3D Equivariant Graph Implicit Functions: Appendices

A Equivariance: Discussions and Proofs

A.1 From layer equivariance to model equivariance

We review the definition of the equivariance of the implicit function model in Section 3. In Eq. (3), we show that equivariance is satisfied if the implicit function is locally invariant to any T_g applied jointly to the observation $\mathbf{X}$ and the query $\mathbf{p}$, for any $\mathbf{X}$ and $\mathbf{p}$. Invariance is a special case of equivariance, where the transformation T_g^* on the output domain is the identity function, i.e., the output is invariant regardless of the input transformation T_g.

Here we clarify the use of layer equivariance in creating the equivariant implicit function model. As in Eq. (4), the graph implicit function model F_{graph} is composed of the graph latent feature extractor Φ_{graph} and the implicit decoder Ψ, where Φ_{graph} can be further decomposed to the point encoder ϕ and the graph local latent aggregator ψ. We integrate the equivariant layers in ϕ and ψ that process the input point set $\mathbf{X}$ and the queries $\mathbf{p}$. As shown by the literature, the composition of two equivariant functions is also an equivariant function [46]. Thus, the graph local feature extractor Φ_{graph} that stacks sequential equivariant graph layers in ϕ and ψ is equivariant. Note that all the operations other than the graph convolutions involved in ϕ and ψ, such as the k-NN graph extration and the farthest-point-sampling for the multi-scale feature, are equivariant to the similarity transformations as well. At the end of all the equivariant graph layers in ϕ and ψ, we apply the invariance function Ω to obtain the locally invariant latent feature. The implicit decoder Ψ, which processes the invariant local latent feature and predicts the occupancy probability, is simply a standard ReLU-MLP as the non-equivariant implicit models. We do not include the query coordinate input $\mathbf{p}$ in the implicit decoder Ψ in order to satisfy translation equivariance, while the local latent feature already contains the position information. Since the local latent feature is locally invariant, the output is locally invariant as well, which satisfies Eq. (3). Thus far, we have shown how the equivariant graph layers help to build the equivariant implicit function model.

A.2 Extension to translation and scale equivariance

While existing vector-based equivariance methods in 3D vision [17,40] apply only to the $\mathrm{SO}(3)$ group[5], we extend our method to be equivariant to the similarity transformation group that further includes translation and scale transformations as subgroups.

Translation. The local graph structure is robust to rotation by design. The method can further achieve numerically guaranteed translation equivariance simply by removing the absolute coordinates input $\mathbf{p}$ from the graph layers in Eq. (5), keeping only the relative positions as the spatial cue.

[5] More generally, it is the $\mathrm{O}(3)$ group. Reflection is handled as well.

Scale. As vectors hold scale information from their norms, we extend the method for scale equivariance by modifying to normalize the invariance function $\Omega(\mathbf{V}) \leftarrow \Omega(\mathbf{V})/\|\Omega(\mathbf{V})\|$ based on Eq. (9). Likewise, in each layer the scalar features are scale invariant and the vectors are scale equivariant.

A.3 Proof of translation equivariance

We discuss the translation equivariance property in separate from other transformation groups. Because the translation equivariance property in our method is from the use of the local graph structure, while the equivariant properties of other similarity transformations, including rotations, reflections and scaling, rely on the hybrid features, especially the vector part. Each of the graph layers are locally invariant to the continuous translation group.

For any input points and queries, we discard the absolute global coordinate inputs, and manipulate on the relative positions within a local graph structure in all graph layers in ϕ and ψ. We consider an edge connects an input point $\mathbf{x}_{i'}$ and a query $\mathbf{p}$ in an equivariant graph layer in the graph latent aggregator ψ, then the input vector feature is $\mathbf{x}_{i'} - \mathbf{p}$. if a translation vector $\mathbf{t}$ is applied on all the points, then the input feature becomes

$$(\mathbf{x}_{i'} - \mathbf{t}) - (\mathbf{p} - \mathbf{t}) = \mathbf{x}_{i'} - \mathbf{p}. \tag{10}$$

Thus, the input is invariant to the translation $\mathbf{t}$, so is the output of the graph layer. And similarly for the graph layers in the point encoder ϕ. Therefore, the whole graph function is translation-invariant for local predictions from any 3D point inputs, which means that our graph implicit function, local to each of the query locations, is translation equivariant.

A.4 Proof of rotation, scaling and reflection equivariance

Next, we show the equivariance properties on other transformations including rotations, reflections and scaling. We assume that the scalar feature $\mathbf{h}$ is invariant to these transformations, while the vector feature $\mathbf{V}$ is equivariant, before the invariance function Ω is applied. Formally, we consider an arbitrary orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{3\times3}$ encoding a 3D rotation with a possible reflection, and an arbitrary positive scalar $m \in \mathbb{R}^+$ for the scale transformation applied on the features. The scalar and vector features $\mathbf{h}$ and $\mathbf{V}$ are then transformed into $s\mathbf{V}\mathbf{Q}^\top$ and $\mathbf{h}$ respectively. Note here the orthogonal matrix applying to a stack of vector features $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{C_\mathbf{v}}]^\top \in \mathbb{R}^{C_\mathbf{v}\times3}$ returns $(\mathbf{Q}\mathbf{V}^\top)^\top = \mathbf{V}\mathbf{Q}^\top$. We study how the output of each layer changes with the change of the inputs.

Invariance layer We first show that the invariance function works for any 3×3 orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{3\times3}$ encoding 3D rotation and reflection and any random scaling factor

$s \in \mathbb{R}$, such that $\Omega(s\mathbf{V}\mathbf{Q}^\top) = \Omega(\mathbf{V})$:

$$\begin{aligned}
\Omega(m\mathbf{V}\mathbf{Q}^\top) &= \frac{\left\langle m\mathbf{V}\mathbf{Q}^\top, \frac{m\mathbf{Q}\overline{\mathbf{v}}}{\|m\mathbf{Q}\overline{\mathbf{v}}\|} \right\rangle}{\left\| \left\langle s\mathbf{V}\mathbf{Q}^\top, \frac{m\mathbf{Q}\overline{\mathbf{v}}}{\|m\mathbf{Q}\overline{\mathbf{v}}\|} \right\rangle \right\|} = \frac{m\left\langle \mathbf{V}\mathbf{Q}^\top, \frac{\mathbf{Q}\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle}{m\left\| \left\langle \mathbf{V}\mathbf{Q}^\top, \frac{\mathbf{Q}\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle \right\|} \\
&= \frac{\left\langle \mathbf{Q}^{-1}\mathbf{Q}\mathbf{V}, \frac{\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle}{\left\| \left\langle \mathbf{Q}^{-1}\mathbf{Q}\mathbf{V}, \frac{\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle \right\|} = \frac{\left\langle \mathbf{I}\mathbf{V}, \frac{\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle}{\left\| \left\langle \mathbf{I}\mathbf{V}, \frac{\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle \right\|} \\
&= \frac{\left\langle \mathbf{V}, \frac{\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle}{\left\| \left\langle \mathbf{V}, \frac{\overline{\mathbf{v}}}{\|\overline{\mathbf{v}}\|} \right\rangle \right\|} \\
&= \Omega(\mathbf{V}),
\end{aligned} \tag{11}$$

where in the second row of Eq. (11), the orthogonal matrices $\mathbf{Q}$ are cancelled out in the inner product. Here we adopt a slightly abused notation to have the inner product between a stack of vector features $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{C_\mathbf{v}}]^\top$ and the average vector $\overline{\mathbf{v}}$, such that $\langle \mathbf{V}, \overline{\mathbf{v}} \rangle = [\langle \mathbf{v}_1, \overline{\mathbf{v}} \rangle, \langle \mathbf{v}_2, \overline{\mathbf{v}} \rangle, \ldots, \langle \mathbf{v}_{C_\mathbf{v}}, \overline{\mathbf{v}} \rangle]^\top$.

Linear layer Next, we show that our hybrid feature linear layer is equivariant to the rotation, reflection, and scaling transformations, encoded by arbitrary $\mathbf{Q}$ and m. Without these transformations, we consider $\{\mathbf{s}', \mathbf{V}'\}$ as the outputs for $\{\mathbf{s}, \mathbf{V}\}$ from the layer denoted by f, i.e., $\{\mathbf{s}', \mathbf{V}'\} = f(\{\mathbf{s}, \mathbf{V}\})$; while under the transformations encoded by $\mathbf{Q}$ and m, we denote the outputs as $\{\mathbf{s}'', \mathbf{V}''\} = f\left(\{\mathbf{s}, m\mathbf{V}\mathbf{Q}^\top\}\right)$. For the equivariance of f, we need to show that:

$$\mathbf{s}'' = \mathbf{s}', \text{ and } \mathbf{V}'' = m\mathbf{V}'\mathbf{Q}^\top, \tag{12}$$

in which we assume that the vector feature $\mathbf{V}$ is equivariant for rotations, reflections and scaling, while the scalar feature $\mathbf{h}$ is invariant to these transformations.

For the scalar feature output in Eq. (6), one can simply verify the invariance

$$\begin{aligned}
\mathbf{s}'' &= \mathbf{W}_s\mathbf{s} + \mathbf{W}_{\mathbf{v}s}\,\Omega(m\mathbf{V}\mathbf{Q}^\top) \\
&= \mathbf{W}_s\mathbf{s} + \mathbf{W}_{\mathbf{v}s}\,\Omega(\mathbf{V}) \\
&= \mathbf{s}'.
\end{aligned} \tag{13}$$

Here the invariance function returns

$$\Omega(m\mathbf{V}\mathbf{Q}^\top) = \Omega(\mathbf{V}), \tag{14}$$

as shown in Eq. (11). For the vector feature output in Eq. (6),

$$\begin{aligned}
\mathbf{V}'' &= \mathbf{W}_\mathbf{v}(m\mathbf{V}\mathbf{Q}^\top) \odot (\mathbf{W}_{s\mathbf{v}}\,\mathbf{s} \,/\, \|\mathbf{W}_{s\mathbf{v}}\,\mathbf{s}\|) \\
&= m\,(\mathbf{W}_\mathbf{v}\mathbf{V} \odot (\mathbf{W}_{s\mathbf{v}}\,\mathbf{s} \,/\, \|\mathbf{W}_{s\mathbf{v}}\,\mathbf{s}\|))\,\mathbf{Q}^\top \\
&= m\mathbf{V}'\mathbf{Q}^\top
\end{aligned} \tag{15}$$

Thus far we have shown the equivariance of the linear layers.

Non-linearity For the non-linearity, the scalar features take simple ReLU activation, hence the invariance is easily ensured as no equivariant vector feature is involved for the output scalar feature.

For the vector non-linearity $\mathbf{v}$-ReLU in Eq. (8), we first reason that the transformations $\mathbf{Q}$ and m does not influence whether the vector feature $\mathbf{v}_c$ at each channel c falls in the positive or the negative part of the piecewise non-linearity. With the untransformed feature $\mathbf{v}_c$, the positive case is judged by $\left\langle \mathbf{v}_c, \frac{\mathbf{q}}{\|\mathbf{q}\|} \right\rangle \geq 0$; while with the transformed feature vector $m\mathbf{Q}\mathbf{v}_c$, the learned direction vector $\mathbf{q}$ is transformed accordingly into $m\mathbf{Q}\mathbf{q}$. Then, the condition of the positive case becomes $\left\langle m\mathbf{Q}\mathbf{v}_c, \frac{m\mathbf{Q}\mathbf{q}}{\|m\mathbf{Q}\mathbf{q}\|} \right\rangle \geq 0$, which is equivalent to the condition without the transformations, as the orthogonal matrices $\mathbf{Q}$ are cancelled out and the positive scaling factor s does not change the sign. The same for the negative case.

Next, we show the equivariance in both positive and negative cases of the non-linearity in Eq. (8). When transformations $\mathbf{Q}$ and m are applied, in the positive case,

$$\begin{aligned} [\mathbf{v}\text{-ReLU}(m\mathbf{V}\mathbf{Q}^\top)]_c &= m\mathbf{Q}\mathbf{v}_c \\ &= m\mathbf{Q}[\mathbf{v}\text{-ReLU}(\mathbf{V})]_c; \end{aligned} \tag{16}$$

while in the negative case,

$$\begin{aligned} [\mathbf{v}\text{-ReLU}(m\mathbf{V}\mathbf{Q}^\top)]_c &= m\mathbf{Q}\mathbf{v}_c - \left\langle m\mathbf{Q}\mathbf{v}_c, \frac{m\mathbf{Q}\mathbf{q}}{\|m\mathbf{Q}\mathbf{q}\|} \right\rangle \frac{m\mathbf{Q}\mathbf{q}}{\|m\mathbf{Q}\mathbf{q}\|} \\ &= m\mathbf{Q}\left(\mathbf{v}_c - \left\langle \mathbf{v}_c, \frac{\mathbf{q}}{\|\mathbf{q}\|} \right\rangle \frac{\mathbf{q}}{\|\mathbf{q}\|}\right) \\ &= m\mathbf{Q}[\mathbf{v}\text{-ReLU}(\mathbf{V})]_c \end{aligned} \tag{17}$$

Thus,

$$\mathbf{v}\text{-ReLU}(m\mathbf{V}\mathbf{Q}^\top) = m\left(\mathbf{v}\text{-ReLU}(\mathbf{V})\right)\mathbf{Q}^\top, \quad \text{or} \tag{18}$$

$$\mathbf{V}'' = m\mathbf{V}'\mathbf{Q}^\top \tag{19}$$

has been proven in both the positive and the negative cases of the vector ReLU function, indicating the equivariance of the non-linear layer with regard to rotation, reflection and scaling.

B Detailed formulations of graph fuctions in multiple scales

We provide the detailed formulation of the layers in the multi-scale graph point encoder ϕ and the multi-scale graph latent feature decoder ψ in Sec 4.2. Here we show the formulations with only the scalar features for the non-equivariant graph implicit model. All these formulations can be easily adapted to the equivariant model by replacing the hidden features $\mathbf{h}$ to the hybrid features of both scalars and vectors $\{\mathbf{s}, \mathbf{V}\}$.

Graph point encoder. The point encoder ϕ is composed of graph convolution layers in the downsampling stage, starting from $l = 0$ to $l = L$, followed by the upsampling layers from $l = L - 1$ back to $l = 0$.

In each graph convolution layer with the sampled point set $\mathbf{X}^{(l)}$ by farthest-point-sampling in the downsampling stage, we obtain the hidden feature $\mathbf{h}_i^{(l)}$ for any sampled point at this level $\mathbf{x}_i \in \mathbf{X}^{(l)}$. To do this, we use graph convolution to aggregate information from the k-nearest neighbor points of $\mathbf{x}_i$, denoted as $\mathbf{x}_{i'}$. The information from the neighboring points to be aggregated is the hidden feature from the previous layer $\mathbf{h}_{i'}^{(l-1)}$. Within the local k-NN graph structure, messages are passed through $\eta_{\downarrow}^{(l)}$, hence concatenating the inputs for a shared two-layer ReLU-MLP, and a permutation-invariant aggregation function AGGRE; e.g., max- or mean-pooling operator:

$$\mathbf{h}_i^{(l)} = \underset{i'}{\text{AGGRE}}\ \eta_{\downarrow}^{(l)}(\mathbf{h}_i^{(l-1)}, \mathbf{h}_{i'}^{(l-1)}, \mathbf{x}_{i'} - \mathbf{x}_i). \tag{20}$$

Note that there is an exception in Eq. 20 with $l = 0$, where the input features are the raw coordinates.

In each upsampling layer, the point feature $\mathbf{h}_i^{(l)}$ for any $\mathbf{x}_i \in \mathbf{X}^{(l)}$ at a finer sampling level l takes information from $\mathbf{h}_{i'}^{(l+1)}$, the hidden feature associated to $\mathbf{x}_i$'s 1-nearest neighbor point $\mathbf{x}_{i'}$ from the sampled point set $\mathbf{X}^{(l+1)}$ from the previous sampling level. In addition, we skip-connect $\mathbf{h}_i^{(l)}$, the feature at the same level from the downsampling stage, which is akin to the U-Net structure in grid-based methods:

$$\mathbf{h}_i^{(l)} \leftarrow \eta_{\uparrow}^{(l)}(\mathbf{h}_{i'}^{(l+1)}, \mathbf{h}_i^{(l)}), \tag{21}$$

where $\eta_{\uparrow}^{(l)}$ is a linear layer with a ReLU activation for the concatenation of inputs.

Graph local latent feature aggregator. Given the query coordinate $\mathbf{p}$, we use graph convolutions to aggregate the k-neighboring features $\{(\mathbf{x}_{i'}, \mathbf{h}_{i'}^{(l)})\}$ at different sampling levels l. The aggregated features $\mathbf{z}_{\mathbf{p}}^{(l)}$ from all sampling levels l are concatenated to yield the local latent vector $\mathbf{z}_{\mathbf{p}}$ as output:

$$\mathbf{z}_{\mathbf{p}} = \mathop{\Big\|}_{l=0}^{L} \mathbf{z}_{\mathbf{p}}^{(l)}, \quad \text{where} \tag{22}$$

$$\mathbf{z}_{\mathbf{p}}^{(l)} = \underset{i'}{\text{AGGRE}}\ \eta^{(l)}(\mathbf{p}, \mathbf{h}_{i'}^{(l)}, \mathbf{x}_{i'}^{(l)} - \mathbf{p}), \tag{23}$$

Likewise, $\eta^{(l)}$ is a two-layer ReLU-MLP for the concatenated inputs, and $\|$ denotes concatenation over sampling levels.

C More Implementation Details

We use PyTorch [33] to implement our method and run experiments on a single NVIDIA GeForce GTX 1080 Ti GPU. We train the network using the Adam optimizer [27] with the initial learning rate is set as 10^{-3} for fast convergence for 200K iterations, followed by a finetuning of 100K iterations with the learning rate 10^{-4}. Other hyperparameters and initializations follow the default setups in PyTorch. For the reconstruction of ShapeNet objects, we follow [30,34] to sample 3000 points from the mesh and apply the Gaussian noise with standard deviation 0.005. For scene-level indoor room reconstruction, the

number of input points is 10000, as in [34]. We reduce the Gaussian noise level to 0.001 standard deviation, in order to match the change of the scale to have comparable level-of-detail information as the object dataset.

The number of neighbors in k-NN graphs is set as $k = 20$ for all the graph convolution layers. For the multi-scale graph structure, the point set is downsampled twice with farthest point sampling (FPS) to 20% and 5% of the original cardinality respectively. The permutation invariant function AGGRE is a mean-pooling aggregation for vector features and a max-pooling for scalar features. Empirically, we find that using vector max-pooling function as in [17] generates artifacts in the qualitative results, so we simply take the average of the vector features. For the non-equivariant GraphONet, the number of output feature channels is set as 64 for all the layers in the graph latent feature extractor function Φ. For the equivariance model E-GraphONets with hybrid features, the number of output channels for the vector features is 8, and 32 for scalar features. For the input geometric features, the 3D coordinates or the relative position are considered as 3 channels for the GraphONet, or 1 vector channel and 0 scalar channel for the equivariant layer. For both equivariant and non-equivariant models, the implicit decoder F is the same as that in the ConvONet [34], which is a light-weight ReLU-MLP architechture with skip-connections.

D Additional Experiments and Results

D.1 Vector vs. scalar channels in hybrid feature equivariant layers.

We extend the ablation experiments on hybrid feature channels in Fig. 7 of the main paper. Here we show that our hybrid feature paradigm benefits different architectures and tasks. For implicit surface reconstruction, we evaluate the equivariant implicit model without a graph embedding. We follow the VN-ONet architecture and the implementation details from [17], and use hybrid layers instead of pure vector neuron layers. In addition, we evaluate point cloud classification on the ModelNet40 dataset. Similarly, the architecture and the experimental setups follow VN-PointNet from [17], and we replace a portion of vector channels with scalars in each layer. We evaluate the performance with different ratios of vector channels, where one vector channel is equivalent to three scalar channels.

In Table 6, we report the performance with different ratio of vector channels, where one vector channel is considered equivalent to three scalar channels. In both cases, our method with hybrid features achieves higher accuracy than pure vector features (100%) as in [17]. The conclusion is consistent with the ablation experiments in Fig. 7 of the main paper, and our hybrid feature paradigm is advantageous in general cases.

D.2 Ablation on the architecture.

We ablate the implementation choices of our models. First, we explore how our models perform without the multi-scale sampling design on the ShapeNet object reconstruction and the Synthetic Room (SynRoom) scene reconstruction tasks. In Table 7, we show that the scene reconstruction performance drops more without the multi-scale architecture,

Table 6: **Ablation on vector vs. scalar channels.** We evaluate on ShapeNet surface reconstruction with non-graph structured equivariant implicit models, and ModelNet40 point cloud classification with equivariant point cloud networks, for which we follow the VN-PointNet and the VN-ONet architectures in [17] and use hybrid layers instead of pure vector neuron layers.

Ratio of vector channels	0%	12.5%	25%	50%	75%	87.5%	100%
ShapeNet implicit surface reconstruction (mIoU)	0.408	0.630	0.707	**0.719**	**0.719**	0.704	0.694
ModelNet40 point cloud classification (mAcc)	0.808	0.830	0.852	**0.856**	0.855	0.852	0.847

Table 7: **Graph functions with and without multi-scale graph neighbor samplings.** The multi-scale structure improves more on scene reconstruction performance.

Model	GraphONet		E-GraphONet	
Dataset	ShapeNet	SynRoom	ShapeNet	SynRoom
Multi-scale sampling	**0.904**	**0.883**	0.890	0.848
Single-scale sampling	0.897 [-0.007]	0.859 [-0.024]	0.884 [-0.006]	0.814 [-0.034]

while the difference in object reconstruction performance is subtle. We argue that scene reconstruction is a more complex task, so the multi-scale design plays a more important role to aggregate global and local context in different scales.

Then, we show the effect of using different point cloud encoders in our graph models and the baseline methods ConvONets [34]. The results are in Table 8. For the graph models, the scene-level reconstruction performance drops much more when the graph encoder is replaced by a PointNet encoder. The results indicate that both the locality modelling and the awareness of the translation equivariance from the graph encoder are more crucial for scene-level reconstruction as a more comlex task. However, in ConvONets, using the graph point encoder instead of the PointNet encoder does not lead to a significantly improved performance. Unlike our graph methods, ConvONets learn the latent feature with an intermediate grid feature tensor. So feature embedding in ConvONet relies more on the regular convolution layers applied on the grid feature, while the point encoder plays a less important role.

Next, we show how the model performs under different numbers of neighboring points k and layers L. In Table 9 we report the mean IoU $\uparrow$ on the ShapeNet dataset. From the results, using too smaller k and L could suffer from underfitting, while using too large values increases computation cost and may cause over-smoothed graph features.

D.3 Learning curve with limited training data.

We explore how the implicit model performs with very few training data of 130 examples, and provide the validation loss curve, as illustrated in Fig. 9. This result is a supplement to the test performance in Table 4 of the main paper. Both ConvONet-2D and ConvONet-3D suffer from overfitting in the very early stage of training, prior to 500 training steps. By contrast, our graph methods are able to learn properly from very few training data. The equivariant model is with better validation loss and more stable learning curve,

Table 8: **Implicit functions with different point encoders.** Graph models with graph point encoders replaced by PointNets would lead to more performance drop on scenes than on objects, because PointNet models no locality or translation equivariance which are more crucial for scenes; ConvONets with different point encoders show similar performance, because in such methods, feature embedding relies more on the grid encoder than the point encoder.

Model	GraphONet		E-GraphONet		ConvONet-2D		ConvONet-3D	
Dataset	ShapeNet	SynRoom	ShapeNet	SynRoom	ShapeNet	SynRoom	ShapeNet	SynRoom
Graph encoder	**0.904**	**0.883**	0.890	0.848	0.881 [-0.003]	0.803 [+0.001]	0.872 [+0.002]	0.853 [+0.006]
PointNet encoder	0.887 [-0.017]	0.826 [-0.057]	0.879 [-0.011]	0.797 [-0.051]	0.884	0.802	0.870	0.847

Table 9: **Graph implicit functions with different numbers of neighboring points k and layers L.** Reporting IoU ↑ on the ShapeNet dataset.

Evaluating IoU↑	$k=6$	$k=12$	$k=20$	$k=32$	$k=64$
GraphONet / E-GraphONet	0.860 / 0.813	0.885 / 0.867	**0.904** / **0.890**	0.902 / **0.890**	0.891 / 0.872
Evaluating IoU↑	$L=1$	$L=2$	$L=3$	$L=4$	$L=5$
GraphONet / E-GraphONet	0.824 / 0.659	**0.904** / **0.890**	0.900 / 0.887	0.890 / 0.874	0.884 / 0.865

which indicates that the equivariance property works as a regularization that controls the model complexity.

D.4 More qualitative and quantitative results.

We evaluate ShapeNet reconstruction performance by each object category. The results are shown in Table 10, where our graph model shows better performance with most of the object categories.

In Fig. 10, we show some additional ShapeNet reconstruction examples under transformations. Our final equivariance model guarantees equivariance to all kinds of similarity transformations.

Though not the main focus of this paper, our method scale to scene-level reconstructions, and we show some more room reconstruction examples in Fig. 11. Our GraphONet models better details. The equivariant model E-GraphONet achieves relatively good performance but generates more noisy artifacts, especially on the synthetic-to-real evaluation on the ScanNet dataset with corrupt areas in the point cloud scans. We argue that the restricted representation power of the equivariant layers limits the model to learn denoising and completion alongside reconstruction while generalize to more complex corrupted scenes. See the discussion on the limitation in the main paper.

E Limitations and Future Work

A limitation that comes with equivariance is that, by constraining the model complexity to conform to equivariant designs, expressive power may be affected as well. As a consequence, accuracy drops in stylized settings and datasets. In particular, we observe

that the shapes generated from equivariance models are usually less smooth than non-equivariant methods, especially for the more complex scenes. See Fig. 8 in the main paper and Fig. 11 in the Appendix. We argue that the restricted power of equivariance models limits the ability to identify the denoised geometry from the noisy point observations, while at the same time, the equivariant model are designed to avoid leveraging the prior of the flat straight and planar structures aligned with the Cartesian coordinate axes. To this end, relevant future directions include exploring more powerful equivariant models, or incorporating filtering techniques for implicit fields.

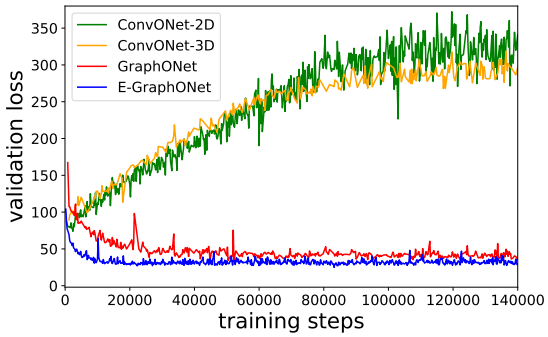

Fig. 9: **Learning curve with 130 training examples.** We show the validation loss curves. Graph methods can learn properly from very few training data, while ConvONets suffer from overfitting.

Table 10: **Category-specific ShapeNet reconstruction performance.** We evaluate our methods and compare with the baseline implicit representation models.

	ConvONet-2D [34]			ConvONet-3D [34]			IF-Net [11]			GraphONet			E-GraphONet-SO(3)		
	IoU	Chamfer	Normal	IoU	Chamfer	Normal	IoU	Chamfer	Normal	IoU	Chamfer	Normal	IoU	Chamfer	Normal
airplane	0.849	0.034	0.931	0.849	0.033	0.932	0.862	0.031	0.936	**0.881**	**0.027**	**0.941**	0.867	0.028	0.930
bench	0.830	0.035	0.921	0.791	0.041	0.911	0.815	0.037	0.915	**0.836**	**0.034**	**0.924**	0.807	0.037	0.906
cabinet	0.940	0.046	0.956	0.923	0.054	0.953	0.936	0.048	0.956	**0.943**	**0.047**	**0.958**	0.927	0.050	0.944
car	0.886	0.075	0.893	0.877	0.080	0.891	0.890	0.072	0.894	**0.897**	**0.068**	**0.895**	0.890	0.072	0.886
chair	0.871	0.046	0.943	0.853	0.049	0.942	0.878	0.043	0.946	**0.895**	**0.039**	**0.951**	0.879	0.043	0.939
display	0.927	0.036	0.968	0.904	0.042	0.965	0.923	0.036	0.968	**0.936**	**0.034**	**0.972**	0.922	0.036	0.963
lamp	0.785	0.059	0.900	0.792	0.066	0.910	0.820	0.047	0.916	0.847	0.042	**0.922**	**0.848**	**0.040**	0.915
loudspeaker	0.918	0.064	0.939	0.914	0.065	0.942	0.928	0.056	0.945	**0.938**	**0.053**	**0.946**	0.936	0.055	0.941
rifle	0.846	0.028	0.929	0.826	0.031	0.924	0.842	0.028	0.928	**0.877**	**0.022**	**0.943**	0.868	0.023	0.933
sofa	0.936	0.042	0.958	0.923	0.046	0.956	0.938	0.040	0.959	**0.946**	**0.037**	**0.963**	0.931	0.041	0.951
table	0.888	0.038	0.959	0.860	0.043	0.956	0.880	0.038	0.959	**0.896**	**0.036**	**0.963**	0.869	0.040	0.950
telephone	**0.955**	0.027	**0.983**	0.942	0.030	0.981	0.949	0.027	**0.983**	0.954	**0.026**	**0.983**	0.946	0.027	0.979
vessel	0.865	0.043	0.919	0.860	0.045	0.919	0.876	0.040	0.923	**0.901**	**0.033**	**0.934**	0.892	0.035	0.924
mean	0.884	0.044	0.938	0.870	0.048	0.937	0.887	0.042	0.941	**0.904**	**0.038**	**0.946**	0.890	0.041	0.936

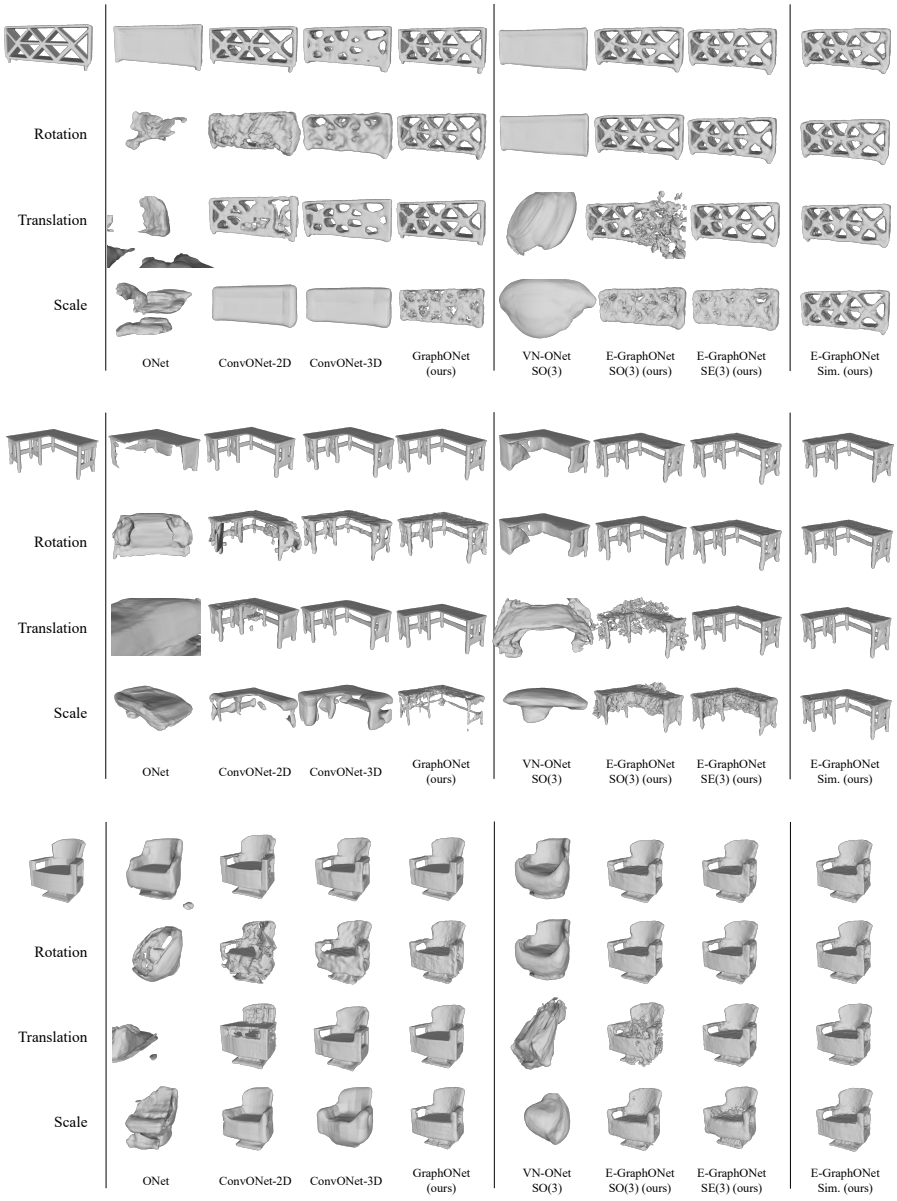

Fig. 10: **More examples on ShapeNet object reconstruction under unseen transformations.** With transformations we show the back-transformed shapes.

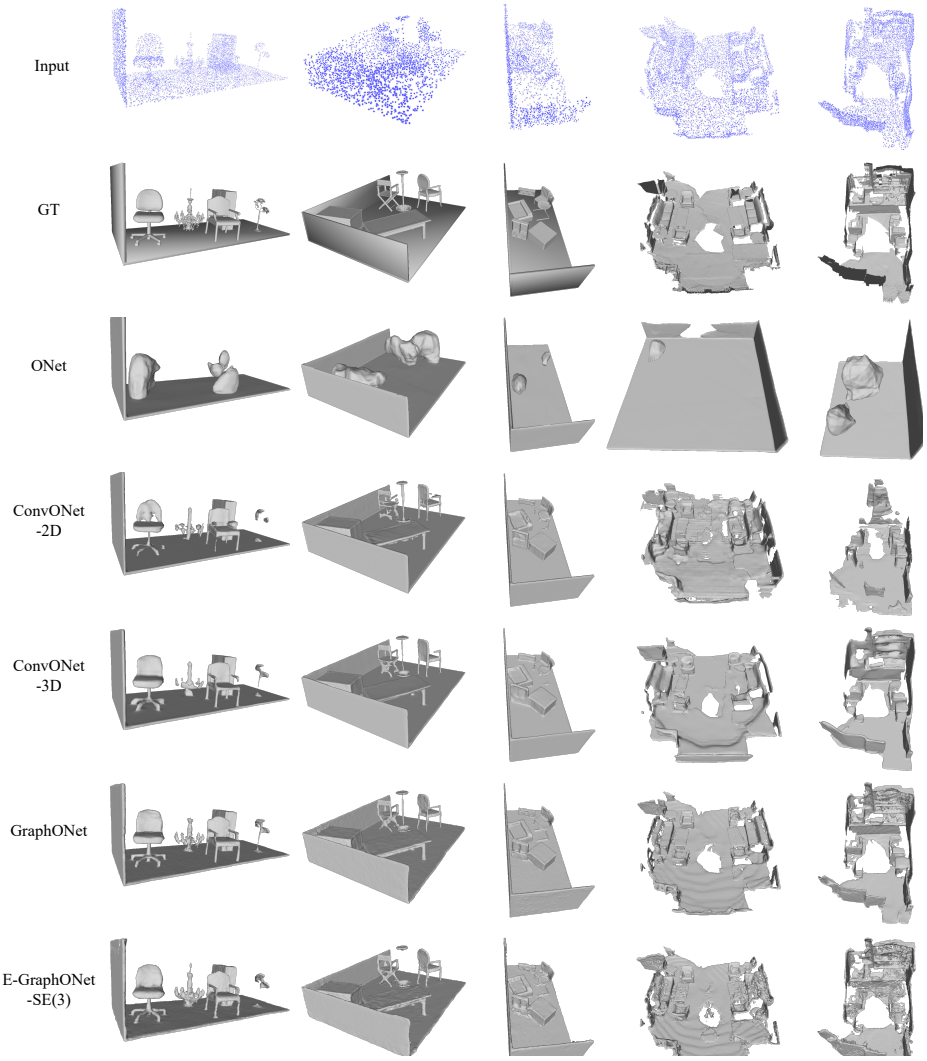

Fig. 11: **More scene reconstruction examples.** The left three columns are from the Synthetic Room dataset. The right two columns are from ScanNet.

References

1. Atzmon, M., Haim, N., Yariv, L., Israelov, O., Maron, H., Lipman, Y.: Controlling neural level sets. arXiv preprint arXiv:1905.11911 (2019) 3
2. Atzmon, M., Lipman, Y.: Sal: Sign agnostic learning of shapes from raw data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2565–2574 (2020) 3
3. Atzmon, M., Lipman, Y.: Sal++: Sign agnostic learning with derivatives. arXiv preprint arXiv:2006.05400 (2020) 3
4. Bautista, M.A., Talbott, W., Zhai, S., Srivastava, N., Susskind, J.M.: On the generalization of learning-based 3d reconstruction. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2180–2189 (2021) 4
5. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In: European Conference on Computer Vision. pp. 608–625. Springer (2020) 3, 6
6. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 11
7. Chatzipantazis, E., Pertigkiozoglou, S., Dobriban, E., Daniilidis, K.: Se (3)-equivariant attention networks for shape reconstruction in function space. arXiv preprint arXiv:2204.02394 (2022) 4
8. Chen, Y., Fernando, B., Bilen, H., Mensink, T., Gavves, E.: Neural feature matching in implicit 3d representations. In: International Conference on Machine Learning. pp. 1582–1593. PMLR (2021) 4
9. Chen, Y., Hu, V.T., Gavves, E., Mensink, T., Mettes, P., Yang, P., Snoek, C.G.: Pointmixup: Augmentation for point clouds. arXiv preprint arXiv:2008.06374 (2020) 4
10. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019) 1, 3
11. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6970–6981 (2020) 2, 3, 5, 7, 11, 26
12. Choy, C.B., Gwak, J., Savarese, S., Chandraker, M.: Universal correspondence network. arXiv preprint arXiv:1606.03558 (2016) 11
13. Cohen, T., Welling, M.: Group equivariant convolutional networks. In: International conference on machine learning. pp. 2990–2999. PMLR (2016) 4, 5
14. Cohen, T.S., Welling, M.: Steerable cnns. arXiv preprint arXiv:1612.08498 (2016) 4
15. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5828–5839 (2017) 14
16. Davies, T., Nowrouzezahrai, D., Jacobson, A.: On the effectiveness of weight-encoded neural implicit 3d shapes. arXiv preprint arXiv:2009.09808 (2020) 2, 4, 8, 11
17. Deng, C., Litany, O., Duan, Y., Poulenard, A., Tagliasacchi, A., Guibas, L.: Vector neurons: A general framework for so (3)-equivariant networks. arXiv preprint arXiv:2104.12229 (2021) 2, 3, 4, 8, 9, 10, 11, 12, 13, 18, 23, 24
18. Erler, P., Guerrero, P., Ohrhallinger, S., Mitra, N.J., Wimmer, M.: Points2surf learning implicit surfaces from point clouds. In: European Conference on Computer Vision. pp. 108–124. Springer (2020) 2, 3
19. Fuchs, F., Worrall, D., Fischer, V., Welling, M.: Se (3)-transformers: 3d roto-translation equivariant attention networks. Advances in Neural Information Processing Systems **33** (2020) 4

20. Fujiwara, K., Hashimoto, T.: Neural implicit embedding for point cloud analysis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11734–11743 (2020) 3
21. Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.: Deep structured implicit functions. arXiv preprint arXiv:1912.06126 (2019) 2
22. Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.: Local deep implicit functions for 3d shape. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4857–4866 (2020) 3
23. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International conference on machine learning. pp. 1263–1272. PMLR (2017) 7
24. Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S., Cohen-Or, D.: Meshcnn: a network with an edge. ACM Transactions on Graphics (TOG) **38**(4), 1–12 (2019) 7
25. Jiang, C., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T.: Local implicit grid representations for 3d scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020) 2, 3, 6
26. Jiang, Y., Ji, D., Han, Z., Zwicker, M.: Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1251–1261 (2020) 3
27. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 11, 22
28. Li, J., Chen, B.M., Lee, G.H.: So-net: Self-organizing network for point cloud analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9397–9406 (2018) 3
29. Liu, S., Saito, S., Chen, W., Li, H.: Learning to infer implicit surfaces without 3d supervision. In: Advances in Neural Information Processing Systems. pp. 8295–8306 (2019) 3
30. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019) 1, 3, 4, 5, 11, 22
31. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3504–3515 (2020) 3
32. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019) 1, 3
33. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703 (2019) 11, 22
34. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16. pp. 523–540. Springer (2020) 2, 3, 5, 7, 8, 11, 13, 14, 22, 23, 24, 26
35. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017) 4, 7
36. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems **30** (2017) 3
37. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015) 7

38. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2304–2314 (2019) 3
39. Satorras, V.G., Hoogeboom, E., Welling, M.: E(n) equivariant graph neural networks. arXiv preprint arXiv:2102.09844 (2021) 4
40. Shen, W., Zhang, B., Huang, S., Wei, Z., Zhang, Q.: 3d-rotation-equivariant quaternion neural networks. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16. pp. 531–547. Springer (2020) 4, 8, 18
41. Simeonov, A., Du, Y., Tagliasacchi, A., Tenenbaum, J.B., Rodriguez, A., Agrawal, P., Sitzmann, V.: Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 6394–6400. IEEE (2022) 4
42. Sitzmann, V., Chan, E., Tucker, R., Snavely, N., Wetzstein, G.: Metasdf: Meta-learning signed distance functions. Advances in Neural Information Processing Systems **33**, 10136–10147 (2020) 4
43. Sosnovik, I., Szmaja, M., Smeulders, A.: Scale-equivariant steerable networks. In: International Conference on Learning Representations (2019) 4
44. Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11358–11367 (2021) 3
45. Tang, J.H., Chen, W., Wang, B., Liu, S., Yang, B., Gao, L., et al.: Octfield: Hierarchical implicit functions for 3d modeling. Advances in Neural Information Processing Systems **34**, 12648–12660 (2021) 3
46. Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., Riley, P.: Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. arXiv preprint arXiv:1802.08219 (2018) 4, 5, 18
47. Wang, P.S., Liu, Y., Tong, X.: Dual octree graph networks for learning adaptive volumetric shape representations. arXiv preprint arXiv:2205.02825 (2022) 3
48. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. Acm Transactions On Graphics (tog) **38**(5), 1–12 (2019) 3, 7
49. Weiler, M., Geiger, M., Welling, M., Boomsma, W., Cohen, T.: 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In: NeurIPS (2018) 4
50. Weiler, M., Hamprecht, F.A., Storath, M.: Learning steerable filters for rotation equivariant cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 849–858 (2018) 4
51. Worrall, D.E., Garbin, S.J., Turmukhambetov, D., Brostow, G.J.: Harmonic networks: Deep translation and rotation equivariance. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5028–5037 (2017) 4
52. Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U.: Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In: Advances in Neural Information Processing Systems. pp. 492–502 (2019) 3
53. Xu, Y., Fan, T., Yuan, Y., Singh, G.: Ladybird: Quasi-monte carlo sampling for deep implicit field based 3d reconstruction with symmetry. In: European Conference on Computer Vision. pp. 248–263. Springer (2020) 3
54. Yuan, Y., Nüchter, A.: An algorithm for the se (3)-transformation on neural implicit maps for remapping functions. IEEE Robotics and Automation Letters (2022) 4
55. Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R., Smola, A.: Deep sets. arXiv preprint arXiv:1703.06114 (2017) 4
56. Zhu, W., Qiu, Q., Calderbank, R., Sapiro, G., Cheng, X.: Scale-equivariant neural networks with decomposed convolutional filters. arXiv preprint arXiv:1909.11193 (2019) 4