Supplementary Materials
3DILG: Irregular Latent Grids for 3D Generative Modeling

Biao Zhang
KAUST
biao.zhang@kaust.edu.sa

Matthias Nießner
Technical University of Munich
niessner@tum.de

Peter Wonka
KAUST
pwonka@gmail.com

A Appendix

A.1 Hardware and implementation

We train all models on 4 A100 GPUs if not specified otherwise. For shape autoencoding, the number of training epochs is 800. For autoregressive models, the number of epochs is 400. Other implementation details can be found in the accompanying code (`https://github.com/1zb/3DILG`).

A.2 Shape reconstruction

More results. We show extended results for the shape reconstruction task in Table. 1, 2, and 3.

Point Patch Size. We investigate how the results (averaged over all categories on ShapeNet-v2) differ with different values of K. The results are shown in Table 4. The best results are achieved when $K = 16$. However, when $8 \leq K \leq 64$, the results are very close. We choose $K = 32$ in other experiments because we want to cover as large an area of the point cloud as possible when $M = 64$. Another interesting observation is that even with $K = 1$ we can achieve good results. In this case, each point patch only contains one point and many points are ignored and not considered in the representation if $N > M$ (the typical case). This is equivalent to reconstructing shapes with 512 points.

Different Patch Sizes when Training and Testing. The value of M can be different when training and testing, which we denote as M_{train} and M_{test}. In the training phase, we set M_{train}, while in the testing phase, we use M_{test}. The metrics can be found in Fig. 1. M_{train} is taken from the set $\{64, 128, 256, 512\}$ and M_{test} is taken from the set $\{64, 128, 256, 512, 1024\}$. We find that, generally results of $M_{test} > M_{train}$ are better than results of $M_{test} = M_{train}$. Results are much worse when $M_{test} < M_{train}$.

A.3 Scene reconstruction

The dataset contains indoor room scenes with different objects from ShapeNet. We preprocess the rooms in the same way as we did for ShapeNet. Other details about the dataset can be found in [4].

We show scene level reconstruction results in Table 5. Usually scenes are more complex than objects and require more points to maintain their structures. We find that our method performs well on very small input point clouds ($N = 2048$). However, for ConvONet, in order to be comparable with our results, the grid resolution needs to be increased to 64^3 and the point cloud size needs to be increased to $N = 16384$.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

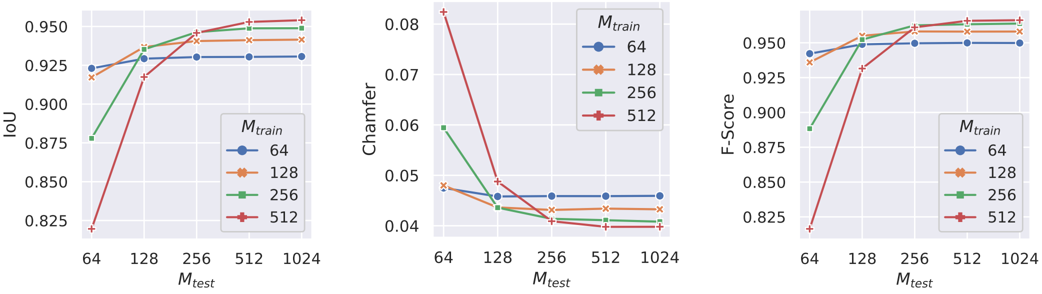


Figure 1: **Patch set sizes.** We show results when M_{train} and M_{test} are different.

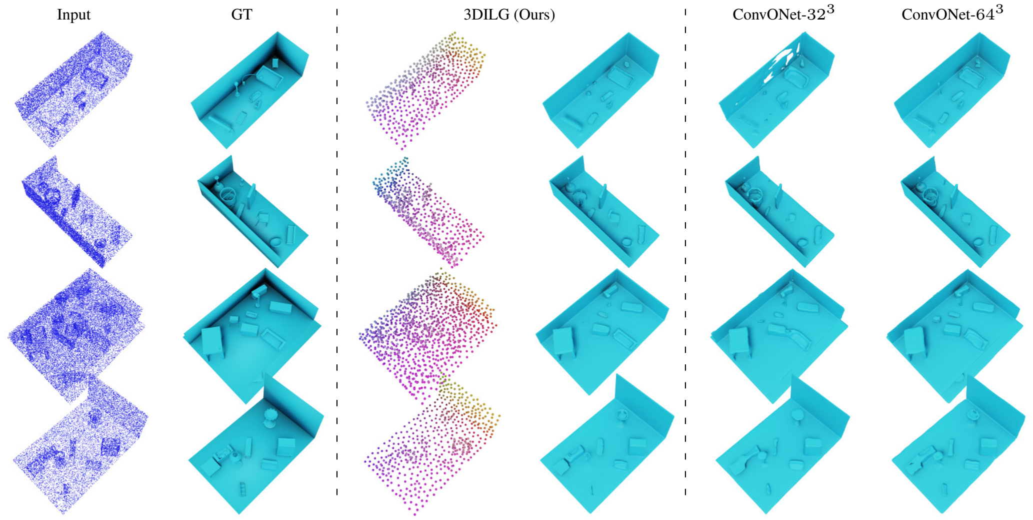


Figure 2: **Scene reconstruction.** We train our model on Synthetic Rooms [4]. We set $N = 16384$, $M = 512$, and $K = 32$. We compare our results with ConvONet of resolution 32^3 and 64^3. The column **Input** shows input point clouds of size 16384. The column **GT** shows ground-truth meshes. We compare our results with ConvONet of resolution 32^3 and 64^3. We also show $\{\mathbf{x}_i\}_{i\in\mathcal{M}}$ obtained via Farthest Point Sampling.

A.4 Limitation

When applied to 3d reconstruction, our method is not sensitive to

1. patch size K according to Table 4,
2. input point cloud size N according to Table 5.

Both experiments hint that we have not fully used information of all points $\{\mathbf{x}_i\}_{i\in\mathcal{N}}$ in the reconstruction. The performance is mainly decided by the choice of M (sub-sampled point cloud size). We would like to investigate this phenomena in future work and think that our results can be even further improved by a different pre-process constructing the patches.

A.5 More generative results on ShapeNet

More results on low-resolution-image-conditioned generation. We show more image-conditioned generation results in Fig. 3.

Mask-conditioned generation. Letting the context $\mathcal{C}$ be a binary mask. We show the conditional generation results in Fig. 4. Comparing to Fig. 3, the task is more constrained and leaves less space for imagination. Thus, the diversity of generated samples is limited.

Perceptual study on category-conditioned generation. We conduct a perceptual study on the quality of generated samples in category-conditioned generation. We show two samples and ask users

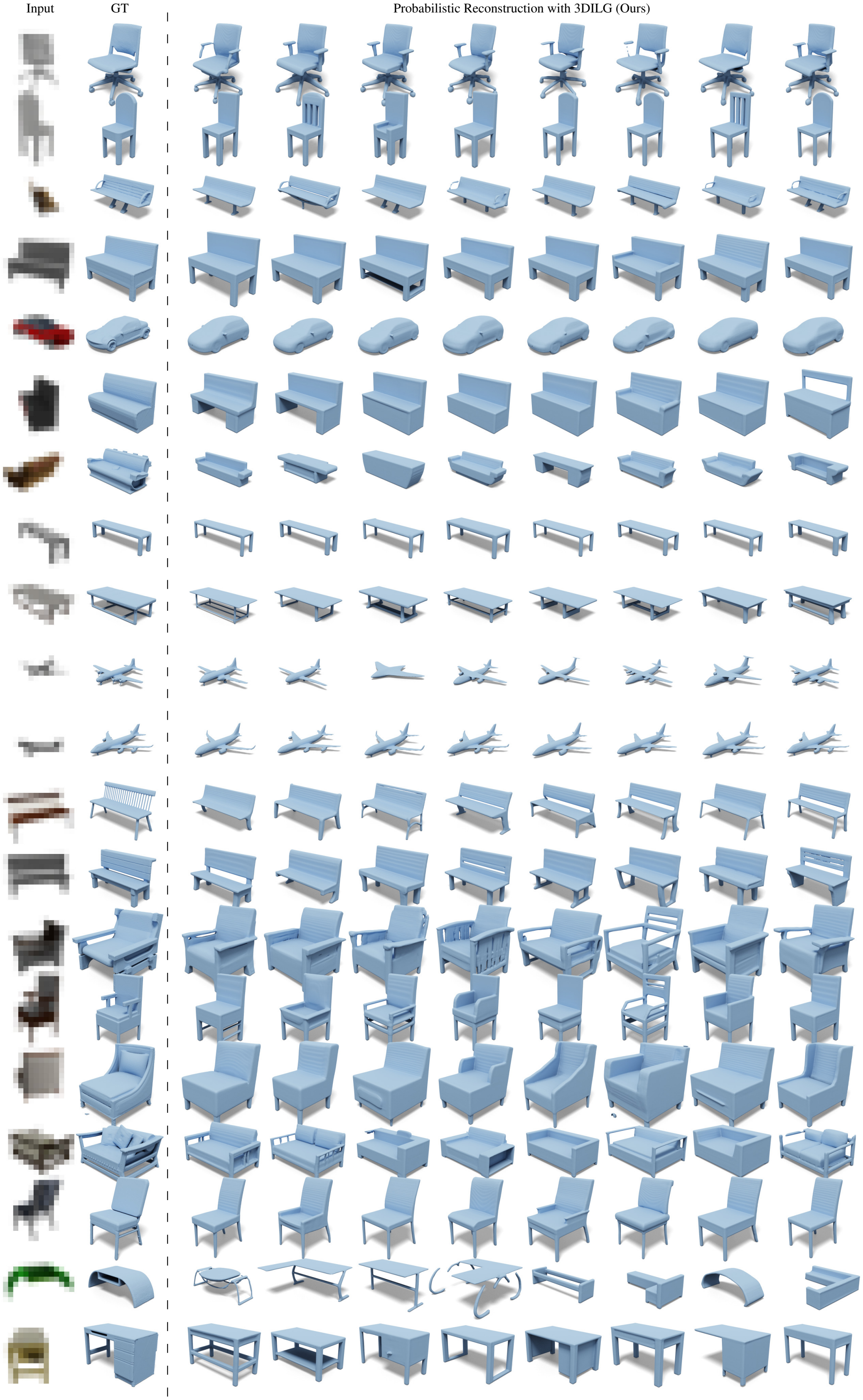


Figure 3: **Image-conditioned generation** (16×16)**.** We sample 8 shapes for each input image.

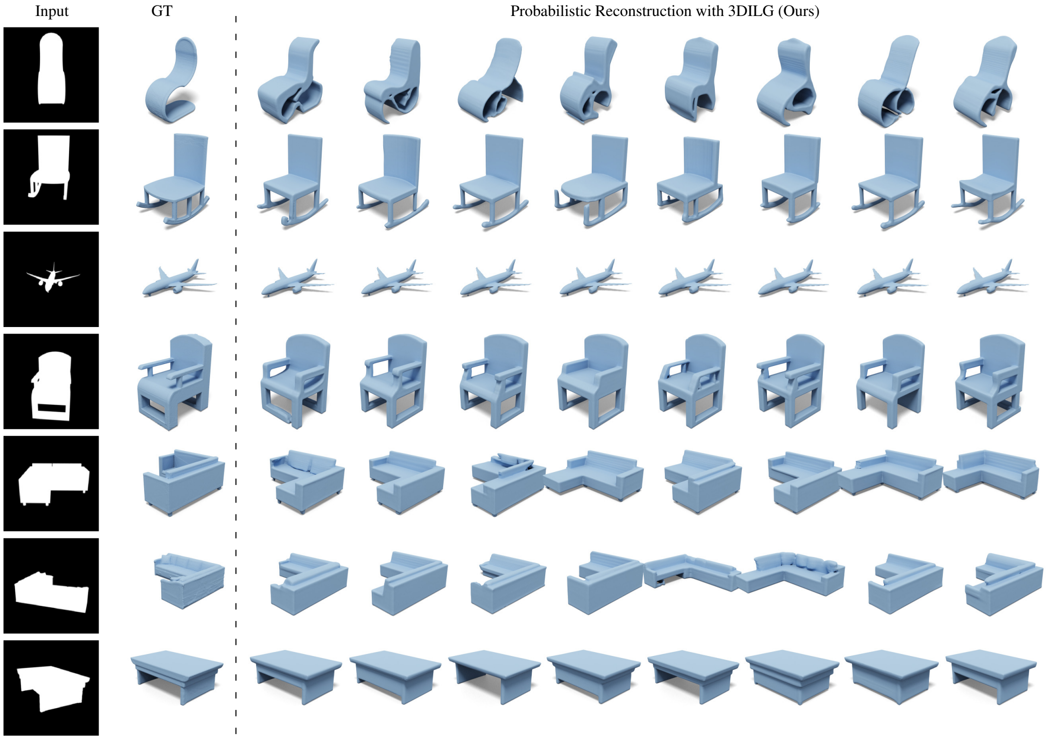


Figure 4: **Mask-conditioned generation.** We sample 8 shapes for each input mask.

to pick the one with higher quality (see a snapshot in Fig. 5). This survey can be finished in less than 10 minutes. We collected 778 pairwise comparisons from 10 users. Results can be found in Table 6. When comparing to the ground-truth test sets, the probability of choosing our results (0.2365) are higher than Grid-8^3 (0.1396).

More results on category-conditioned generation. We show generated samples on more categories in Fig. 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. We compare our results with Grid-8^3. Bad samples are highlighted in red.

A.6 Real world image conditioned generation

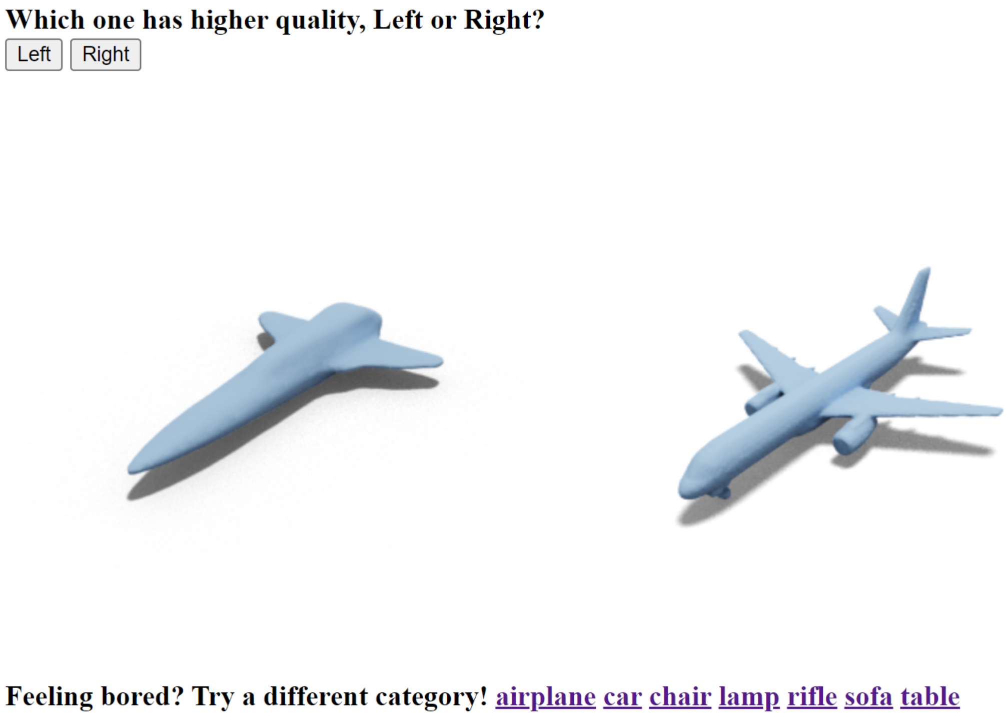
We take real world images from the dataset ABO [2] to show the *generalization* ability of our method. The results can be found in Fig. 16. Note that these images contain real textures which are often missing in rendered ShapeNet objects. We can see that OccNet often predicts blurred objects.

A.7 Real world point cloud surface reconstruction

Here we test the *generalization* ability of our surface reconstruction network. We choose 129 human meshes from the dataset D-FAUST [1]. We evaluate IF-Net and our method on these meshes. Both networks are trained on ShapeNet. The results can be found in Table 7. The reconstruction meshes are shown in Fig. 17.

A.8 Generative results on ABO

Here we show additional results on the dataset ABO [2]. We select 1184 samples of *chairs* from the dataset. We split these samples to train/val/test sets (90%/5%/5%). The generated samples are shown in Fig. 18.



Which one has higher quality, Left or Right?

Feeling bored? Try a different category! airplane car chair lamp rifle sofa table

Figure 5: Interface of user study.

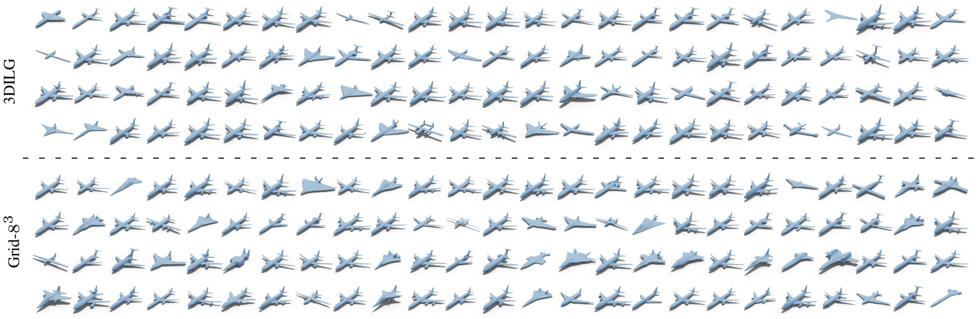


Figure 6: Generated samples on the category *airplane*.

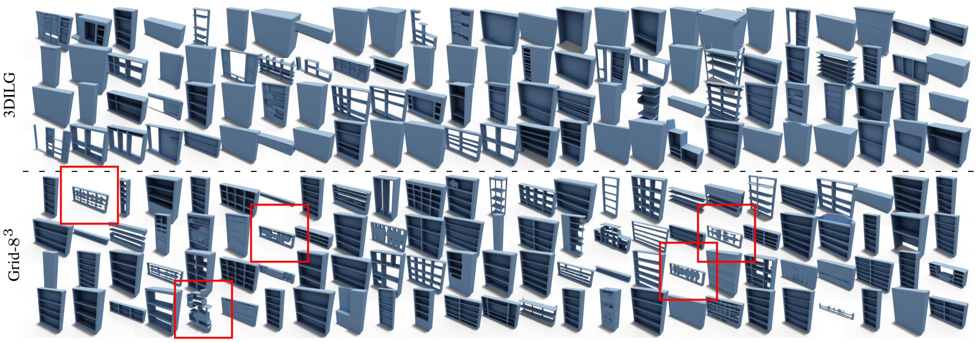


Figure 7: Generated samples on the category *bookshelf*.

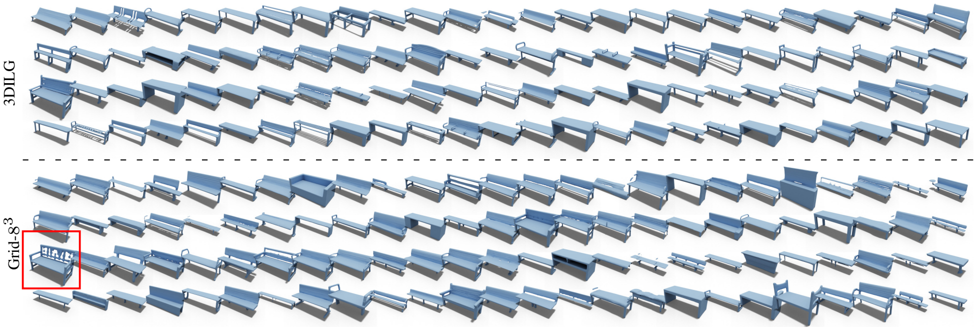


Figure 8: Generated samples on the category *bench*.

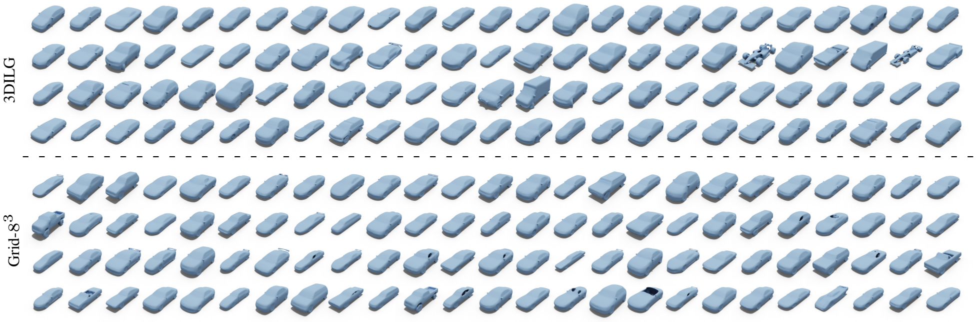


Figure 9: Generated samples on the category *car*.

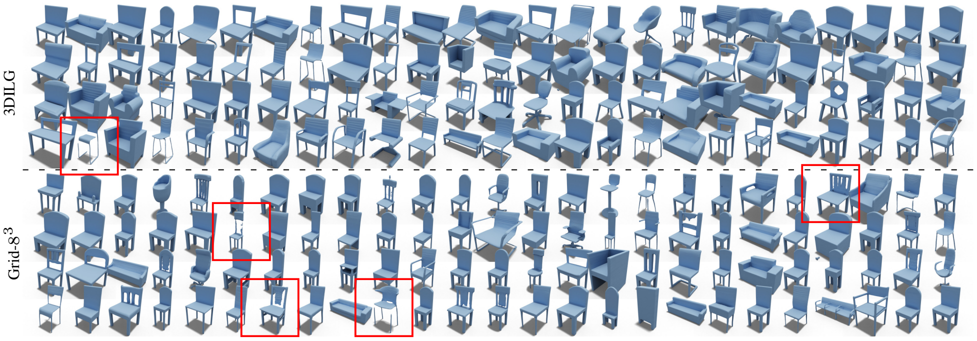


Figure 10: Generated samples on the category *chair*.

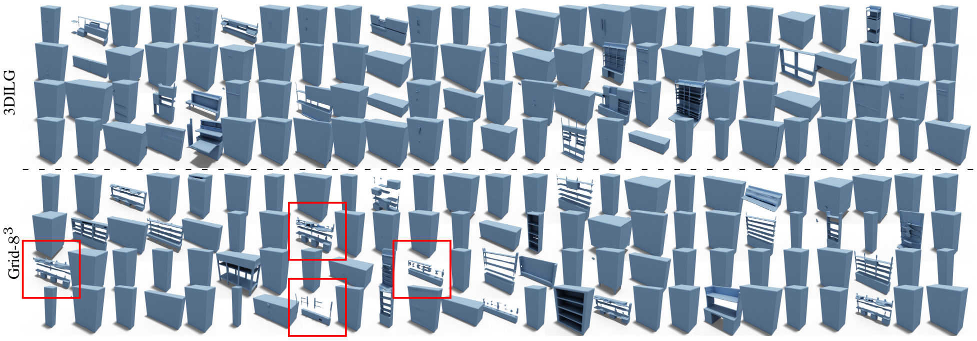


Figure 11: Generated samples on the category *file cabinet*.

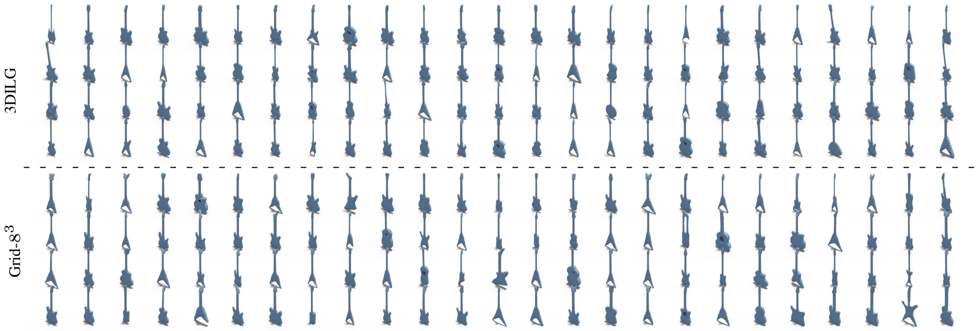


Figure 12: Generated samples on the category *guitar*.

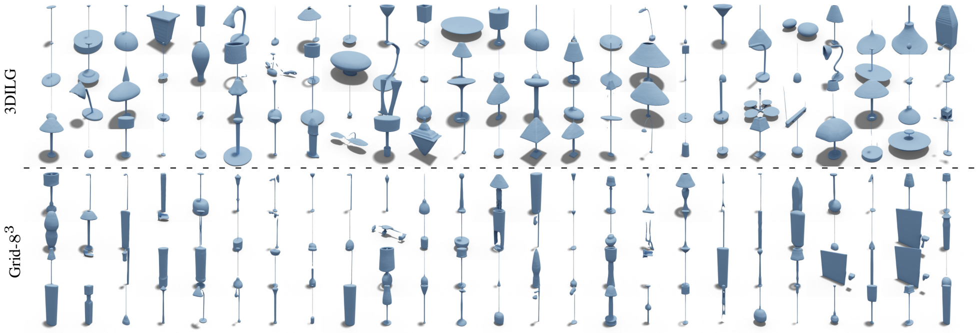


Figure 13: Generated samples on the category *lamp*.

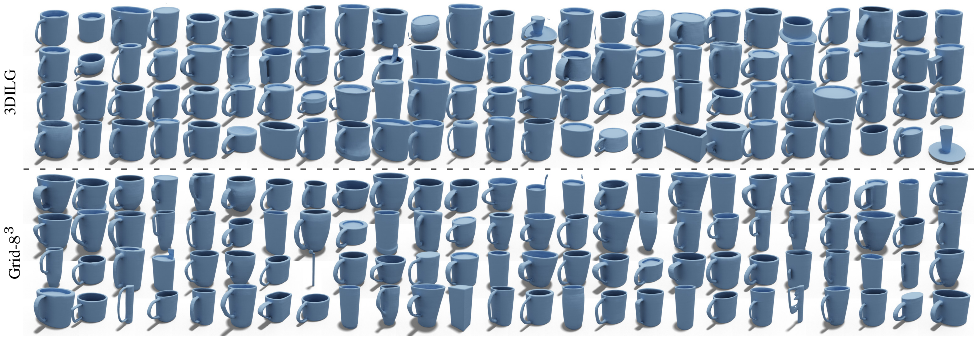


Figure 14: Generated samples on the category *mug*.

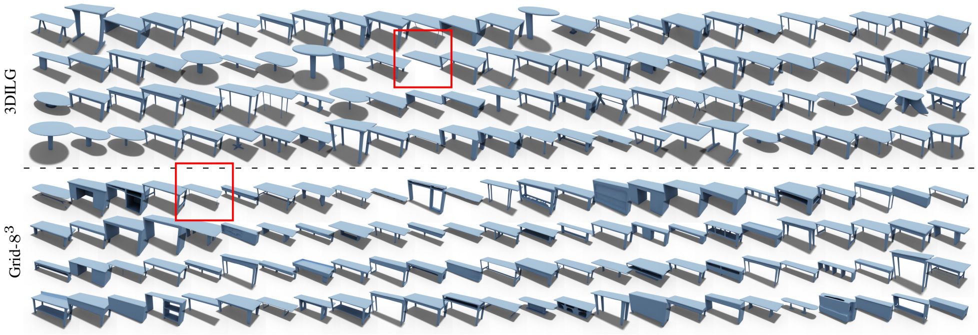


Figure 15: Generated samples on the category *table*.

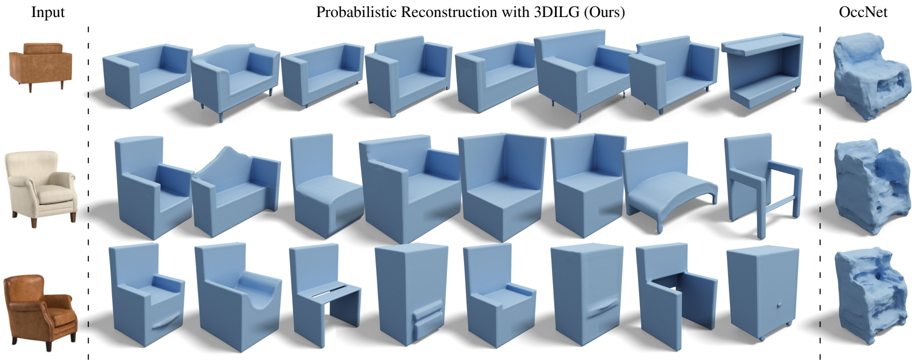


Figure 16: Real world image conditioned generation.

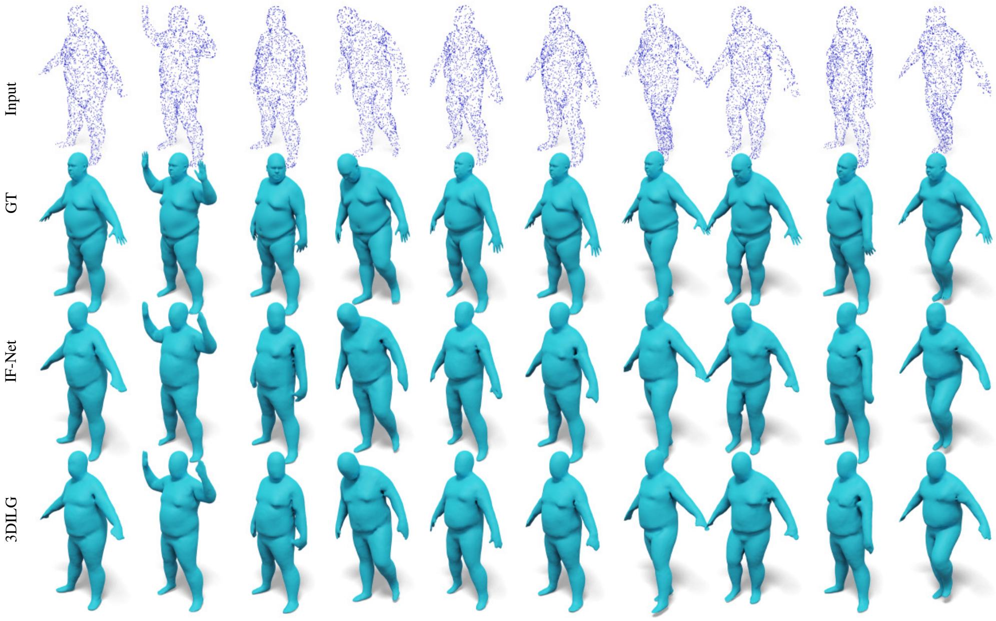


Figure 17: Reconstruction results on D-FAUST

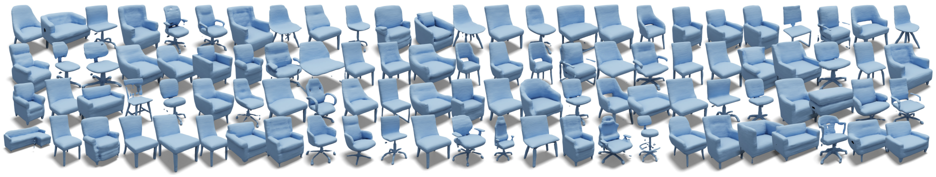


Figure 18: Generated samples trained on ABO.

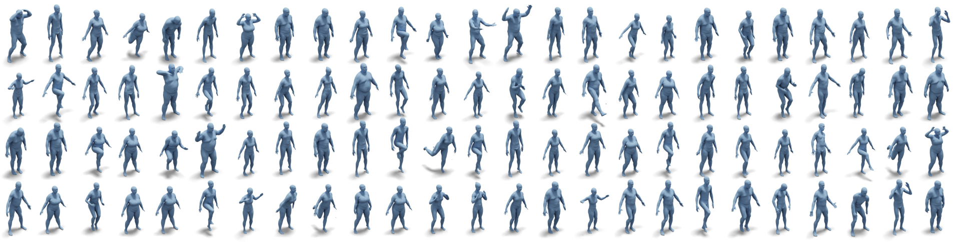


Figure 19: Generated samples trained on D-FAUST.

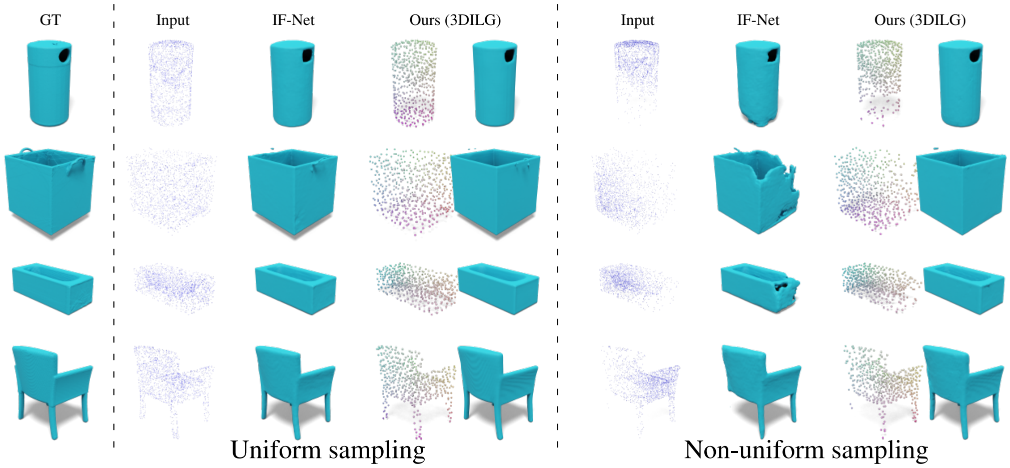


Figure 20: **Uniform v.s. non-uniform sampling**

A.9 Generative results on D-FAUST

Similarly we train a unconditioned generative models on D-FAUST. The generated samples are shown in Fig. 19.

A.10 Non-uniform density sampling

In our main paper, the input point clouds are sampled from shape surfaces. We assumed each point on an surface have the same probability to be sampled. Here, we consider a different sampling strategy. We randomly choose an "anchor" point $\mathbf{x}_0$ on an shape surface. We want points near the "anchor" point have high probabilities to be sampled. The probabilities are defined by a Gaussian function

$$p(\mathbf{x}, \mathbf{x}_0) = \exp(-\beta \cdot \operatorname{dist}(\mathbf{x}, \mathbf{x}_0)), \tag{1}$$

Here small β gives rise to uniform sampling, while large β assigns large sampling probability near the "anchor" point (in an extreme case, some areas on the surface will never be sampled from). We use this function to simulate an non-uniform density sampling on the surface. We train IF-Net and our method with this sampling strategy. The results when $\beta = 1$ can be found in Table 8. Also see Fig. 20 for visualizations of reconstructed meshes. It is evidently that IF-Net is unable to reconstruct correct meshes where the point sampling density is low.

A.11 Poisson surface reconstruction

We run Poisson surface reconstruction (PSR) [3] on ShapeNet. We compare other neural-network-based methods with PSR. The results can be found in Table 9. Note that PSR requires per-point normals as input. In the terms of all metrics, PSR performs worse than IF-Net and our method. However, it is comparable with ConvOccNet.

References

[1] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J Black. Dynamic faust: Registering human bodies in motion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6233–6242, 2017. (Cited on 4)

[2] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21126–21136, 2022. (Cited on 4)

[3] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. (Cited on 9)

[4] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European Conference on Computer Vision*, pages 523–540. Springer, 2020. (Cited on 1, 2, 14)

Table 1: **IoU** ↑

	OccNet	ConvONet	IF-Net	3DILG (Proposed)			
				$M = 64$	$M = 128$	$M = 256$	$M = 512$
airplane	0.835	0.881	0.937	0.926	0.938	0.948	**0.952**
trash bin	0.862	0.871	0.918	0.909	0.920	0.926	**0.933**
bag	0.785	0.946	0.970	0.961	0.971	0.977	**0.981**
basket	0.733	0.801	0.889	0.903	0.912	0.914	**0.922**
bathtub	0.865	0.871	0.933	0.922	0.935	0.947	**0.953**
bed	0.722	0.823	0.910	0.888	0.910	0.928	**0.939**
bench	0.775	0.815	0.891	0.888	0.910	0.921	**0.931**
birdhouse	0.791	0.908	0.926	0.904	0.915	0.918	**0.932**
bookshelf	0.602	0.806	0.890	0.876	0.909	0.927	**0.940**
bottle	0.905	0.938	0.963	0.950	0.961	0.967	**0.969**
bowl	0.894	0.905	0.943	0.936	0.947	0.952	**0.958**
bus	0.923	0.933	0.950	0.962	0.964	0.969	**0.973**
cabinet	0.918	0.920	0.959	0.962	0.971	0.976	**0.979**
camera	0.851	0.945	0.974	0.964	0.974	0.980	**0.984**
can	0.979	0.979	0.989	0.986	0.989	0.991	**0.992**
cap	0.789	0.832	0.896	0.888	0.909	0.920	**0.928**
car	0.911	0.921	0.952	0.941	0.948	0.955	**0.961**
cellphone	0.935	0.947	0.958	0.983	0.985	0.988	**0.989**
chair	0.803	0.856	0.927	0.908	0.926	0.940	**0.950**
clock	0.871	0.911	0.945	0.956	0.963	0.970	**0.973**
keyboard	0.777	0.872	0.929	0.926	0.941	0.947	**0.952**
dishwasher	0.982	0.982	0.985	0.993	0.995	0.995	**0.995**
display	0.853	0.901	0.953	0.956	0.969	0.975	**0.978**
earphone	0.554	0.843	0.913	0.835	0.869	0.900	**0.914**
faucet	0.721	0.864	0.931	0.889	0.909	0.925	**0.936**
file cabinet	0.882	0.930	0.963	0.962	0.973	0.981	**0.983**
guitar	0.809	0.877	0.926	0.924	0.937	0.947	**0.953**
helmet	0.705	0.834	0.902	0.841	0.868	0.891	**0.908**
jar	0.737	0.873	0.913	0.886	0.901	0.912	**0.922**
knife	0.768	0.877	0.917	0.901	0.921	0.930	**0.938**
lamp	0.735	0.859	0.914	0.887	0.906	0.916	**0.926**
laptop	0.862	0.861	0.925	0.954	0.962	0.967	**0.971**
loudspeaker	0.846	0.916	0.943	0.928	0.943	0.954	**0.958**
mailbox	0.909	0.946	0.969	0.970	0.976	0.983	**0.984**
microphone	0.959	0.965	0.979	0.983	0.986	0.987	**0.990**
microwaves	0.981	0.976	0.984	0.994	0.995	0.996	**0.996**
motorbike	0.603	0.729	0.827	0.717	0.760	0.794	**0.812**
mug	0.863	0.898	0.933	0.929	0.936	0.942	**0.953**
piano	0.748	0.864	0.924	0.897	0.922	0.937	**0.941**
pillow	0.916	0.946	0.968	0.960	0.965	0.969	**0.971**
pistol	0.848	0.913	0.944	0.933	0.947	0.954	**0.960**
flowerpot	0.754	0.784	0.843	0.833	0.852	0.863	**0.876**
printer	0.878	0.934	0.962	0.955	0.967	0.973	**0.978**
remote	0.884	0.930	0.964	0.959	0.968	0.978	**0.979**
rifle	0.755	0.871	0.914	0.898	0.919	0.927	**0.938**
rocket	0.743	0.846	0.930	0.885	0.904	0.922	**0.933**
skateboard	0.756	0.770	0.884	0.886	0.905	0.926	**0.932**
sofa	0.894	0.930	0.960	0.953	0.963	0.971	**0.975**
stove	0.873	0.914	0.950	0.936	0.942	0.954	**0.963**
table	0.823	0.847	0.901	0.943	0.954	0.960	**0.963**
telephone	0.887	0.931	0.951	0.960	0.970	0.976	**0.979**
tower	0.775	0.829	0.889	0.873	0.903	0.913	**0.930**
train	0.840	0.889	0.926	0.917	0.930	0.939	**0.947**
watercraft	0.760	0.873	0.932	0.904	0.926	0.939	**0.946**
washer	0.961	0.967	0.980	0.984	0.985	0.990	**0.991**
mean	0.825	0.888	0.934	0.923	0.937	0.946	**0.953**

Table 2: **Chamfer** ↓

	OccNet	ConvONet	IF-Net	3DILG (Proposed)			
				$M = 64$	$M = 128$	$M = 256$	$M = 512$
airplane	0.037	0.028	0.020	0.023	0.021	0.020	**0.019**
trash bin	0.098	0.107	**0.093**	0.098	0.100	0.095	**0.093**
bag	0.091	0.073	**0.059**	0.069	0.064	0.061	0.060
basket	0.078	0.051	0.039	0.039	0.038	0.039	**0.037**
bathtub	0.043	0.044	0.032	0.034	0.032	0.031	**0.030**
bed	0.094	0.057	**0.038**	0.047	0.044	0.042	**0.038**
bench	0.045	0.034	0.025	0.027	0.024	0.023	**0.022**
birdhouse	0.183	0.158	**0.125**	0.138	0.133	0.147	0.134
bookshelf	0.105	0.044	0.033	0.037	0.032	0.030	**0.029**
bottle	0.069	0.050	**0.039**	0.047	0.044	0.043	0.040
bowl	0.057	0.048	0.040	0.042	0.040	0.039	**0.038**
bus	0.048	0.044	0.036	0.038	0.034	0.034	**0.034**
cabinet	0.062	0.060	0.045	0.049	0.045	0.044	**0.043**
camera	0.118	0.052	0.035	0.044	0.038	0.035	**0.032**
can	0.047	0.047	**0.038**	0.046	0.041	0.039	**0.039**
cap	0.041	0.036	0.029	0.031	0.029	0.028	**0.027**
car	0.082	0.083	0.067	0.076	0.072	0.068	**0.066**
cellphone	0.032	0.030	0.026	0.024	0.023	0.023	**0.023**
chair	0.058	0.044	0.031	0.036	0.032	0.030	**0.029**
clock	0.060	0.047	0.035	0.037	0.036	0.033	**0.033**
keyboard	0.054	0.038	0.030	0.032	0.031	0.024	**0.026**
dishwasher	0.089	0.083	**0.082**	0.086	0.085	0.085	0.084
display	0.061	0.053	0.039	0.042	0.040	0.038	**0.038**
earphone	0.178	0.059	0.038	0.069	0.052	0.040	**0.037**
faucet	0.074	0.035	**0.022**	0.031	0.028	0.024	0.023
file cabinet	0.059	0.046	0.034	0.035	0.033	0.031	**0.030**
guitar	0.040	0.030	0.023	0.026	0.023	0.023	**0.022**
helmet	0.091	0.057	**0.043**	0.066	0.055	0.048	0.046
jar	0.140	0.073	**0.060**	0.090	0.076	0.069	0.065
knife	0.036	0.020	0.016	0.019	0.017	0.016	**0.014**
lamp	0.090	0.050	0.038	0.046	0.042	0.039	**0.036**
laptop	0.040	0.041	0.031	0.030	0.029	0.029	**0.029**
loudspeaker	0.082	0.055	0.039	0.047	0.042	0.039	**0.037**
mailbox	0.068	0.058	0.051	0.052	0.049	0.049	**0.047**
microphone	0.029	0.027	0.022	0.022	0.020	0.020	**0.020**
microwaves	0.046	0.052	0.043	0.044	0.043	0.043	**0.042**
motorbike	0.099	0.068	**0.047**	0.072	0.062	0.056	0.052
mug	0.080	0.063	**0.047**	0.053	0.046	0.047	0.048
piano	0.098	0.052	0.040	0.049	0.042	0.040	**0.039**
pillow	0.076	0.054	0.041	0.054	0.046	0.041	**0.038**
pistol	0.053	0.034	0.026	0.031	0.027	0.025	**0.024**
flowerpot	0.131	0.071	**0.056**	0.089	0.084	0.066	0.070
printer	0.119	0.098	0.088	0.095	0.092	0.086	**0.084**
remote	0.056	0.039	0.024	0.029	0.027	0.023	**0.023**
rifle	0.046	0.025	0.018	0.022	0.020	0.018	**0.017**
rocket	0.059	0.035	**0.020**	0.030	0.025	0.023	**0.020**
skateboard	0.034	0.030	0.020	0.022	0.021	0.019	**0.019**
sofa	0.051	0.042	0.032	0.036	0.033	0.031	**0.030**
stove	0.088	0.076	0.062	0.071	0.065	0.067	**0.062**
table	0.041	0.036	0.029	0.028	0.027	0.027	**0.026**
telephone	0.051	0.035	0.028	0.032	0.028	0.027	**0.026**
tower	0.085	0.070	0.054	0.066	0.058	0.056	**0.053**
train	0.065	0.046	0.033	0.041	0.035	0.034	**0.032**
watercraft	0.068	0.036	**0.023**	0.031	0.027	0.024	**0.023**
washer	0.052	0.050	0.042	0.042	0.043	0.039	**0.039**
mean	0.072	0.052	0.041	0.048	0.044	0.041	**0.040**

Table 3: **F-Score** ↑

	OccNet	ConvONet	IF-Net	3DILG (Proposed)			
				$M = 64$	$M = 128$	$M = 256$	$M = 512$
airplane	0.948	0.982	0.994	0.986	0.990	0.992	**0.993**
trash bin	0.830	0.829	**0.879**	0.859	0.867	0.876	0.878
bag	0.769	0.909	**0.938**	0.907	0.927	0.935	**0.938**
basket	0.904	0.952	0.987	0.986	0.988	0.989	**0.992**
bathtub	0.959	0.956	**0.995**	0.987	0.990	0.993	0.994
bed	0.760	0.888	**0.975**	0.930	0.950	0.962	0.973
bench	0.952	0.979	**0.997**	0.987	0.994	0.995	**0.997**
birdhouse	0.626	0.761	**0.837**	0.827	0.834	0.844	0.826
bookshelf	0.756	0.952	0.992	0.967	0.983	0.991	**0.994**
bottle	0.864	0.926	**0.957**	0.935	0.943	0.947	0.954
bowl	0.918	0.963	**0.980**	0.973	0.976	0.979	0.979
bus	0.941	0.956	**0.976**	0.957	0.969	0.971	0.973
cabinet	0.909	0.915	**0.965**	0.944	0.957	0.962	**0.965**
camera	0.667	0.934	0.987	0.937	0.965	0.980	**0.991**
can	0.962	0.954	**0.974**	0.965	0.969	0.971	0.971
cap	0.977	0.980	0.997	0.992	0.999	0.999	**1.000**
car	0.830	0.852	0.888	0.861	0.873	0.882	**0.892**
cellphone	0.990	0.994	**0.995**	0.995	0.995	0.995	**0.995**
chair	0.890	0.943	0.990	0.969	0.982	0.989	**0.992**
clock	0.894	0.959	**0.984**	0.970	0.977	0.983	**0.984**
keyboard	0.912	0.967	0.978	0.970	0.972	0.987	**0.981**
dishwasher	0.926	0.937	0.944	0.939	0.943	0.945	**0.946**
display	0.880	0.934	0.972	0.957	0.967	0.972	**0.973**
earphone	0.495	0.874	0.960	0.858	0.905	0.955	**0.962**
faucet	0.794	0.971	**0.995**	0.969	0.978	0.987	0.991
file cabinet	0.909	0.958	0.990	0.970	0.982	0.991	**0.993**
guitar	0.954	0.975	**0.983**	0.976	0.980	0.980	**0.983**
helmet	0.712	0.908	**0.962**	0.890	0.924	0.947	0.952
jar	0.690	0.870	**0.918**	0.848	0.875	0.890	0.902
knife	0.971	0.996	**1.000**	0.993	0.998	0.999	0.999
lamp	0.820	0.945	0.970	0.949	0.961	0.966	**0.971**
laptop	0.979	0.989	**0.999**	0.996	0.997	0.999	**0.999**
loudspeaker	0.823	0.917	0.969	0.932	0.954	0.965	**0.972**
mailbox	0.873	0.924	0.942	0.942	0.947	0.948	**0.950**
microphone	0.997	0.998	**1.000**	1.000	1.000	1.000	**1.000**
microwaves	0.973	0.959	**0.977**	0.969	0.973	0.975	0.975
motorbike	0.657	0.826	**0.921**	0.799	0.846	0.877	0.895
mug	0.827	0.914	0.956	0.932	0.953	0.955	**0.957**
piano	0.755	0.921	**0.970**	0.928	0.954	0.963	0.968
pillow	0.826	0.925	0.950	0.925	0.946	0.950	**0.954**
pistol	0.899	0.973	0.987	0.971	0.980	0.985	**0.988**
flowerpot	0.782	0.836	**0.897**	0.843	0.857	0.878	0.882
printer	0.730	0.832	**0.899**	0.847	0.874	0.893	0.898
remote	0.893	0.958	**0.997**	0.970	0.985	0.996	0.994
rifle	0.922	0.987	**0.998**	0.990	0.993	0.995	0.997
rocket	0.830	0.949	**0.994**	0.963	0.982	0.983	0.992
skateboard	0.972	0.986	**0.999**	0.992	0.996	0.999	**0.999**
sofa	0.918	0.967	**0.988**	0.966	0.977	0.983	0.986
stove	0.845	0.890	**0.932**	0.898	0.909	0.919	0.927
table	0.961	0.982	0.998	0.993	0.996	0.997	**0.999**
telephone	0.917	0.976	**0.990**	0.966	0.980	0.985	0.988
tower	0.783	0.872	0.916	0.887	0.914	0.919	**0.926**
train	0.848	0.918	0.965	0.930	0.952	0.960	**0.967**
watercraft	0.832	0.958	**0.989**	0.960	0.979	0.985	0.988
washer	0.937	0.956	0.977	0.967	0.972	0.979	**0.980**
mean	0.858	0.933	**0.967**	0.942	0.955	0.963	0.966

Table 4: **Patch sizes.** The models are trained on ShapeNet-v2. We set $N = 2048$ and $M = 512$. We show results of different patch sizes K.

metrics	$K = 1$	$K = 2$	$K = 4$	$K = 8$	$K = 16$	$K = 32$	$K = 64$
IoU ↑	0.940	0.947	0.951	0.952	**0.953**	**0.953**	**0.953**
Chamfer ↓	0.042	0.041	0.041	**0.040**	**0.040**	**0.040**	**0.040**
F-Score ↑	0.959	0.962	0.963	**0.966**	**0.966**	**0.966**	**0.966**

Table 5: **Scene reconstruction.** We train our model on Synthetic Rooms [4]. We set $M = 512$, and $K = 32$. We compare our results with ConvONet of resolution 32^3 and 64^3. The point cloud size N is 2048 and 16384.

	$N = 2048$			$N = 16384$		
	IoU ↑	Chamfer ↓	F-Score ↑	IoU ↑ (Δ)	Chamfer ↓ (Δ)	F-Score ↑ (Δ)
ConvONet-32^3	0.761	0.040	0.950	0.816 (+0.055)	0.036 (-0.005)	0.967 (+0.017)
ConvONet-64^3	0.808	0.036	0.965	0.847 (+0.039)	**0.031** (-0.005)	**0.986** (+0.021)
3DILG (Ours)	**0.854**	**0.032**	**0.975**	**0.866** (+0.011)	**0.031** (-0.001)	0.979 (+0.005)

Table 6: **Perceptual study.** In each "X v.s. Y" group, we show the probability of choosing "X" (>) and "Y" (<) by users. The probability of choosing our method is highlighted in blue.

	GT v.s. 3DILG		Grid-8^3 v.s. 3DILG		GT v.s. Grid-8^3	
	>	<	>	<	>	<
probability	0.7635	0.2365	0.4375	0.5625	0.8604	0.1396

Table 7: Quantitative results on D-FAUST.

	IF-Net	Ours
Chamfer	0.0238	**0.0210**
F1	0.9940	**0.9953**

Table 8: Uniform v.s. non-uniform sampling.

	IF-Net			3DILG (Ours)		
	Uniform	Non-uniform	Δ	Uniform	Non-uniform	Δ
IoU ↑	0.934	0.902	-0.032	0.953	0.954	+0.001
Chamfer ↓	0.041	0.050	+0.009	0.040	0.040	+0.000
F1 ↑	0.967	0.937	-0.030	0.966	0.964	-0.002

Table 9: **Comparison with Poisson Surface Reconstruction (PSR).** Best results are shown in **bold**. Second best results are shown in *italic*.

	PSR	OccNet	ConvOccNet	IF-Net	3DILG (Ours)
Chamfer ↓	0.043	0.072	0.052	*0.041*	**0.040**
F-Score ↑	0.922	0.858	0.933	**0.967**	*0.966*