DITTO: Dual and Integrated Latent Topologies for Implicit 3D Reconstruction

Supplementary Material

Jaehyeok Shim and Kyungdon Joo[1]
Artificial Intelligence Graduate School, UNIST
{jh.shim,kyungdon}@unist.ac.kr
https://vision3d-lab.github.io/ditto

Overview

In this supplementary material, we provide detailed descriptions of DITTO that could not be handled in the main paper due to space constraints. In Sec. 1, we describe more details necessary for implementing our work, such as detailed network architecture and hyperparameters. Additional ablation studies related to DITTO are available in Sec. 2. We provide additional experiment results in Sec. 3.

1. Implementation Details

Additional details of the DITTO network architecture are available in Sec. 1.1, and hyperparameters that are used for training DITTO are available in Sec. 1.2.

1.1. Network Architecture Details

In this section, we provide additional network architecture details of our work. DITTO mainly consists of the dual latent encoder and the integrated implicit decoder (IID). The dual latent encoder can be subdivided into a point encoder and an UNet with dual latent layers (DLLs). An illustration of the dual latent encoder is available in Fig. 1. We describe details of each of the modules below. In addition, we provide layer-level details in Table 4.

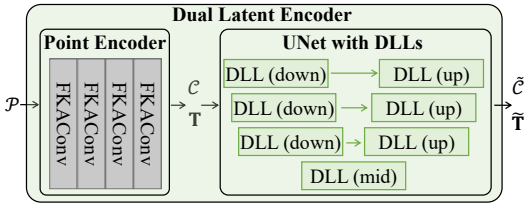


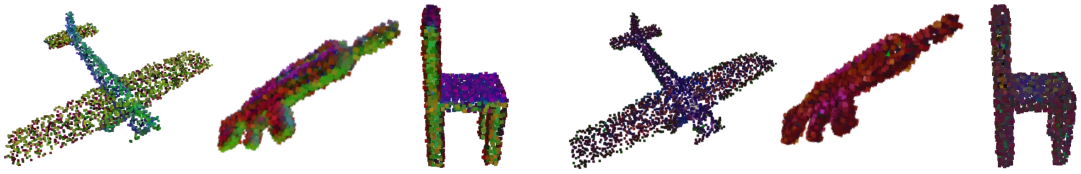
Figure 1. **Detailed illustration of our dual latent encoder**.

Point Encoder. The point encoder receives input point cloud $\mathcal{P}$, and generates point latents $\mathcal{C}$. This module then produces grid latents $\mathbf{T}$ by projecting $\mathcal{C}$ onto either triplanes or voxels. While extracting $\mathcal{C}$, we employ a stack of four FKAConv layers [2] instead of the conventional architecture based on PointNet [6], called as local pooled PointNet. Local pooled PointNet generates point features by directly encoding the point coordinates, but this layer processes each point independently, without considering the relationship between points. In contrast, simply employing a FKAConv-based encoder similar to POCO [1] gives additional performance gains. This improvement is further reinforced by DLL and IID due to their emphasis on point latents.

Dual Latent Layer. Our DLL focuses on refining point latents. To analyze what DLL learns, we visualize point features in Fig. 2. The features of DITTO exhibit clear boundaries between different parts of the object. For instance, the body of the airplane has a distinct color compared to its wings, and similarly, the bottom and side parts of the chair and the gun have different colors. In contrast, most of the point features of ALTO [10] have similar colors. From this difference, we infer that DITTO appears to implicitly learn semantic information, such as planes and their direction or curvature. We expect that these point features can help provide a clear surface boundary between two surfaces that are adjacent yet not in contact. We would like to note that DITTO considers point-level geometry with a point feature extractor, such as the proposed DSPT, while ALTO handles point latents using MLPs that account for each point independently.

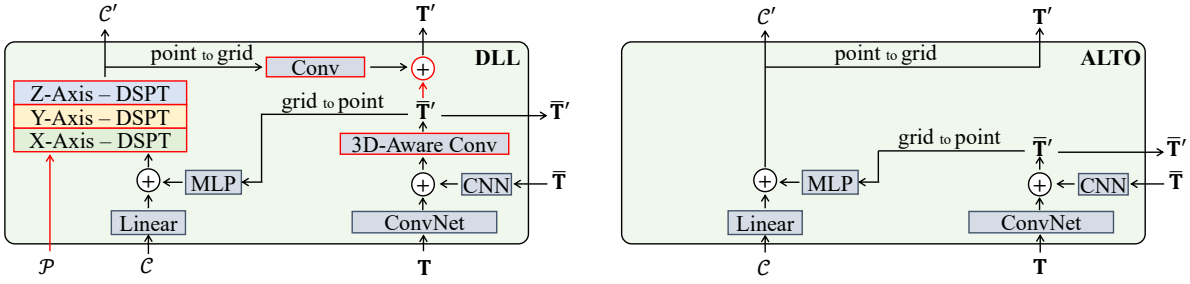
*Corresponding author.

We provide a visual comparison of our DLL, along with the architecture design of ALTO (see Fig. 3). Our design of DLL mainly differs in two aspects: the DSPT layers for point features refinement and the skip-connection from point latents to grid latents. ALTO outputs refined grid latents $\mathbf{T}'$ that are directly projected from the point latents. These grid latents have many empty cells, hindering feature extraction of grid latents. To address this problem, we find that simply creating a residual connection between grid latents and those projected from point latents can enhance performance (see our additional ablation study in Sec. 2.1). Additionally, we incorporate a convolutional layer to reduce sparseness of grid latents projected from point latents.



(a) Point features of DITTO (b) Point features of ALTO [10]

Figure 2. **Visualization of point features**. We visualize the of the refined point features $\tilde{\mathcal{C}}$. These features are colored by reducing feature dimension into three channels by using principal component analysis (PCA).



(a) DITTO (ours) (b) ALTO [10]

Figure 3. **Visual comparison of DLL and a layer of ALTO**. Key differences are highlighted with red outlines and red arrows.

Dynamic Sparse Transformer. IID receives point coordinates $\mathcal{P}$ and their features $\mathcal{C}$, and enhances the point features. When dividing the point features into windows $\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L}$, we recycle the sorting indices in each DSPT layer to reduce computation load. Concretely, since both the number and the coordinates of points remain constant throughout all processes of DITTO, we initially calculate the sorted indices of point coordinates and reuse them in every DSPT layer. This method effectively reduces the computation needed for recalculating sorted indices.

UNet with Dual Latent Layers. Our UNet architecture is similar to traditional UNet [4, 9], but each layer is replaced with the proposed DLL module. Our UNet has three down DLLs, a mid DLL, and three up DLLs (see Fig. 1). The down DLL can optionally downsample the grid latents using MaxPooling, while the up DLL can upsample them using transposed convolution. Specifically, within our UNet, the second and third down DLLs downsample the grid latents, whereas the first and second up DLLs upsample them.

Integrated Implicit Decoder. IID receives the refined latents and estimates occupancy by comprehensive consideration of these latents. Note that, while Table 4 describes IID with a single query point $\mathbf{q}$ for convenience, IID actually processes multiple query points $\mathcal{Q}$ in parallel.

1.2. Hyperparameters

The detailed hyperparameters can be found in Table 1.

2. Ablation Studies

We conduct ablation studies to demonstrate effectiveness of each module of DITTO. Specifically, we perform ablation studies on the point encoder, a residual connection, and the number of windows for DSPT in Sec. 2.1, and Sec. 2.2, respectively.

Table 1. **Detailed hyperparameters of DITTO.**

Notation	Meaning	Object (3K) Triplane	Object (1K) Triplane	Object(0.3K) Triplane	Scene (10K) Triplane	Scene (10K) Voxel	Scene (3K) Triplane	Scene (3K) Voxel
	Epoch	1,000	1,000	1,000	2,500	2,500	2,500	2,500
	Learning rate	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
	Batch size	32	32	32	32	16	32	16
R	Feature resolution	64	64	64	128	64	128	64
d	Channel size	32	32	32	32	32	32	32
L	# of windows in DSPT	25	25	25	20	20	25	25
K	# of neighbor points in IID	32	32	32	32	32	32	32
M	# of query points per training iteration	2,048	2,048	2,048	2,048	2,048	2,048	2,048

2.1. Ablation Study on Residual Connection

To demonstrate the impact of residual connections between grid latents and those projected from point latents, we conduct an ablation study comparing ALTO with and without this residual connection (see Table 2). The results suggest that only a simple addition of residual connection can significantly enhance performance. Moreover, a convolutional layer can improve performance by reducing the sparsity of grid latents projected from point latents.

Table 2. **Ablation study on residual connection**. Training and inference are conducted on the Synthetic Rooms dataset with 10K input points and 0.005 noise level.

# of windows	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑
ALTO (triplane)	0.895	0.37	0.910	0.974
ALTO + residual connection	0.904	**0.36**	**0.915**	0.976
ALTO + residual connection + conv layer	**0.907**	**0.36**	**0.915**	**0.977**

2.2. Ablation Study on DSPT

We conduct an ablation study to determine the appropriate number of windows for DSPT. The results can be found in Table 3. The results demonstrate robustness across various window numbers. However, the current window number ($L = 20$) shows slightly improved performance.

Table 3. **Ablation study on the number of windows in our DSPT**. The training and inference are conducted on the Synthetic Rooms dataset with 10K input points and 0.005 noise level.

# of windows	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑
40	0.929	0.34	0.930	**0.984**
25	0.930	0.34	0.930	0.983
20 (DITTO)	**0.931**	**0.33**	**0.931**	**0.984**
10	0.929	0.34	0.930	**0.984**

3. Additional Experiment Results

In this section, we provide additional qualitative results in Sections 3.1, 3.2, 3.3.

3.1. Additional Results on ShapeNet

Quantitative Results. We provide additional object-level 3D surface reconstruction results on ShapeNet [3]. Detailed per-category quantitative results are presented with different input point densities: 3K input points in Table 5, 1K input points in Table 6, and 0.3K input points in Table 7, all at a consistent noise level of 0.005. Each of these tables is a per-category extension to Table 1 in the main paper. DITTO outperforms previous methods in most categories. Note that, while ALTO [10] outperforms POCO [1] in most of metrics, POCO shows higher F-score than ALTO when dealing with 3K input points. In contrast, DITTO demonstrates superior performance in most metrics and categories.

Qualitative Results. We present additional object-level 3D reconstruction results on ShapeNet. The result meshes are visualized in Fig. 4 for 3K input points at 0.005 noise level. DITTO shows high-fidelity reconstruction especially thin and

intricate structures such as the legs of the chairs and the tables (second and third rows). Enhanced details are also notable, such as the boundary interface between two parts of the chair (second row).

To demonstrate robustness for sparsity, we visualize qualitative results in sparse case: 1K input points in Fig. 5, 0.3K input points in Fig. 6 with consistent noise level. Even with sparse point clouds, DITTO shows superior reconstruction quality in intricate shapes, such as the bookshelf (first row in Fig. 5) and underside of the car (third row in Fig. 5). Moreover, the results of DITTO generate more clear shape details, such as the chair (second row in Fig. 5).

3.2. Additional Results on Synthetic Rooms

We provide additional qualitative results for scene-level 3D surface reconstruction on the Synthetic Rooms dataset [8]. We visualize the results in Fig. 7 for 10K input points with 0.005 noise level. Result meshes of DITTO exhibit clear surface boundaries in intricate cases, such as the bookshelf (box in the left scene), and the lamps (boxes in the middle and right scenes). In addition, due to the precise detail reconstruction capability of DITTO, it can successfully reconstruct fine details of chairs (boxes in the middle and right scenes) and lamps (boxes in the middle scene).

To demonstrate the performance with sparse input point clouds, we visualize the results for 3K input points in Fig. 8. Even with sparse point clouds, DITTO outperforms previous methods. DITTO is the only method that successfully reconstructs the chairs (boxes in the left and middle scenes). In addition, DITTO is the most successful method in reconstructing the bookshelves (boxes in the right scene).

3.3. Additional Results on ScanNet-V2

We present additional qualitative results on ScanNet-v2 [5] to demonstrate generalization performance. Consistent with the main paper, we test on this dataset using models pre-trained with the Synthetic Rooms dataset (see Fig. 9). DITTO successfully reconstructs the tables and sofas (boxes in the left and middle scenes). For the right scene, our method successfully reconstructs the tables and chairs, even though every previous method fails to generate the scene accurately. These results demonstrate the robustness of DITTO in handling complex geometries and details, even in the dataset that is not used during the training phase.

Table 4. **Layer-level network module architecture designs of `DITTO`.**

Layer Name	Input	Output
Point Encoder		
Input / Output	$\mathcal{P}\ (N \times 3)$	$\mathcal{C}\ (N \times d), \mathbf{T}\ (3 \times R \times R \times d)$
FKAConv layer	$\mathcal{P}\ (N \times 3)$	$\mathcal{C}\ (N \times d)$
FKAConv layer	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d)$	$\mathcal{C}\ (N \times d)$
FKAConv layer	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d)$	$\mathcal{C}\ (N \times d)$
FKAConv layer	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d)$	$\mathcal{C}\ (N \times d)$
Quantization	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d)$	$\mathbf{T}\ (3 \times R \times R \times d)$
DLL		
Input / Output	$\mathbf{T}\ (3 \times R \times R \times d_{\text{in}}), \mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d_{\text{in}})$	$\mathbf{T}'\ (3 \times R/2 \times R/2 \times d_{\text{out}}), \mathcal{C}'\ (N \times d_{\text{out}})$
ConvNet	$\mathbf{T}\ (3 \times R \times R \times d_{\text{in}})$	$\mathbf{T}\ (3 \times R \times R \times d_{\text{out}})$
Conv2d	$\bar{\mathbf{T}}\ (3 \times R \times R \times d_{\text{in}})$	$\bar{\mathbf{T}}\ (3 \times R \times R \times d_{\text{out}})$
Sum	$\mathbf{T}\ (3 \times R \times R \times d_{\text{out}}), \bar{\mathbf{T}}\ (3 \times R \times R \times d_{\text{out}})$	$\bar{\mathbf{T}}\ (3 \times R \times R \times d_{\text{out}})$
3D-Aware-Conv	$\bar{\mathbf{T}}\ (3 \times R \times R \times d_{\text{out}})$	$\bar{\mathbf{T}}'\ (3 \times R \times R \times d_{\text{out}})$
Linear	$\mathcal{C}\ (N \times d_{\text{in}})$	$\mathcal{C}\ (N \times d_{\text{out}})$
Grid-to-Point	$\bar{\mathbf{T}}'\ (3 \times R \times R \times d_{\text{out}}), \mathcal{P}\ (N \times 3)$	$(N \times d_{\text{out}})$
MLP	$(N \times d_{\text{out}})$	$(N \times d_{\text{out}})$
Sum	$(N \times d_{\text{out}}), \mathcal{C}\ (N \times d_{\text{out}})$	$\mathcal{C}\ (N \times d_{\text{out}})$
X-Axis DSPT	$\mathcal{P}(N \times 3), \mathcal{C}\ (N \times d_{\text{out}})$	$\mathcal{C}\ (N \times d_{\text{out}})$
Y-Axis DSPT	$\mathcal{P}(N \times 3), \mathcal{C}\ (N \times d_{\text{out}})$	$\mathcal{C}\ (N \times d_{\text{out}})$
Z-Axis DSPT	$\mathcal{P}(N \times 3), \mathcal{C}\ (N \times d_{\text{out}})$	$\mathcal{C}'\ (N \times d_{\text{out}})$
Point-to-Grid	$\mathcal{P}(N \times 3), \mathcal{C}'\ (N \times d_{\text{out}})$	$(3 \times R \times R \times d_{\text{out}})$
Conv2d	$(3 \times R \times R \times d_{\text{out}})$	$(3 \times R \times R \times d_{\text{out}})$
Sum	$(3 \times R \times R \times d_{\text{out}}), \bar{\mathbf{T}}'\ (3 \times R \times R \times d_{\text{out}})$	$\mathbf{T}'\ (3 \times R \times R \times d_{\text{out}})$
Pooling	$\mathbf{T}'\ (3 \times R \times R \times d_{\text{out}})$	$\mathbf{T}'\ (3 \times R/2 \times R/2 \times d_{\text{out}})$
UNet with DLLs		
Input / Output	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d), \mathbf{T}\ (3 \times R \times R \times d)$	$\mathbf{T}'\ (3 \times R \times R \times d), \tilde{\mathcal{C}}\ (N \times d)$
DLL (down)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d), \mathbf{T}\ (3 \times R \times R \times d)$	$\mathcal{C}'\ (N \times d), \mathbf{T}'\ (3 \times R \times R \times d)$
DLL (down)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d), \mathbf{T}\ (3 \times R \times R \times d)$	$\mathcal{C}'\ (N \times 2d), \mathbf{T}'\ (3 \times R/2 \times R/2 \times 2d)$
DLL (down)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times 2d), \mathbf{T}\ (3 \times R/2 \times R/2 \times 2d)$	$\mathcal{C}'\ (N \times 4d), \mathbf{T}'\ (3 \times R/4 \times R/4 \times 4d)$
DLL (mid)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times 4d), \mathbf{T}\ (3 \times R/4 \times R/4 \times 4d)$	$\mathcal{C}'\ (N \times 8d), \mathbf{T}'\ (3 \times R/4 \times R/4 \times 8d)$
DLL (up)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times 8d), \mathbf{T}\ (3 \times R/4 \times R/4 \times 8d)$	$\mathcal{C}'\ (N \times 4d), \mathbf{T}'\ (3 \times R/2 \times R/2 \times 4d)$
DLL (up)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times 4d), \mathbf{T}\ (3 \times R/2 \times R/2 \times 4d)$	$\mathcal{C}'\ (N \times 2d), \mathbf{T}'\ (3 \times R \times R \times 2d)$
DLL (up)	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times 2d), \mathbf{T}\ (3 \times R \times R \times 2d)$	$\mathcal{C}'\ (N \times d), \mathbf{T}'\ (3 \times R \times R \times d)$
DSPT		
Input / Output	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d)$	$\mathcal{C}\ (N \times d)$
sort	$\mathcal{P}\ (N \times 3), \mathcal{C}\ (N \times d)$	$(N \times d)$
split	$(N \times d)$	$\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L}$
Self-attention	$\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L}$	$\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L}$
split^{-1}	$\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L}$	$(N \times d)$
sort^{-1}	$(N \times d)$	$\mathcal{C}\ (N \times d)$
Layer norm	$\mathcal{C}\ (N \times d)$	$\mathcal{C}\ (N \times d)$
Linear	$\mathcal{C}\ (N \times d)$	$\mathcal{C}\ (N \times 4d)$
ReLU	$\mathcal{C}\ (N \times 4d)$	$\mathcal{C}\ (N \times 4d)$
Linear	$\mathcal{C}\ (N \times 4d)$	$\mathcal{C}\ (N \times d)$
IID		
Input / Output	$\tilde{\mathbf{T}}\ (3 \times R \times R \times d), \mathcal{P}\ (N \times 3), \tilde{\mathcal{C}}\ (N \times d), \mathbf{q}\ (3)$	$\tilde{o}\ (1)$
Interpolation	$\tilde{\mathbf{T}}\ (3 \times R \times R \times d), \mathbf{q}\ (3)$	(d)
Linear	(d)	$\mathbf{z}_0\ (2d)$
KNN	$\mathbf{q}\ (3), \mathcal{P}\ (N \times 3), \tilde{\mathcal{C}}\ (N \times d)$	$\mathcal{P}^{\mathbf{q}}\ (K \times 3), \tilde{\mathcal{C}}^{\mathbf{q}}\ (K \times d)$
interpolation	$\tilde{\mathbf{T}}\ (3 \times R \times R \times d), \mathcal{P}^{\mathbf{q}}\ (K \times 3)$	$\tilde{\mathbf{T}}^{\mathbf{q}}\ (K \times d)$
concat	$\tilde{\mathcal{C}}^{\mathbf{q}}\ (K \times d), \tilde{\mathbf{T}}^{\mathbf{q}}\ (K \times d)$	$\mathcal{Z}^{\mathbf{q}}\ (K \times 2d)$
Self-attention	$\{\mathbf{z}_0^{\mathbf{q}}, \ldots, \mathbf{z}_K^{\mathbf{q}}\}\ ((K{+}1) \times 2d), \{\mathbf{q}, \mathbf{p}_1^{\mathbf{q}}, \ldots, \mathbf{p}_K^{\mathbf{q}}\}\ ((K{+}1) \times 3)$	$\mathbf{z}_0^{\mathbf{q}}\ (2d)$
Self-attention	$\{\mathbf{z}_0^{\mathbf{q}}, \ldots, \mathbf{z}_K^{\mathbf{q}}\}\ ((K{+}1) \times 2d), \{\mathbf{q}, \mathbf{p}_1^{\mathbf{q}}, \ldots, \mathbf{p}_K^{\mathbf{q}}\}\ ((K{+}1) \times 3)$	$\mathbf{z}_0^{\mathbf{q}}\ (2d)$
Self-attention	$\{\mathbf{z}_0^{\mathbf{q}}, \ldots, \mathbf{z}_K^{\mathbf{q}}\}\ ((K{+}1) \times 2d), \{\mathbf{q}, \mathbf{p}_1^{\mathbf{q}}, \ldots, \mathbf{p}_K^{\mathbf{q}}\}\ ((K{+}1) \times 3)$	$\mathbf{z}_0^{\mathbf{q}}\ (2d)$
Self-attention	$\{\mathbf{z}_0^{\mathbf{q}}, \ldots, \mathbf{z}_K^{\mathbf{q}}\}\ ((K{+}1) \times 2d), \{\mathbf{q}, \mathbf{p}_1^{\mathbf{q}}, \ldots, \mathbf{p}_K^{\mathbf{q}}\}\ ((K{+}1) \times 3)$	$\hat{\mathbf{z}}_0^{\mathbf{q}}\ (2d)$
Linear	$\hat{\mathbf{z}}_0^{\mathbf{q}}\ (2d)$	$\tilde{o}\ (1)$

Table 5. **Object-level quantitative comparison on ShapeNet with 3K input points with noise level 0.005**.

Method	IoU ↑					Chamfer-L_1 ↓				
	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)
Airplane	0.734	0.849	0.902	0.908	**0.935**	0.64	0.34	0.23	0.22	**0.19**
Bench	0.682	0.830	0.865	0.890	**0.919**	0.67	0.35	0.28	0.26	**0.23**
Cabinet	0.855	0.940	0.960	0.965	**0.976**	0.82	0.46	0.37	0.34	**0.31**
Car	0.830	0.886	0.921	0.924	**0.943**	1.04	0.75	0.41	0.43	**0.36**
Chair	0.720	0.871	0.919	0.925	**0.948**	0.95	0.46	0.33	0.32	**0.29**
Display	0.799	0.927	0.956	0.962	**0.973**	0.82	0.36	0.28	0.27	**0.25**
Lamp	0.546	0.785	0.877	0.868	**0.914**	1.59	0.59	0.33	0.34	**0.28**
Loudspeaker	0.826	0.918	0.957	0.953	**0.970**	1.18	0.64	0.41	0.41	**0.35**
Rifle	0.668	0.846	0.897	0.898	**0.925**	0.66	0.28	0.19	0.19	**0.16**
Sofa	0.865	0.936	0.963	0.966	**0.976**	0.73	0.42	0.30	0.29	**0.26**
Table	0.739	0.888	0.924	0.937	**0.956**	0.76	0.38	0.31	0.29	**0.27**
Telephone	0.896	0.955	0.968	0.977	**0.982**	0.46	0.27	0.22	0.21	**0.20**
Vessel	0.729	0.865	0.927	0.924	**0.948**	0.94	0.43	0.25	0.26	**0.22**
mean	0.761	0.884	0.926	0.931	**0.949**	0.87	0.44	0.30	0.30	**0.27**

Method	NC ↑					F-Score ↑				
	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)
Airplane	0.886	0.931	0.944	0.949	**0.958**	0.829	0.965	0.994	0.992	**0.997**
Bench	0.871	0.921	0.928	0.941	**0.950**	0.827	0.964	0.988	0.991	**0.996**
Cabinet	0.913	0.956	0.961	0.967	**0.970**	0.833	0.956	0.979	0.982	**0.989**
Car	0.874	0.893	0.894	**0.917**	0.914	0.747	0.849	0.946	0.940	**0.963**
Chair	0.886	0.943	0.956	0.959	**0.968**	0.730	0.939	0.985	0.985	**0.994**
Display	0.926	0.968	0.975	0.976	**0.981**	0.795	0.971	0.994	0.993	**0.997**
Lamp	0.809	0.900	0.929	0.924	**0.942**	0.581	0.892	0.975	0.962	**0.984**
Loudspeaker	0.903	0.939	0.952	0.951	**0.961**	0.727	0.892	0.964	0.955	**0.976**
Rifle	0.849	0.929	0.949	0.949	**0.960**	0.818	0.980	0.998	0.996	**0.999**
Sofa	0.928	0.958	0.967	0.971	**0.975**	0.832	0.953	0.989	0.987	**0.994**
Table	0.917	0.959	0.966	0.968	**0.975**	0.824	0.967	0.991	0.990	**0.996**
Telephone	0.970	0.983	0.985	0.987	**0.988**	0.930	0.989	0.998	0.998	**0.999**
Vessel	0.857	0.919	0.940	0.940	**0.952**	0.734	0.931	0.989	0.982	**0.992**
mean	0.891	0.938	0.950	0.954	**0.957**	0.785	0.942	0.984	0.981	**0.988**

Table 6. **Object-level quantitative comparison on ShapeNet with 1K input points with noise level 0.005**.

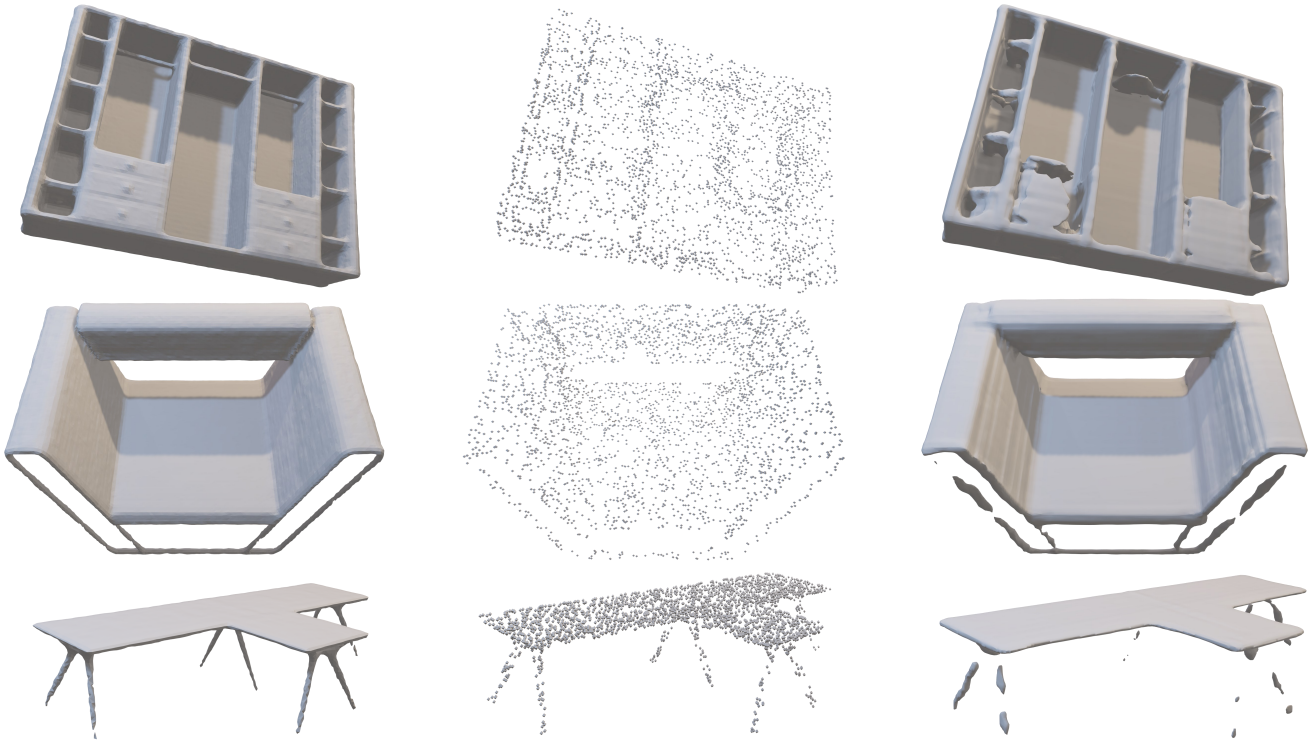
Method	IoU ↑					Chamfer-L_1 ↓				
	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)
Airplane	0.748	0.825	0.850	0.872	**0.908**	0.59	0.39	0.32	0.29	**0.23**
Bench	0.702	0.798	0.804	0.856	**0.891**	0.62	0.40	0.38	0.30	**0.26**
Cabinet	0.862	0.926	0.936	0.953	**0.964**	0.76	0.50	0.46	0.37	**0.35**
Car	0.837	0.867	0.878	0.901	**0.921**	0.99	0.83	0.60	0.50	**0.45**
Chair	0.736	0.837	0.867	0.894	**0.922**	0.89	0.55	0.44	0.39	**0.33**
Display	0.812	0.911	0.930	0.946	**0.960**	0.78	0.41	0.34	0.31	**0.28**
Lamp	0.567	0.741	0.807	0.820	**0.877**	1.44	0.68	0.50	0.50	**0.35**
Loudspeaker	0.831	0.899	0.923	0.933	**0.951**	1.14	0.72	0.54	0.48	**0.42**
Rifle	0.680	0.801	0.850	0.862	**0.892**	0.63	0.36	0.27	0.25	**0.20**
Sofa	0.873	0.921	0.937	0.952	**0.964**	0.69	0.47	0.38	0.33	**0.30**
Table	0.757	0.858	0.880	0.913	**0.937**	0.70	0.44	0.38	0.33	**0.30**
Telephone	0.897	0.946	0.953	0.968	**0.975**	0.46	0.29	0.26	0.23	**0.21**
Vessel	0.736	0.840	0.880	0.893	**0.923**	0.91	0.51	0.37	0.33	**0.27**
mean	0.772	0.859	0.884	0.905	**0.926**	0.82	0.50	0.40	0.35	**0.32**

Method	NC ↑					F-Score ↑				
	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)
Airplane	0.894	0.922	0.920	0.933	**0.949**	0.850	0.946	0.970	0.976	**0.990**
Bench	0.882	0.911	0.902	0.925	**0.940**	0.849	0.943	0.956	0.979	**0.990**
Cabinet	0.925	0.949	0.945	0.957	**0.964**	0.852	0.939	0.951	0.972	**0.978**
Car	**0.904**	0.885	0.867	0.889	**0.904**	0.763	0.819	0.868	0.912	**0.934**
Chair	0.893	0.931	0.930	0.946	**0.960**	0.753	0.902	0.943	0.965	**0.982**
Display	0.930	0.961	0.962	0.970	**0.976**	0.805	0.956	0.976	0.984	**0.991**
Lamp	0.820	0.885	0.895	0.905	**0.929**	0.606	0.845	0.924	0.926	**0.964**
Loudspeaker	0.914	0.929	0.928	0.936	**0.950**	0.740	0.863	0.908	0.926	**0.951**
Rifle	0.859	0.916	0.928	0.936	**0.949**	0.828	0.957	0.984	0.987	**0.994**
Sofa	0.937	0.950	0.950	0.960	**0.969**	0.846	0.932	0.961	0.974	**0.985**
Table	0.918	0.950	0.949	0.961	**0.970**	0.842	0.947	0.964	0.979	**0.989**
Telephone	0.972	0.980	0.979	0.984	**0.986**	0.940	0.983	0.990	0.994	**0.996**
Vessel	0.866	0.906	0.913	0.923	**0.940**	0.740	0.899	0.952	0.961	**0.979**
mean	0.901	0.929	0.928	0.940	**0.949**	0.801	0.918	0.950	0.964	**0.975**

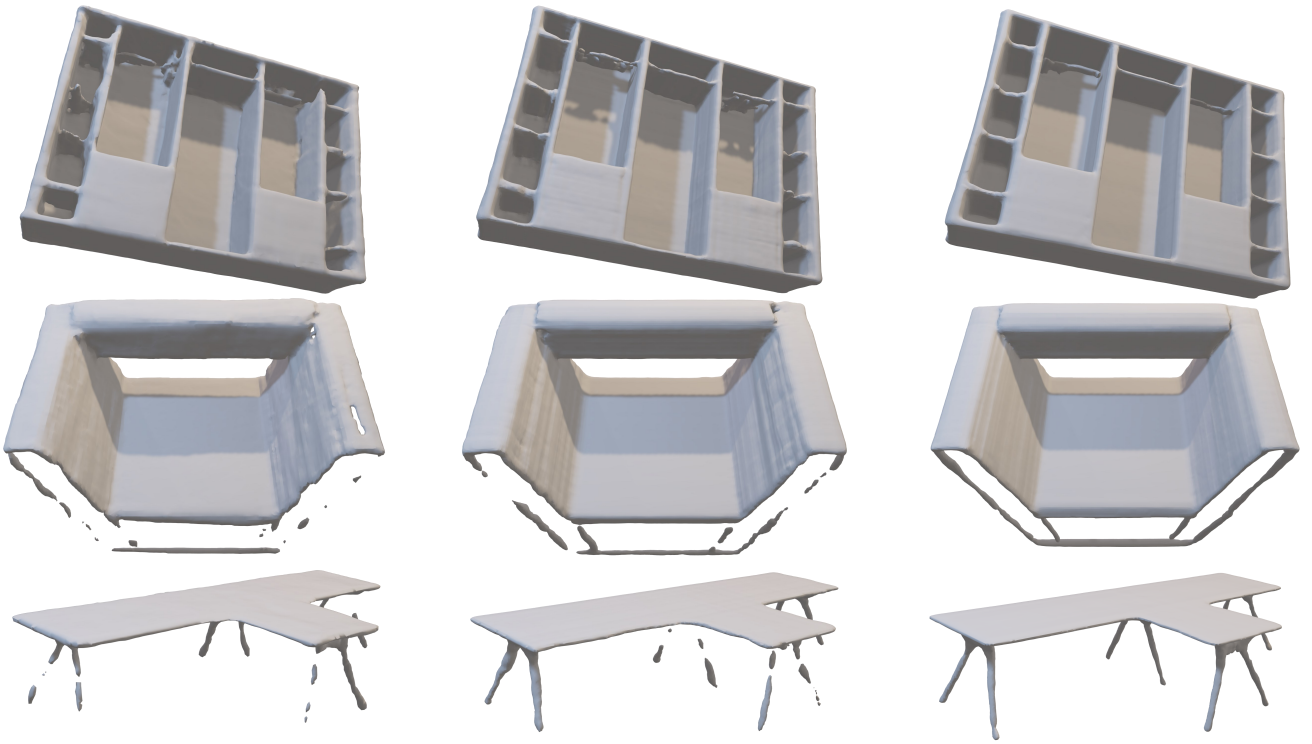
Table 7. **Object-level quantitative comparison on ShapeNet with 0.3K input points with noise level 0.005**.

Method	IoU ↑					Chamfer-L_1 ↓				
	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)
Airplane	0.760	0.782	0.744	0.825	**0.857**	0.57	0.48	0.57	0.39	**0.32**
Bench	0.716	0.743	0.707	0.801	**0.835**	0.60	0.50	0.56	0.39	**0.34**
Cabinet	0.867	0.900	0.889	0.927	**0.941**	0.73	0.52	0.58	0.46	**0.43**
Car	0.834	0.843	0.817	0.867	**0.885**	0.99	0.76	0.83	0.67	**0.61**
Chair	0.736	0.787	0.776	0.840	**0.871**	0.89	0.67	0.71	0.52	**0.45**
Display	0.817	0.885	0.878	0.917	**0.931**	0.76	0.47	0.49	0.38	**0.35**
Lamp	0.567	0.663	0.681	0.747	**0.808**	1.38	1.02	0.93	0.76	**0.61**
Loudspeaker	0.827	0.870	0.867	0.901	**0.916**	1.16	0.78	0.79	0.64	**0.59**
Rifle	0.691	0.757	0.742	0.801	**0.832**	0.61	0.43	0.45	0.35	**0.30**
Sofa	0.872	0.898	0.893	0.926	**0.938**	0.69	0.52	0.53	0.42	**0.38**
Table	0.758	0.813	0.794	0.868	**0.894**	0.72	0.52	0.57	0.42	**0.37**
Telephone	0.916	0.939	0.927	0.952	**0.960**	0.41	0.31	0.33	0.27	**0.25**
Vessel	0.748	0.797	0.795	0.846	**0.872**	0.85	0.63	0.60	0.47	**0.40**
mean	0.778	0.821	0.808	0.863	**0.882**	0.80	0.59	0.61	0.47	**0.43**

Method	NC ↑					F-Score ↑				
	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)	ONet [7]	ConvONet [8]	POCO [1]	ALTO [10]	DITTO (ours)
Airplane	0.897	0.901	0.867	0.914	**0.931**	0.864	0.902	0.867	0.938	**0.962**
Bench	0.878	0.886	0.864	0.906	**0.920**	0.860	0.912	0.882	0.947	**0.966**
Cabinet	0.916	0.931	0.917	0.943	**0.953**	0.856	0.916	0.896	0.943	**0.957**
Car	0.875	0.864	0.835	0.873	**0.887**	0.757	0.810	0.766	0.850	**0.879**
Chair	0.889	0.905	0.885	0.923	**0.940**	0.754	0.850	0.833	0.910	**0.941**
Display	0.926	0.947	0.938	0.956	**0.964**	0.813	0.926	0.916	0.957	**0.967**
Lamp	0.813	0.853	0.834	0.875	**0.902**	0.618	0.771	0.781	0.857	**0.908**
Loudspeaker	0.897	0.911	0.897	0.916	**0.932**	0.737	0.832	0.819	0.871	**0.899**
Rifle	0.863	0.890	0.883	0.909	**0.925**	0.838	0.919	0.918	0.952	**0.968**
Sofa	0.928	0.935	0.924	0.946	**0.956**	0.846	0.906	0.899	0.941	**0.956**
Table	0.917	0.933	0.917	0.945	**0.957**	0.839	0.913	0.894	0.947	**0.966**
Telephone	0.970	0.975	0.970	0.978	**0.982**	0.942	0.975	0.971	0.984	**0.988**
Vessel	0.860	0.879	0.867	0.898	**0.914**	0.758	0.850	0.851	0.909	**0.935**
mean	0.895	0.908	0.892	0.922	**0.931**	0.806	0.883	0.869	0.924	**0.940**

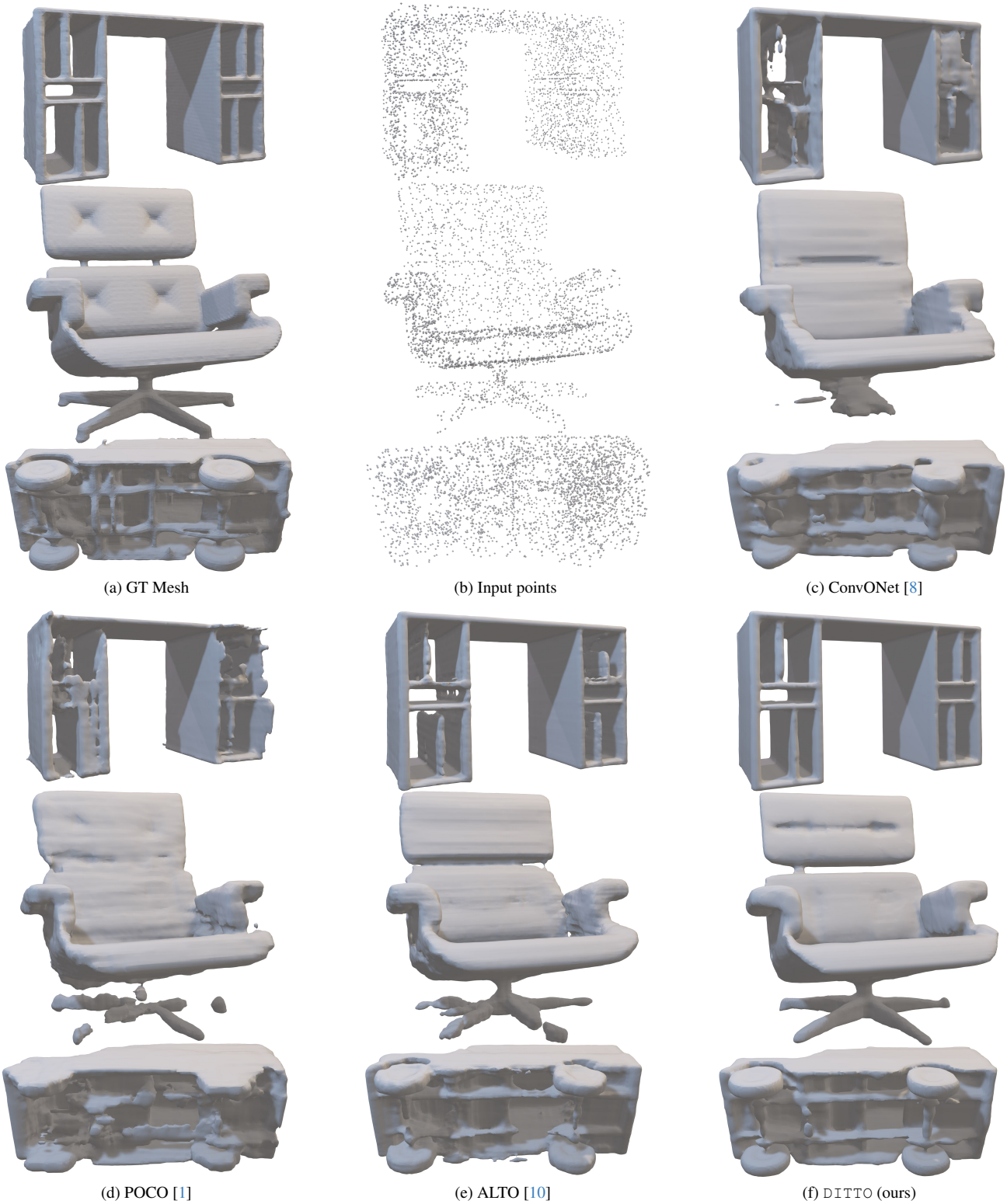


(a) GT Mesh (b) Input points (c) ConvONet [8]



(d) POCO [1] (e) ALTO [10] (f) DITTO (ours)

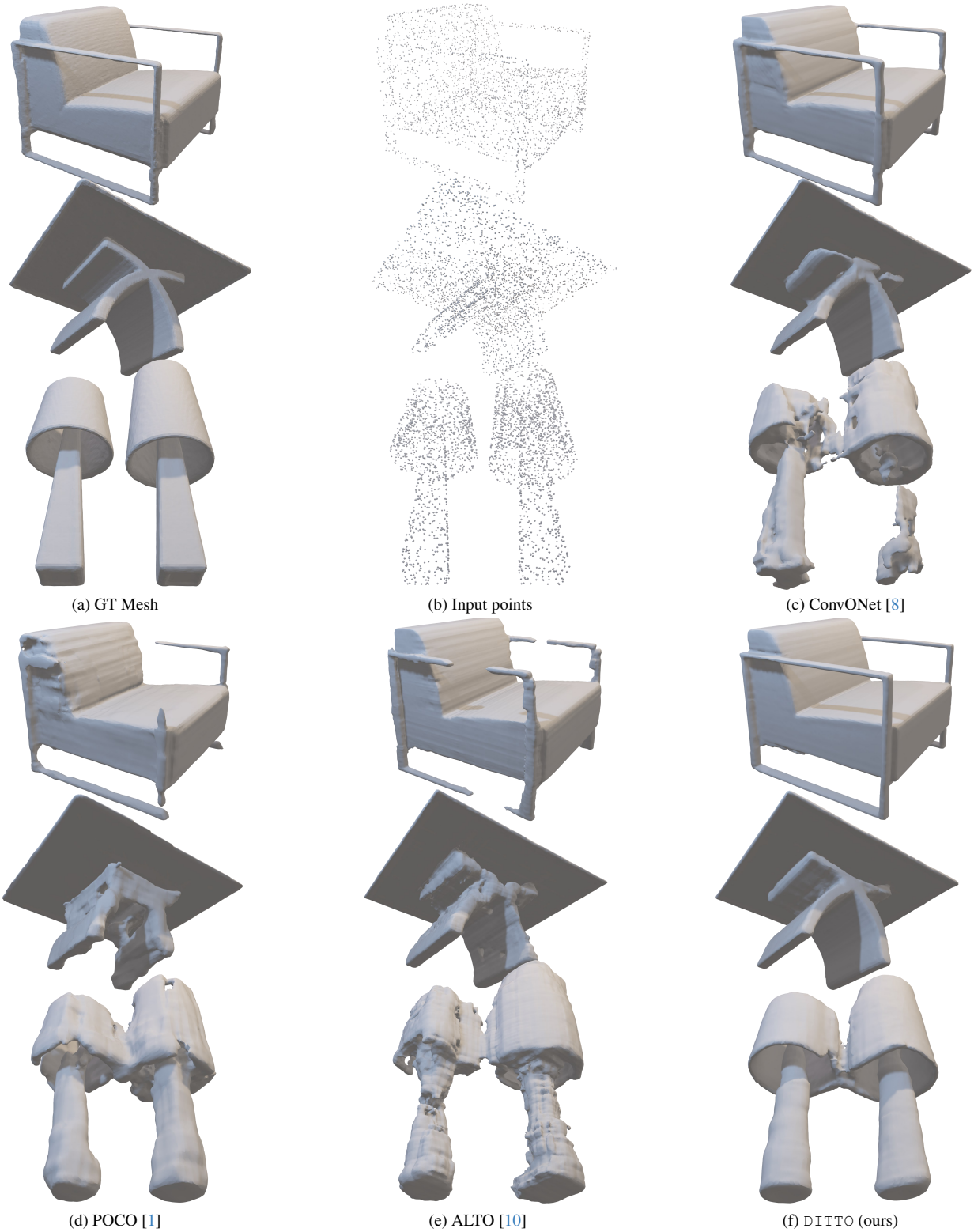
Figure 4. **Object-level 3D reconstruction comparison on ShapeNet [3] with 3K input points and noise level 0.005**.



(a) GT Mesh (b) Input points (c) ConvONet [8]

(d) POCO [1] (e) ALTO [10] (f) DITTO (ours)

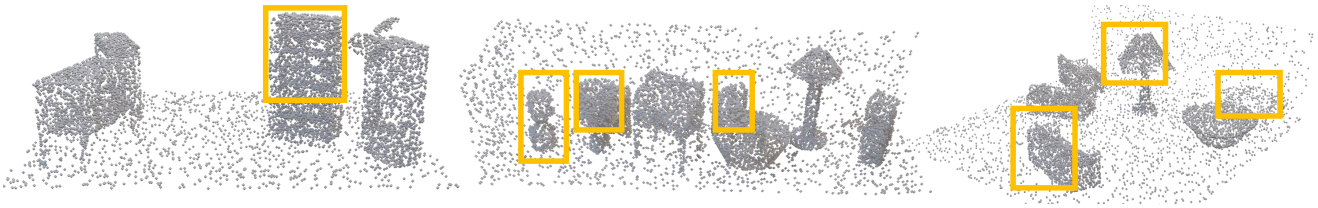
Figure 5. **Object-level 3D reconstruction comparison on ShapeNet [3] with 1K input points and noise level 0.005**.



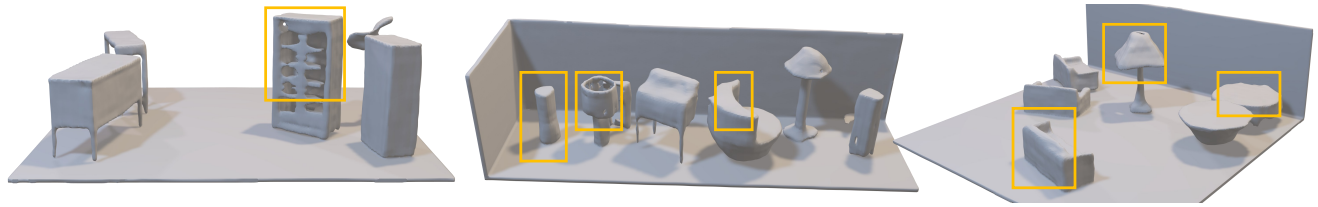
(a) GT Mesh (b) Input points (c) ConvONet [8]

(d) POCO [1] (e) ALTO [10] (f) DITTO (ours)

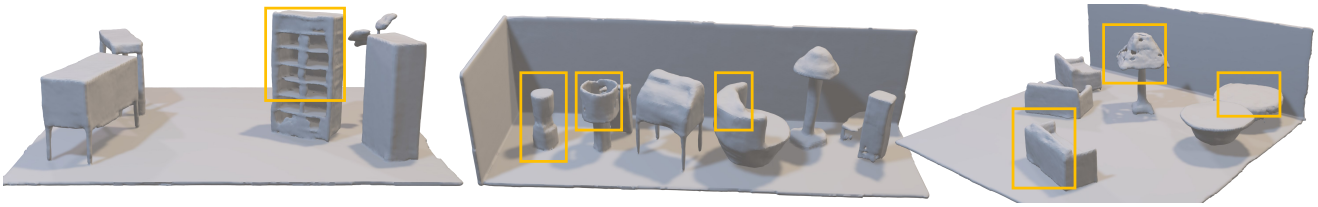
Figure 6. **Object-level 3D reconstruction comparison on ShapeNet [3] with 0.3K input points and noise level 0.005**.



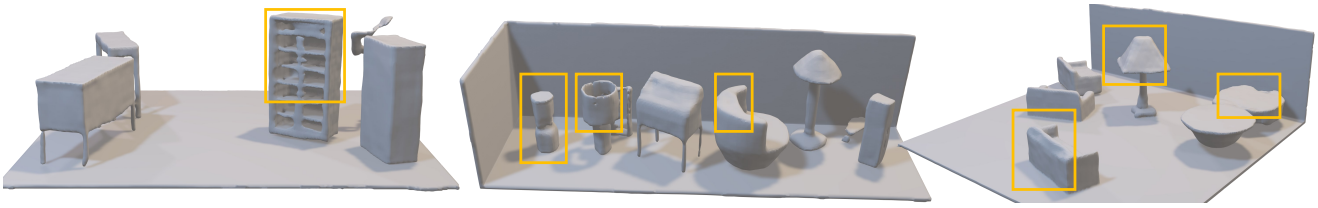
(a) Input points



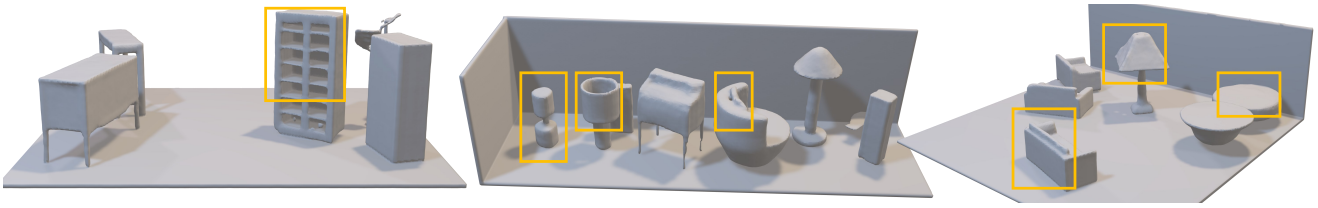
(b) ConvONet [8]



(c) POCO [1]

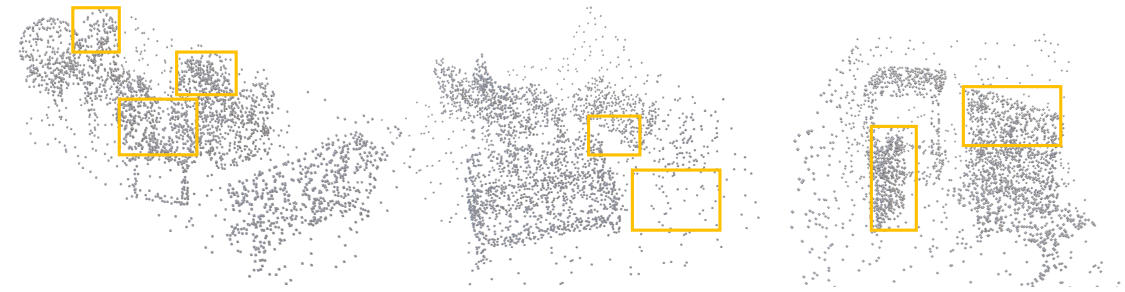


(d) ALTO [10]

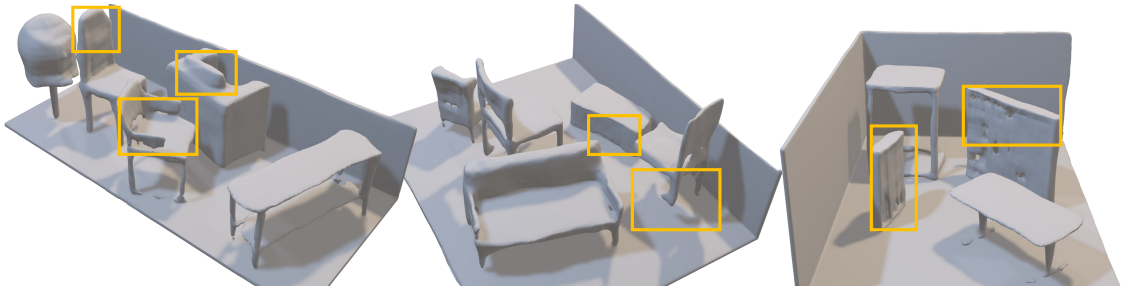


(e) DITTO (ours)

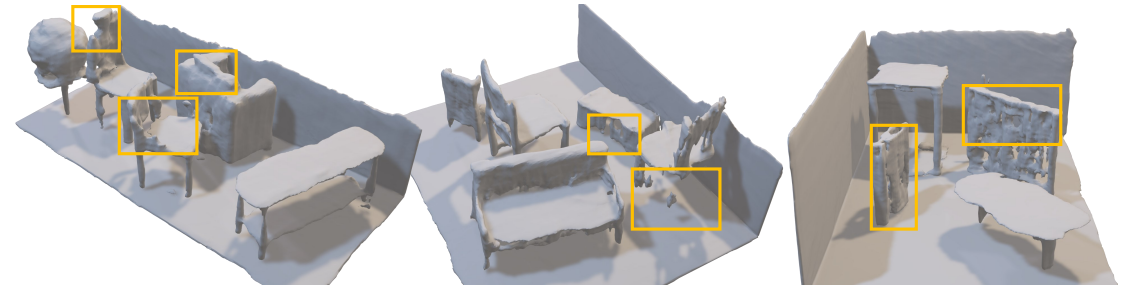
Figure 7. **Scene-level 3D reconstruction results on the Synthetic Rooms dataset [8] with 10K input points and noise level 0.005**.



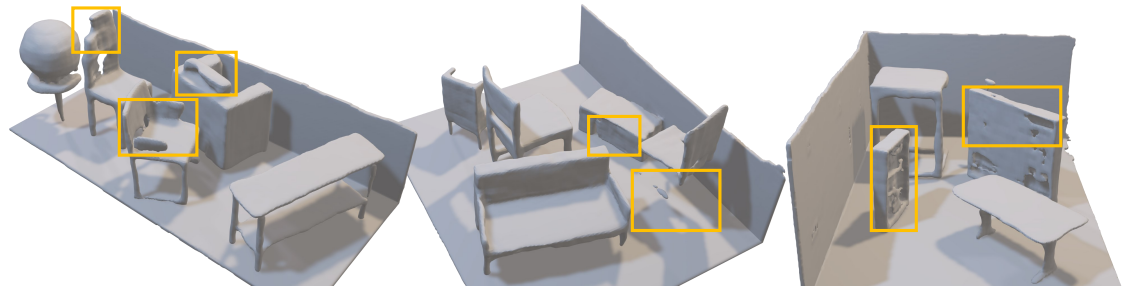
(a) Input points



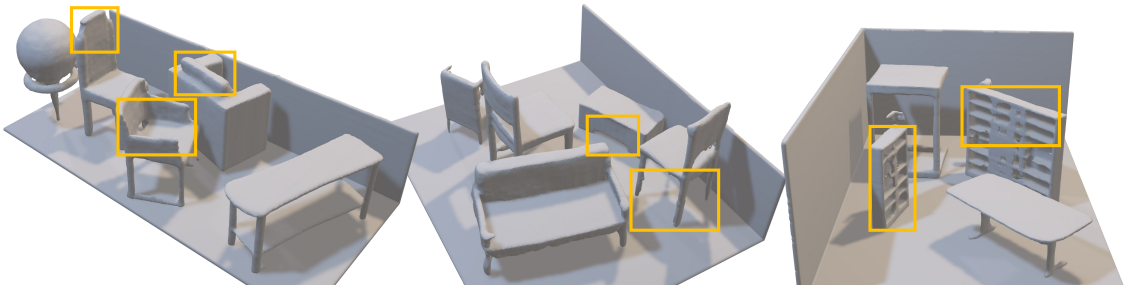
(b) ConvONet [8]



(c) POCO [1]



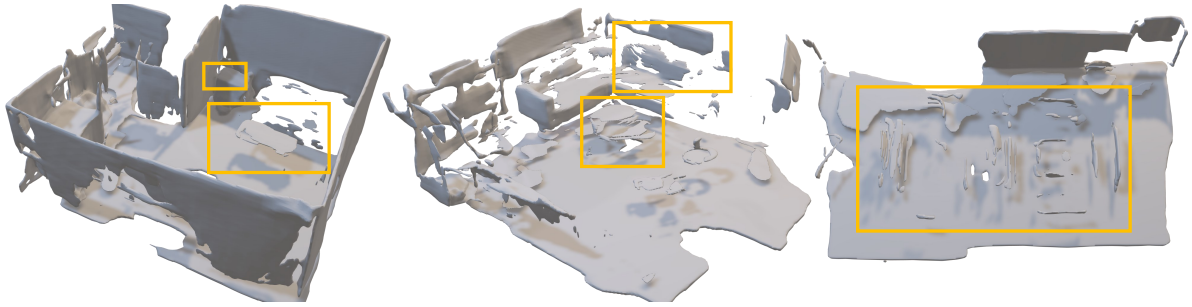
(d) ALTO [10]



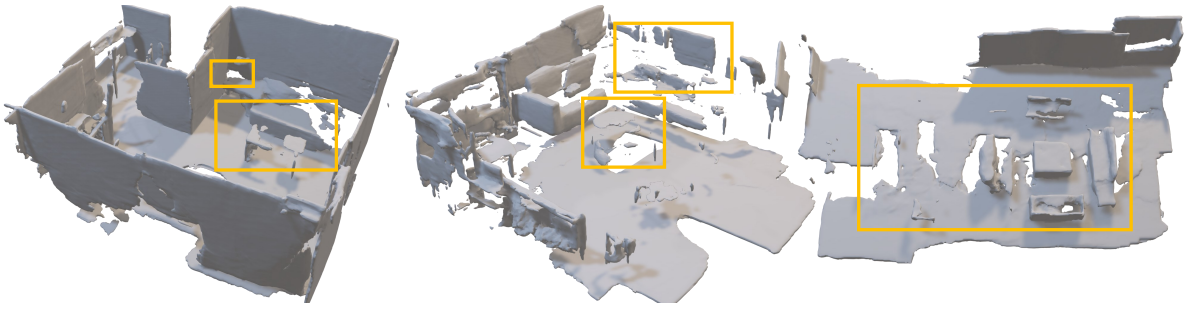
(e) DITTO (ours)

Figure 8. **Scene-level 3D reconstruction results on the Synthetic Rooms dataset [8] with 3K input points and noise level 0.005**.

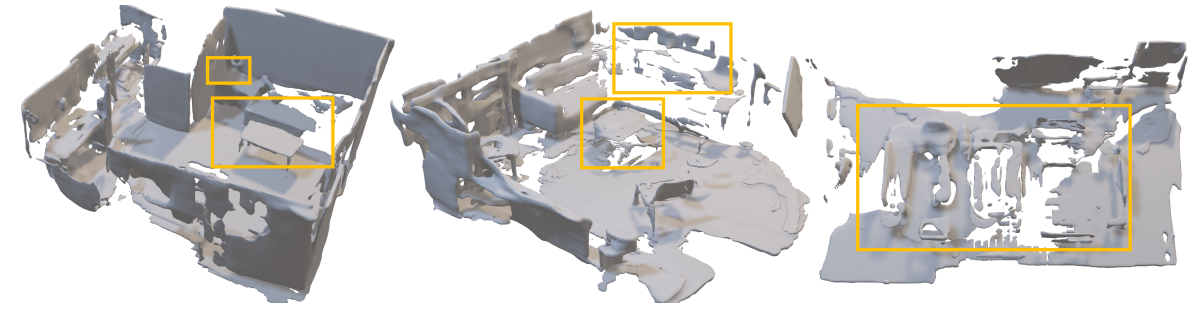
(a) Input points



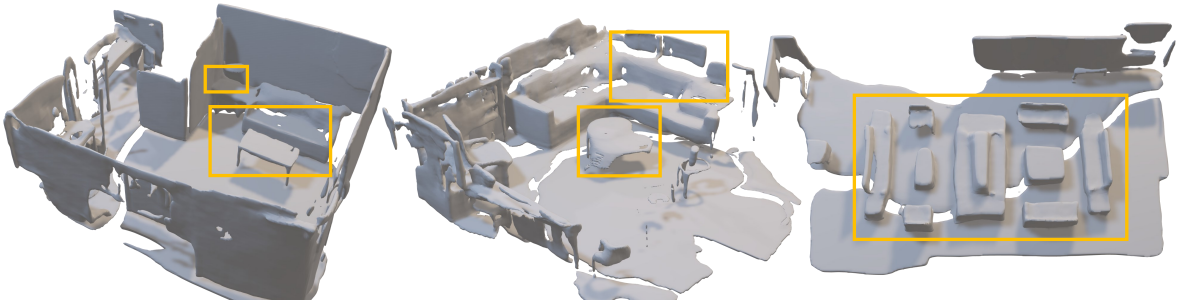
(b) ConvONet [8]



(c) POCO [1]



(d) ALTO [10]



(e) DITTO (ours)

Figure 9. **Scene-level 3D reconstruction results on the ScanNet-v2 dataset [5].**

References

[1] Alexandre Boulch and Renaud Marlet. Poco: Point convolution for surface reconstruction. In *CVPR*, 2022. 1, 3, 6, 7, 8, 9, 10, 11, 12, 13, 14

[2] Alexandre Boulch, Gilles Puy, and Renaud Marlet. Fkaconv: Feature-kernel alignment for point cloud convolution. In *ACCV*, 2020. 1

[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. 3, 9, 10, 11

[4] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, 2016. 2

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 4, 14

[6] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *NeurIPS*, 2019. 1

[7] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 6, 7, 8

[8] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 4, 6, 7, 8, 9, 10, 11, 12, 13, 14

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 2

[10] Zhen Wang, Shijie Zhou, Jeong Joon Park, Despoina Paschalidou, Suya You, Gordon Wetzstein, Leonidas Guibas, and Achuta Kadambi. Alto: Alternating latent topologies for implicit 3d reconstruction. In *CVPR*, 2023. 1, 2, 3, 6, 7, 8, 9, 10, 11, 12, 13, 14