatkin, and AC Buckingham. Engineering applications of noncommutative harmonic analysis: with emphasis on rotation and motion groups. *Appl. Mech. Rev.*, 54(6):B97–B98, 2001. 3

[8] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 1, 2

[9] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018. 2

[10] Florin Cutzu and Shimon Edelman. Canonical views in object representation and recognition. *Vision Research*, 34(22):3037–3056, 1994. 1, 2

[11] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. *arXiv preprint arXiv:1908.04725*, 2019. 1, 5, 6

[12] Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–68, 2018. 2

[13] Jin Fang, Dingfu Zhou, Xibin Song, Shengze Jin, Ruigang Yang, and Liangjun Zhang. Rotpredictor: Unsupervised canonical viewpoint learning for point cloud classification. In *2020 International Conference on 3D Vision (3DV)*, pages 987–996, 2020. 1

[14] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 1, 2

[15] Jiayuan Gu, Wei-Chiu Ma, Sivabalan Manivasagam, Wenyuan Zeng, Zihao Wang, Yuwen Xiong, Hao Su, and Raquel Urtasun. Weakly-supervised 3d shape completion in the wild, 2020. 1, 2

[16] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2

[17] Andrew E Johnson. Spin-images: a representation for 3-d surface matching. 1997. 2

[18] Cem Keskin, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun. Real time hand pose estimation using depth sensors. In *Consumer depth cameras for computer vision*, pages 119–137. Springer, 2013. 2

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[20] Anthony W Knapp. *Representation theory of semisimple groups: an overview based on examples*, volume 36. Princeton university press, 2001. 3

[21] Leon Lang and Maurice Weiler. A wigner-eckart theorem for group equivariant convolution kernels. *arXiv preprint arXiv:2010.10952*, 2020. 3

[22] Jiahui Lei, Srinath Sridhar, Paul Guerrero, Minhyuk Sung, Niloy Mitra, and Leonidas J Guibas. Pix2surf: Learning parametric 3d surface models of objects from images. In *European Conference on Computer Vision*, pages 121–138. Springer, 2020. 2

[23] Kangcheng Liu, Zhi Gao, Feng Lin, and Ben M Chen. Fg-net: Fast large-scale lidar point cloudsunderstanding network leveraging correlatedfeature mining and geometric-aware modelling. *arXiv preprint arXiv:2012.09439*, 2020. 1, 2

[24] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 2

[25] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 1, 2

[26] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 1

[27] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level

3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 909–918, 2019. 2

[28] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 38(6), Dec. 2019. 8

[29] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7688–7697, 2019. 1, 2, 3, 4, 6

[30] Stephen E Palmer. *Vision science: Photons to phenomenology*. MIT press, 1999. 2

[31] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 1, 2

[32] Adrien Poulenard and Leonidas J Guibas. A functional approach to rotation equivariant non-linearities for tensor field networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13174–13183, 2021. 2, 3, 5, 11

[33] 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. 2, 7

[34] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016. 2

[35] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, pages 5099–5108, 2017. 2

[36] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10901–10911, 2021. 1

[37] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009. 2

[38] Rahul Sajnani, AadilMehdi Sanchawala, Krishna Murthy Jatavallabhula, Srinath Sridhar, and K Madhava Krishna. Draco: Weakly supervised dense reconstruction and canonicalization of objects. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10302–10309. IEEE, 2021. 1, 2, 11

[39] Samuele Salti, Federico Tombari, and Luigi Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, 2014. 2

[40] Roger N Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971. 1, 2

[41] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304. Ieee, 2011. 2

[42] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013. 2

[43] Riccardo Spezialetti, Federico Stella, Marlon Marcon, Luciano Silva, Samuele Salti, and Luigi Di Stefano. Learning to orient surfaces by self-supervised spherical cnns. *arXiv preprint arXiv:2011.03298*, 2020. 1, 2, 5, 6, 7, 12

[44] Srinath Sridhar, Davis Rempe, Julien Valentin, Sofien Bouaziz, and Leonidas J Guibas. Multiview aggregation for learning category-specific shape reconstruction. *arXiv preprint arXiv:1907.01085*, 2019. 1, 5, 8, 11

[45] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proc. ICCV*, pages 945–953, 2015. 2

[46] Weiwei Sun, Andrea Tagliasacchi, Boyang Deng, Sara Sabour, Soroosh Yazdani, Geoffrey Hinton, and Kwang Moo Yi. Canonical capsules: Unsupervised capsules in canonical pose. *arXiv preprint arXiv:2012.04718*, 2020. 1, 2, 5, 6, 7, 12, 15

[47] Michael J Tarr and Steven Pinker. Mental rotation and orientation-dependence in shape recognition. *Cognitive psychology*, 21(2):233–282, 1989. 1, 2

[48] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019. 1, 2

[49] 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. 2, 3

[50] 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. 2

[51] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Robotics: Science and Systems*, volume 1, pages 10–15. Rome, Italy, 2015. 2

[52] 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. 1, 2

[53] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3523–3532, 2019. 2, 7, 12

[54] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2

[55] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *NIPS*, pages 10381–10392, 2018. 3

[56] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proc. CVPR*, pages 1912–1920, 2015. 1, 2, 4, 5, 13

[57] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4541–4550, 2019. 2

[58] Zhangsihao Yang, Or Litany, Tolga Birdal, Srinath Sridhar, and Leonidas Guibas. Continuous geodesic convolutions for learning on 3d shapes. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 134–144, 2021. 2

[59] Wentao Yuan, Benjamin Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. Deepgmr: Learning latent gaussian mixture models for registration. *CoRR*, abs/2008.09088, 2020. 2

[60] Wentao Yuan, Benjamin Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. Deepgmr: Learning latent gaussian mixture models for registration. In *European Conference on Computer Vision*, pages 733–750. Springer, 2020. 7, 12

[61] Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Joshua B Tenenbaum, William T Freeman, and Jiajun Wu. Learning to reconstruct shapes from unseen classes. *arXiv preprint arXiv:1812.11166*, 2018. 2

[62] Zhiyuan Zhang, Binh-Son Hua, David W Rosen, and Sai-Kit Yeung. Rotation invariant convolutions for 3d point clouds deep learning. In *2019 International Conference on 3D Vision (3DV)*, pages 204–213. IEEE, 2019. 2

Appendix

A. Network details

A.1. Architecture

We reuse the classification architecture described in Section 3.1 of [32] as our backbone. The architecture comprises of three equivariant convolution layers followed by a global max-pooling layer, and the remaining layers specialize for classification; we drop these last layers and specialize the network for our tasks instead. The global max-pooling layer of [32] proceeds by first interpreting each point-wise signal as coefficients of spherical functions in the SH basis and performing a discrete inverse spherical harmonics transform to convert them into functions over a discrete sampling of the sphere. For any direction, the resulting signal is then spatially pooled over the shape, resulting in a single function over the sphere sampling (specifically, a single map from the sphere sampling to $\mathbf{R}^C$, where $C = 256$ as we have 256 channels). We then apply point-wise MLPs (with ReLU activations) on this sphere map and convert it back to TFN-like features via forward spherical harmonics transform (SHT) [32].

Spherical Harmonic Coefficients: In order to predict the coefficients $F(X)$ of the invariant embedding $H(X)$, we apply a $[128, 64]$-MLP whose last layer is linear and convert to types $\ell \in [\![0, 3]\!]$ via SHT.

Rotation-Invariant Point Cloud: We obtain our 3D invariant point cloud X^c by applying a linear layer to $H(X)$.

Rotation-Equivariant Frame: To predict E, we apply a $[64, 3]$-MLP whose last layer has a linear activation. We then extract type 1 features with SHT, giving us a collection of 3 equivariant 3D vectors.

Segmentation: To predict the segmentation we apply a point-wise $[256, 128, 10]$-MLP whose last layer is soft-max to get the segmentation masks S described in Appendix B.

A.2. Training Details

Cropping operator $\mathcal{O}$: We introduce synthetic occlusion in our training setting by slicing full shapes using the cropping operator $\mathcal{O}$. To perform a crop, we uniformly sample a direction v on the unit sphere and remove the top $K/2$ points in the shape that have the highest value of $x^T v$ for $x \in X$. Additionally, we train our model on the ShapeNet-COCO dataset [38, 44] which has pre-determined occlusion due to camera motion, as seen in Figure 6. In order to preprocess this data for training, we aggregate the parts in the canonical NOCS space of every sequence to obtain the full shape and perform a nearest neighbor search in the NOCS space to find correspondences between the full and partial shape.

Hyper-parameters: During training, we use a batch size of 16 in every step for all our models. We set an L^1 kernel regularizer at every layer of the network with weight 0.1. We weigh the loss functions by their effect on reducing the Canonical Shape loss $\mathcal{L}_{canon}$. The loss functions are weighed as: $\mathcal{L}_{canon}$ (2), $\mathcal{L}_{rest}$ (1), $\mathcal{L}_{ortho}$ (1), $\mathcal{L}_{sep}$ (0.8), and $\mathcal{L}_{amod}$ (1).

B. Unsupervised Co-segmentation

B.1. Predicting parts

We predict the part segments $S \in \mathbb{R}^{K \times C}$ wherein C are the number of parts. We use the rotation-invariant embedding $H^\ell(X)$ with all the types $0 \leq \ell \leq 3$ to predict the segmentation S. We define the following notation for normalized parts $A(X)$ and part centroids $\theta(X)$ similar to [46].:

$$\begin{aligned} S(X) &:= \text{Softmax}[\text{MLP}(H(X))] \\ A_{ij}(X) &:= \frac{S_{ij}(X)}{\sum_i S_{ij}(X)} \\ \theta_j(X) &:= \sum_i A_{ij}(X) X_{i,:} \end{aligned} \tag{8}$$

B.2. Loss functions

We use part segmentation to enforce semantic consistency between full and partial shapes. We borrow the localization loss ($\mathcal{L}_{localization}$) and equilibrium loss ($\mathcal{L}_{equilibrium}$) from [46] for the full shape to evenly spread part segmentation across the shape. Additionally, we employ the following losses.

Part Distribution loss: We compute the two-way Chamfer distance (CD) between the part centroids and the input shape. In practice, this helps to distribute parts more evenly across the shape.

$$\mathcal{L}_{dist} = \text{CD}\left(X, \theta(X)\right) \tag{9}$$

Part Restriction loss: The parts discovered by the network for the partial shape should be congruent to the parts discovered by the network for the full shape. We penalize the part prediction for corresponding parts by minimizing the negative Cosine Similarity (CS) for our capsule predictions.

$$\mathcal{L}_{rest(part)} = -\frac{2}{K} \sum_{i \in S} \mathbf{CS}(S(\mathcal{O}(X))_{i,:}, \mathcal{O}(S(X))_{i,:}) \tag{10}$$

Part Directional loss: To avoid part centers of the visible parts of a shape from deviating from the part centers of the full shape, we use a soft loss to ensure that the directional vector between part centers are consistent between the full and partial shape. $\text{dir}(\theta(X))$ computes the vector directions between every ${}^C C_2$ centroid pairs for C part centroids.

$$\mathcal{L}_{direc} = -\frac{1}{{}^C C_2} \sum_{i \in S} \mathbf{CS}(\text{dir}(\theta(\mathcal{O}(X_i))), \text{dir}(\mathcal{O}(\theta(X_i)))) \tag{11}$$

C. Registration

Table 5. **Registration** – Distance in terms of root mean-square error (RMSE) and Chamfer distance between registered and ground-truth points on the ShapeNet (core) dataset for full shapes only.

	RMSE↓			**Chamfer (CD)↓**		
Method	**Airplane**	**Chair**	**Multi**	**Airplane**	**Chair**	**Multi**
PCA	0.616	0.695	0.715	0.050	0.097	0.054
Deep Closest Points [53]	0.318	0.160	0.131	-	-	-
Deep GMR [60]	0.079	0.082	0.077	-	-	-
CaCa [46]	**0.024**	**0.027**	**0.070**	**0.009**	0.026	0.040
Compass [43]	0.361	0.369	0.487	0.061	0.079	0.051
Ours (F)	0.254	0.314	0.496	0.015	0.026	0.040
Ours (F + P)	0.201	0.280	0.404	0.014	**0.023**	**0.033**

We note in Table 5 that our method does not perform well in this task as we predict a frame $E \in O(3)$ which can have reflection symmetries, we observe symmetries such as left-right reflection for planes. Symmetries cause high RMSE error because points are matched with their image under symmetry which are often very distant. However, when using Chamfer Distance metric which is symmetry agnostic our registration error decreases by an order of magnitude achieving competitive results on this benchmark. We also note that Ours(F+P) noticeably decreases RMSE compared to Ours(F) as during training the frame consistency is enforced between the full shape and a randomly rotated partial by the $\mathcal{L}_{rest}$ loss.

D. Ablations

We now provide detailed ablations to justify the following key design choices: the effect of increasing amounts of occlusion/partiality, and loss functions.

Degree of Occlusion/Partiality: We examine the ability of our model to handle varying amounts of occlusion/partiality for the car category in Table 6. Our occlusion function, $\mathcal{O}$, occludes shapes to only keep a fraction of the original shape between 25% and 75% (*i.e.*, 75% is more occluded than 25%). We observe that our method performs optimally over all metrics when trained at 50% occlusion.

Loss Functions: We evaluate our F+P model on both full and partial shapes trained with all losses, without the separation loss $\mathcal{L}_{sep}$, and without the restriction loss $\mathcal{L}_{rest}$. From Table 7, we observe that using restriction loss $\mathcal{L}_{rest}$ helps in canonicalization of both full and partial shapes in categories *plane*, *table*, and *chair*. However, separation loss, $\mathcal{L}_{sep}$, helps in *plane*, *table* but not in *chair*. Since, both losses help in most of the categories, we utilize them for training our final model.

Effect of introducing occlusion on full shapes: We evaluate the canonicalization of full shapes using our network trained on full and partial shapes. We observe that on average both our models **Ours(F)** and **Ours(F+P)** perform the

Test partiality	**Degree of partiality during training**			
	75%	50%	25%	[25%, 75%]
Ground Truth Consistency (GC)↓				
75%	0.0451	**0.0438**	0.1420	0.0681
50%	0.0375	0.0356	0.0504	**0.0296**
25%	0.0388	0.0301	**0.0241**	0.0299
[25%, 75%]	**0.0438**	0.0553	0.0894	0.0558
Instance-Level Consistency (IC)↓				
75%	0.0728	**0.0719**	0.1542	0.0797
50%	0.0452	**0.0349**	0.0526	0.0380
25%	0.0456	0.0333	**0.0221**	0.0334
[25%, 75%]	**0.0719**	0.0792	0.1049	0.0804
Category-Level Consistency (CC)↓				
75%	0.0914	**0.0895**	0.1702	0.0966
50%	0.0652	0.0632	0.0731	**0.0617**
25%	0.0657	0.0608	**0.0582**	0.0606
[25%, 75%]	**0.0895**	0.0985	0.1216	0.0982
Average	0.0594	**0.0580**	0.0886	0.0610

Table 6. **Degree of partiality** - Partiality introduced during training (vertical) is evaluated on the canonicalization metrics with different fraction of partiality (horizontal). [25%, 75%] indicates that degrees of partiality between 25% and 75% are randomly introduced in the shapes. Our model trained with partiality 50% performs better on average over all the canonicalization metrics. [*Note: 75% is more occluded than 25%.*]

same on the canonicalization metrics for full shapes. For a few categories such as *lamp, car, chair, watercraft*, introducing partial shapes in the training improves its performance on the canonicalization metrics. Whereas introducing occlusion during training degrades the performance for category *bench*.

E. Applications

E.1. Co-Canonicalization

Commonly used datasets in 3D vision, such as ShapeNet [3], are manually pre-canonicalized, making expansion of such datasets expensive. Since our method performs better than others on canonicalization, we believe that it can be used to extend these datasets by canonicalizing corpora of *in-the-wild* shapes into a common pose. Figure 5 shows the results of our model, trained on the ShapeNet (core) dataset [3], being used to canonicalize shapes from the (uncanonicalized) ModelNet40 dataset [56]. These shapes can now be merged into ShapeNet by applying a single category-wide rotation to match the obtained canonical frame with the existing frame used by ShapeNet, instead of the per-instance rotation that would otherwise be required. Furthermore, these results qualitatively demonstrate the ability of our method to generalize to datasets not seen during training.

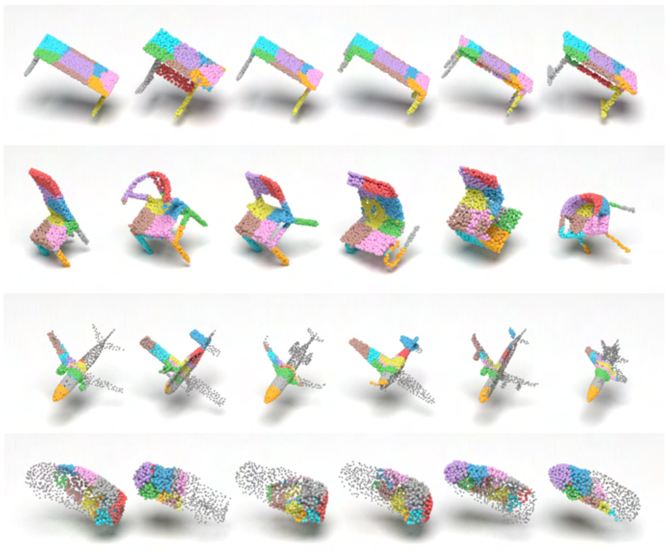

Figure 5. Co-canonicalizing object instances from ModelNet40 using our method trained on ShapeNet (core). (***top***) Canonicalized full shapes. (***bottom***) Canonicalized partial shapes.

E.2. Depth Map Canonicalization

Since our method operates on partial shapes, we can canonicalize objects in **depth images**. We use the depth maps from the ShapeNetCOCO dataset, which have predetermined occlusion due to camera motion, and canonicalize partial point clouds. Specifically, we first take depth maps and utilize them to generate groundtruth pointclouds. We then trained and tested our model on it. Figure 6 present examples to demonstrate that our model is capable of canonicalizing depth maps.

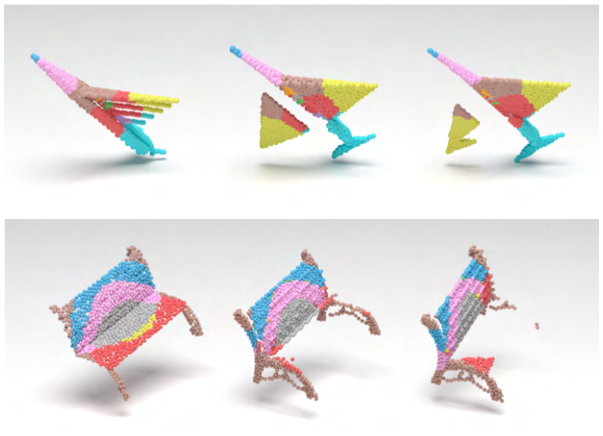

Figure 6. Canonicalizing point clouds obtained from depth maps from the ShapeNetCOCO dataset.

Category !	**Plane**			**Table**			**Chair**			**Average**		
Metric #	**Ours**	**w/o sep**	**w/o rest**	**Ours**	**w/o sep**	**w/o rest**	**Ours**	**w/o sep**	**w/o rest**	**Ours**	**w/o sep**	**w/o rest**
GC (full)	**0.0286**	0.0321	0.0303	0.0738	**0.0641**	0.0729	0.0509	**0.0430**	0.0532	0.0511	**0.0464**	0.0521
IC (full)	**0.0144**	0.0187	0.0169	**0.0361**	0.0612	0.0411	0.0235	**0.0224**	0.0245	**0.0247**	0.0341	0.0275
CC (full)	**0.0679**	0.0697	0.0683	**0.1432**	0.1510	0.1434	0.1145	0.1150	**0.1143**	**0.1085**	0.1119	0.1087
GC (partial)	0.0360	0.0389	**0.0332**	0.0662	**0.0523**	0.0683	0.0780	**0.0681**	0.0850	0.0601	**0.0531**	0.0622
IC (partial)	**0.0265**	0.0324	0.0479	**0.0739**	0.0791	0.0805	0.0622	**0.0537**	0.0841	**0.0542**	0.0551	0.0708
CC (partial)	**0.0713**	0.0733	0.0765	**0.1579**	0.1590	0.1598	0.1270	**0.1250**	0.1377	**0.1187**	0.1191	0.1247
Average	**0.0408**	0.0442	0.0455	**0.0912**	0.0945	0.0943	0.0760	**0.0712**	0.0831	**0.0696**	0.0700	0.0743

Table 7. Ablation study to investigate the effect of different loss functions. "w/o sep" and "w/o rest" denote training without separation and without restriction loss, respectively.

E.3. Annotation Transfer

Since a category-level canonical frame is consistent with respect to the geometry and local shape of different object instances of a category, annotations can be transferred across instances that share the same canonical frame. Particularly, we demonstrate the transfer of sparse key-point annotations in Figure 7. We randomly assign labels to a few points of one point cloud in each category, which serves as the source. We then use a remarkably simple transfer function to transfer these labels to points in each target point cloud, making use of the predicted segmentation. To every labeled point in the source point cloud, we obtain a directional vector originating from the centroid of the segment it belongs to. Starting from the corresponding centroid in the target point cloud, we move along this directional vector and then pick the nearest point. While this scheme works well in our case, more nuanced transfer functions may be required depending on the application.

F. Proof of Rotation-Invariance Property of our Embedding

Given rotation-equivariant embeddings F^ℓ and Y^ℓ the tensors $H^\ell(X)$ are rotation invariant as:

$$\begin{aligned} H^\ell_{ijk}(R.X) &= \langle F^\ell_{i,:,j}(R.X), Y^\ell_{:,j,k}(R.X)\rangle \\ &= \langle D^\ell(R)F^\ell_{i,:,j}(X), D^\ell(R)Y^\ell_{:,j,k}(X)\rangle \\ &= \langle F^\ell_{i,:,j}(X), Y^\ell_{:,j,k}(X)\rangle = H^\ell_{ijk}(X) \end{aligned}$$

G. Commutative Property of Canonicalization with the Cropping Operator

Canonicalization commutes with the cropping operator $\mathcal{O}$. For a (full) point cloud X and predicted canonicalizing frame $\mathcal{R}(X)$, we prove the commutative property here, we assume X is mean centered for simplification.

$$\begin{aligned} &\mathcal{O}[X]^c + \mathcal{R}(X)\overline{\mathcal{O}[X]} = \mathcal{R}(X)(\mathcal{O}[X] + \overline{\mathcal{O}[X]}) \\ &= \mathcal{R}(X)(\mathcal{O}[X]) = \mathcal{O}[\mathcal{R}(X)X] = \mathcal{O}[X^c] \end{aligned}$$

Figure 7. Transferring key-point annotations from one shape to another in the same category. We annotate only the first column of shapes and transfer key-points to all the other columns

The above commutative property enables us to a predict a rotation-equivariant translation $\mathcal{T}(\mathcal{O}(X))$ from the mean centered partial shape $\mathcal{O}(X)$ only that aligns the partial shape to its corresponding points in the full shape.

$$\begin{aligned} &\mathcal{O}[X]^c + \mathcal{R}(\mathcal{O}[X])\overline{\mathcal{O}[X]} \simeq \mathcal{O}[X]^c + \mathcal{R}(\mathcal{O}[X])\mathcal{T}(\mathcal{O}(X)) \\ &= \mathcal{O}[X^c] \end{aligned}$$

H. Discussion on Canonicalization Metrics

We complement the discussion of our canonicalization metrics with a few remarks. Our 3 metrics Instance-Level

(IC), Category-Level (CC) and Ground Truth (GC) Consistency measure three aspects of canonicalization. The instance-level metric is a measure of the ”variance” of the canonical pose under rotation of the input. By definition the canonical pose must be invariant to the input pose. The GC metric provides a way of measuring canonicalization consistency across the entire class of objects by measuring how our canonicalization deviates from a ground truth canonicalization up to a constant rotation. In the absence of a ground truth alignment, we propose the CC metric which compares canonicalization of different shapes within the same class using Chamfer distance (as we don’t assume pointwise correspondences between different shapes). The CC metric relies on the assumption that aligned shapes of the same category are similar to each other.

We observe in table (1) of our article that some methods have high IC but low GC and vice versa (e.g. CaCa [46] (cabinet), Ours (F + P) speaker). This occurs as we canonicalize based on geometric similarity instead of semantic aspects of the object. The IC and CC metrics measure geometric properties of the canonicalization while GC measures semantic properties of the canonicalization according to manually aligned shapes.

We build our metrics using the Chamfer distance as it does not assume pointwise correspondences between shapes, this allows measuring the canonicalization quality of symmetric shapes where there may not be a single correct canonical orientation. However, we observe a performance gap with our method when using distances based on pointwise correspondences such as L^2 or root mean square (RMSE) errors as seen in Appendix C of this appendix. We believe our Chamfer distance based metrics are representative of the quality of canonicalization and are consistent with our visual evaluation.

I. Discussion on PCA

PCA Over-Performance on the CC Metric: We note that the competitiveness of PCA is limited to certain experiments for **full shapes** and **multi-category** experiments only. The CC metric compares canonicalized shapes of the same category with possibly different geometry – note that PCA even outperforms ground truth canonicalization for this metric. Thus a method which is optimal for GC metric cannot outperform PCA in CC.

PCA Under-Performance on the IC Metric: The most likely reason why PCA underperforms on the IC metric is because of frame ambiguity. The PCA principal directions are defined up to symmetries of the covariance matrix eigenspaces – the shape does not necessarily share these symmetries. For instance, when eigenvalues are distinct, eigenvectors are defined up to sign, causing random flips over principal directions: *e.g.*, an airplane can be flipped on its back. When two or more eigenvalues are identical, eigenvectors are defined up to rotation, *e.g.*, in chairs, the major component can be from the left leg to the top right corner or bottom right leg to top left corner. Thus, PCA canonicalization of rotated copies of a given shape may not be equal due to symmetries of the shape, resulting in higher Chamfer/IC error.

J. Qualitative Results

We now present more qualitative results in Figure 8, 9 to demonstrate the effectiveness of our method.

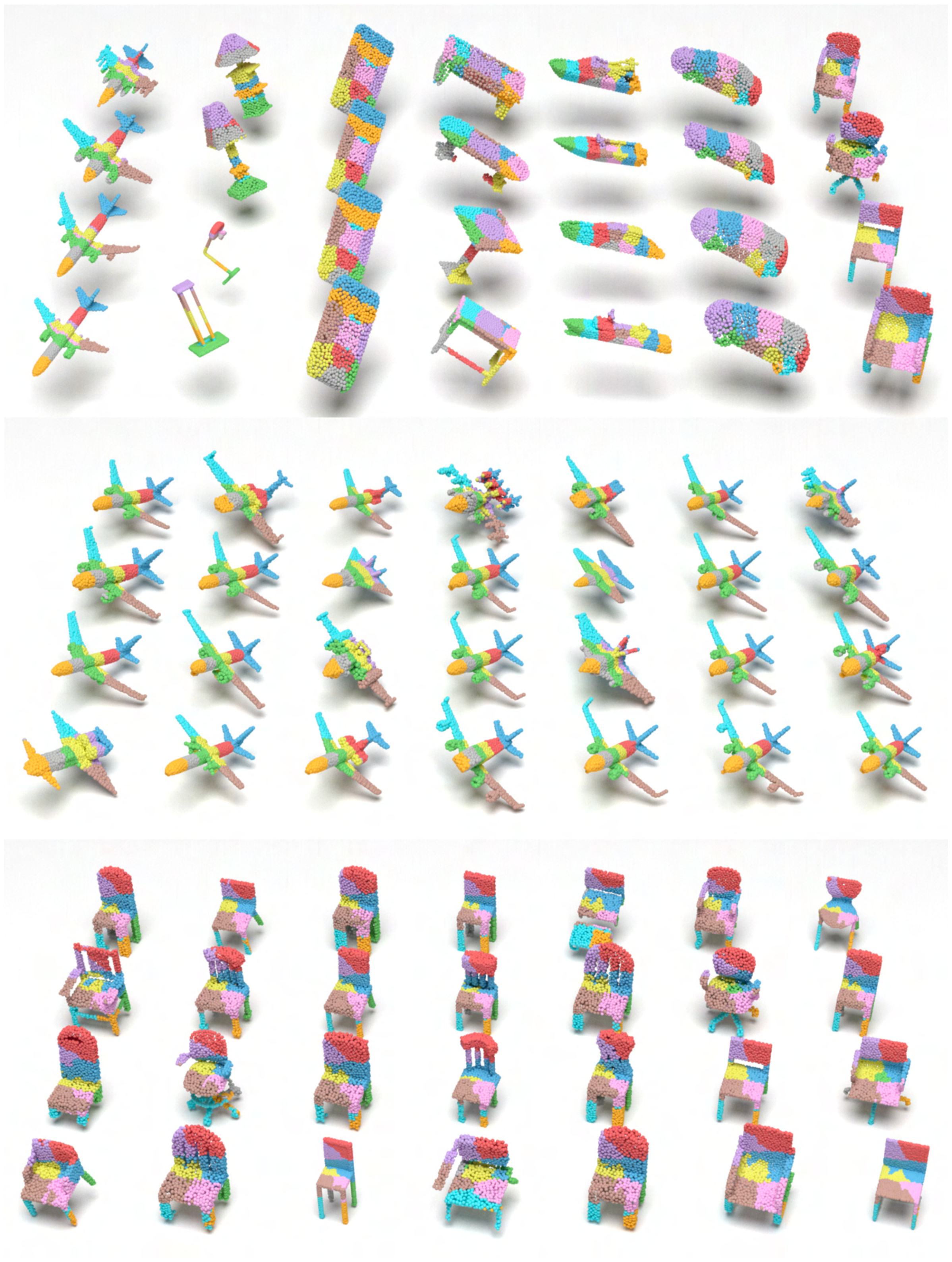
Figure 8. Parking lot for full shape canonicalization for multi-category(***top***), plane (***middle***) and chair (***bottom***).

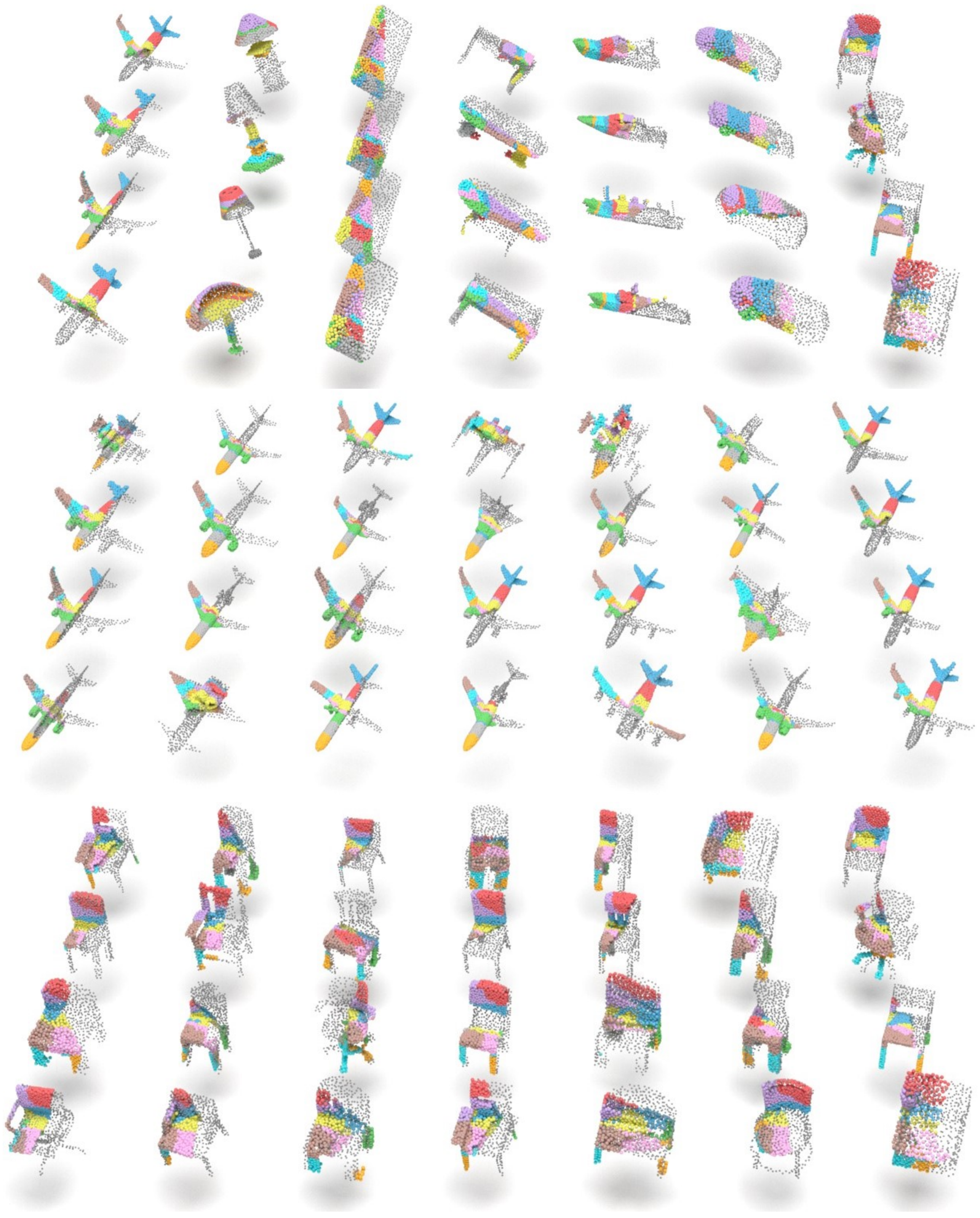

Figure 9. Parking lot for partial shape canonicalization for multi-category(***top***), plane (***middle***) and chair (***bottom***). Note: missing parts only shown for visualization.