GeoUDF: Surface Reconstruction from 3D Point Clouds via Geometry-guided Distance Representation (Supplementary Materials)

Siyu Ren[1,2] Junhui Hou[1*] Xiaodong Chen[2] Ying He[3] Wenping Wang[4]

[1]City University of Hong Kong [2]Tianjin University

[3] Nanyang Technological University [4]Texas A&M University

siyuren2-c@my.cityu.edu.hk, jh.hou@cityu.edu.hk, xdchen@tju.edu.cn, yhe@ntu.edu.sg, wenping@cs.hku.hk

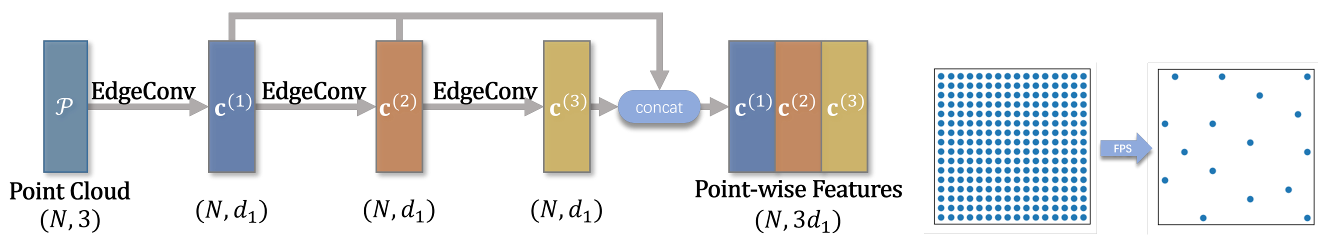
In this supplementary material, we will provide more details of our framework GeoUDF in Section 1, and more details of experiment settings and more results in Section 2. We also refer the readers to the *project page*.

1. More Details of GueUDF

1.1. Local Geometry Representation

Feature Extraction. As shown in Fig. 14a, we employ 3-layer EdgeConv [11] to embed $\mathcal{P}$ into a high-dimensional feature space, producing hierarchical features, $\{\mathbf{c}_i^{(l)} \in \mathbb{R}^{d_1}\}_{i=1}^N$, $l = 1, 2, 3$, which are concatenated as the point-wise features of $\mathcal{P}$.

Poisson Disk Sampling over $\mathcal{D}$. As shown in Fig. 14b, we apply the farthest point sampling (FPS) on the uniform 2D grids to approximate Poisson Disk Sampling on the 2D area $\mathcal{D}$ for obtaining 2D coordinates.



(a) (b)

Figure 14: (a) Feature Extraction. (b) Poisson Disk Sample in $\mathcal{D}$.

1.2. Geometry-guided UDF Estimation

Discussion on the differences from IMLS-based methods DeepIMLS [6] and DOG [10]. In DeepIMLS [6] and DOG [10], the *SDF* of a query point is formulated as the weighted averaging of the distances of the query point to the tangent planes of a set of *MLS (moving least-squares) points*, i.e., Eq. (2) of DeepIMLS [6] and Eq. (3) of DOG [10]. In the GUE module of our method (i.e., Eq. (4)), we formulate the *UDF* of a query point as the weighted averaging of the distances of the query point to the tangent planes of a set of *neighboring points on the surface*. Although the IMLS-based methods DeepIMLS [6] and DOG [10] and our method adopt similar formulas, they are *significantly* different.

- In DeepIMLS [6] and DOG [10], the MLS points are NOT distributed on surfaces, and the MLS points and their weights are *simultaneously* learned i.e., the MLP points are learned, which are then used to calculate their weights via a *pre-defined* function with a learnable factor directly (see Eq. (2) of [6] or Eq. (3) of [10]), resulting in that they may suffer from the *coupling* issue. *Differently*, in the GUE module of our method (i.e., Eq. (4)), *only* the weights are learned. The points used to calculate the distances (i.e., $\{\mathbf{p}_k\}$) are distributed *on the surface* and obtained by an upsampling module (i.e., LGR) pre-trained (see Lines 491 - 497 of our manuscript) under the supervision of dense point clouds. In other words, our method decouples the learning of the weights and the points.

- In DeepIMLS [6], the gradient estimation of the SDF (i.e., Eq. (7) of [6]) is part of the real derivative of the SDF (i.e., Eq. (2) of [6]), thus coupled with SDF estimation and suffering from the inaccuracy of SDF estimation. DOG [10] estimates the SDF gradients through the auto-differentiation of SDF (i.e., Eq. (3) of [10]), which is time-consuming. *Differently*, the GUE module of our method *separates* the estimation of UDFs and gradients. Specifically, based on the continuity of a surface, GUE learns another set of weights to average the aligned normals of K-NNs to estimate UDF gradients, i.e., Eq. (5) of our manuscript. Thus, our UDF gradient can achieve *bias-free*. Note that the learned weights in Eqs. (4) and (5) of our manuscript are independent.
- More importantly, we demonstrate the superiority of our method over DOG [10] both quantitatively and qualitatively in Table 1 and Fig. 7 of the manuscript. Note that according to the results reported in the original paper of DOG [10], DOG [10] achieves better performance than DeepIMLS [6].

1.3. Edge-based Marching Cube

Edge Intersection Detection. Fig. 15 shows all kinds of the position relationship between the surface and edge.

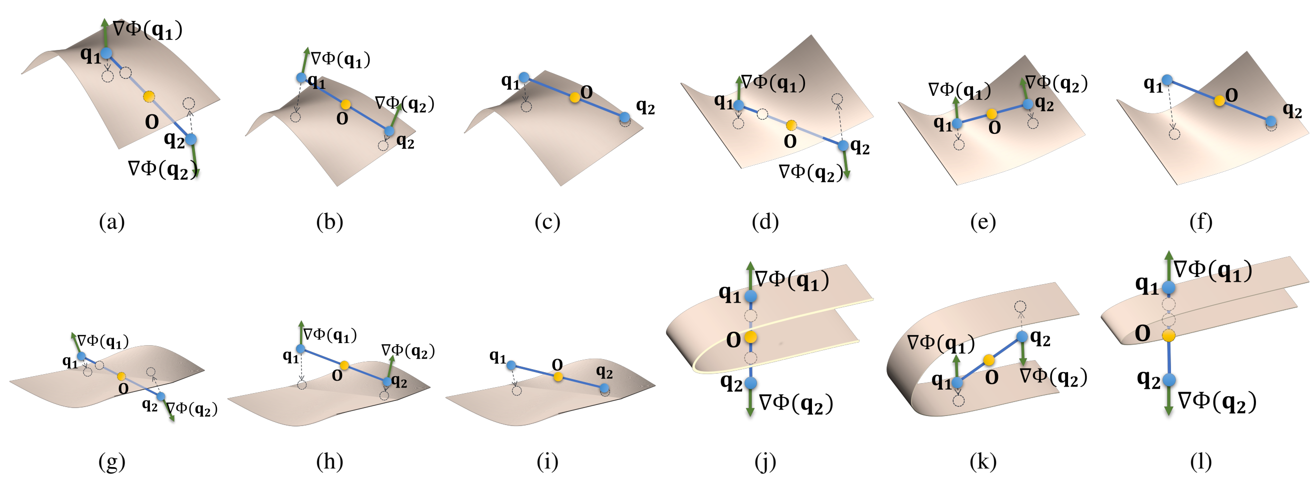


Figure 15: The position relationship between the surface and edge.

2. Experiments

2.1. Implementation Details

During training, we set the upsampling factor $M = 16$, and the boundary of $\mathcal{D}$ $\delta = 0.1$. The ground-truth dense point clouds were obtained by sampling the corresponding mesh models through Poisson Disk Sampling [2]. For each shape, we randomly sampled $N' = 2048$ points on the surface, then added Gaussian noise with the standard deviation of 0.05, 0.02, or 0.03 to move these points away from the mesh to generate query points in each training iteration. The feature dimensions were $d_1 = 384$ and $d_2 = 128$. We used the Adam [5] optimizer to train our network and set the batch size to 16.

During inference, the threshold for edge intersection detection in Sec. 3.3 of the manuscript was set to $\tau = 5 \times 10^{-4}$, and the resolution of E-MC was 128 for the shapes and garments in the ShapeNet and MGN datasets, and 192 for the shapes or scenes in the ShapeNet car and ScanNet datasets.

2.2. Evaluation Metric

Following previous work [7, 9, 8, 1, 12], we used CD and F-Score to evaluate the reconstruction accuracy. We denote the reconstructed and ground-truth mesh by $\mathcal{S}_{\text{REC}}$ and $\mathcal{S}_{\text{GT}}$, on which we randomly sample a set of 10^5 points, denoted as $\mathcal{P}_{\text{REC}}$ and $\mathcal{P}_{\text{GT}}$ and the CD of these two meshes is that of the two point sets, i.e.,

$$\mathtt{CD}(\mathcal{S}_{\text{REC}}, \mathcal{S}_{\text{GT}}) = \mathtt{CD}(\mathcal{P}_{\text{REC}}, \mathcal{P}_{\text{GT}}). \tag{15}$$

The F-Score is defined as the harmonic mean between the precision and the recall of points that lie within a certain distance threshold ϵ between $\mathcal{S}_{\text{REC}}$ and $\mathcal{S}_{\text{GT}}$,

$$\texttt{F-Score}(\mathcal{S}_{\text{REC}}, \mathcal{S}_{\text{GT}}, \epsilon) = \frac{2 \cdot \texttt{Recall} \cdot \texttt{Precision}}{\texttt{Recall} + \texttt{Precision}}, \tag{16}$$

where

$$\begin{aligned}\texttt{Recall}(\mathcal{S}_{\text{REC}}, \mathcal{S}_{\text{GT}}, \epsilon) &= \left| \left\{ \mathbf{p}_1 \in \mathcal{P}_{\text{REC}}, \text{ s.t. } \min_{\mathbf{p}_2 \in \mathcal{P}_{\text{GT}}} \|\mathbf{p}_1 - \mathbf{p}_2\| < \epsilon \right\} \right|, \\ \texttt{Precision}(\mathcal{S}_{\text{REC}}, \mathcal{S}_{\text{GT}}, \epsilon) &= \left| \left\{ \mathbf{p}_2 \in \mathcal{P}_{\text{GT}}, \text{ s.t. } \min_{\mathbf{p}_1 \in \mathcal{P}_{\text{REC}}} \|\mathbf{p}_1 - \mathbf{p}_2\| < \epsilon \right\} \right|.\end{aligned}$$

In the ablation study, we evaluate the accuracy of UDFs and their gradients. We first set uniform grids of 64^3 in a unit cube, and then for each shape, we chose the grids near the surface, i.e., their UDFs are larger than 5×10^{-4} and smaller than 0.02. We use the absolute error and angle error to measure the evaluated UDFs and their gradients of these query points.

2.3. Settings of the Experiments on Clean Data

In our manuscript, the methods under comparison, including ONet [7], CONet [9], SAP [8], POCO [1], and DOG [10], only conducted experiments on the noisy data in their original papers. And they did not release the pre-trained networks on clean data. Although POCO [1] released the pre-trained network on clean data, it requires ground-truth normals. Due to the different distributions of clean and noisy data, directly applying their pre-trained networks with noisy data to clean data would result in an obvious performance drop. Thus, for a fair comparison, we retrained their models on the clean data using their official codes. The released pre-trained networks of NDF [4] and GIFS [12] were only trained on the ShapeNet car dataset, rather than the whole 13 classes, and thus, we retrained them on both clean and noisy data for fair comparisons.

From the results in Table 1 of our manuscript, it can be seen that some methods, including ONet, CONet, SAP, and DOG, achieve better performance on noisy data than clean data. The possible reasons are: (**1**) with slight perturbation, some thin structures could be extracted through OCC or SDF; and (**2**) the noisy data enhance the robustness of the models during training.

2.4. More Experimental Results

Complex Shapes. Fig. 17 shows the reconstruction results of complex shapes in the ShapeNet dataset.

More Difficult Inputs We also tested the performance of our GeoUDF when inputs are more difficult. Specifically, we tested the trained networks on point clouds with more severe noise (the standard deviation (STD) of Gaussian noise is set to 0.01 and 0.02), visibly non-uniform distribution, and fewer points (the number of pint is 500). Fig. 18 shows our GeoUDF is much more robust than GIFS. Since our GeoUDF relies on the local geometry of input data, it would fail in those regions with extremely sparse points.

Error Map. For each reconstructed shape in Fig. 6 of the manuscript, we calculated the distance of its vertices to the ground-truth mesh, then used *hot* colormap to assign colors for the vertices. The visual results are shown in Fig. 19.

More Visual Results. We present more visual results in Figs. 20, 21, 22, and 23.

2.5. Ablation Study

Upsampling Process. As mentioned in our manuscript, the quadratic polynomial is sufficient to approximate a small surface area. To verify this, we trained a network by replacing Eq. (1) of the manuscript with the cubic polynomial as a comparison. The results are shown in the following Table 1, where it can be seen that the cubic polynomial decreases the upsampling accuracy because of overfitting.

Flowchart of the "Regress" Method. In Table 6 of the manuscript, we set a baseline named "Regress" to predict UDFs and gradients by using an MLP. Fig. 16 shows the detailed flowchart.

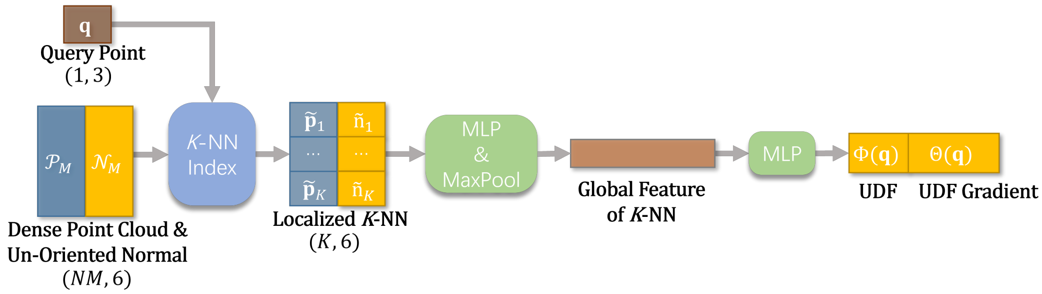
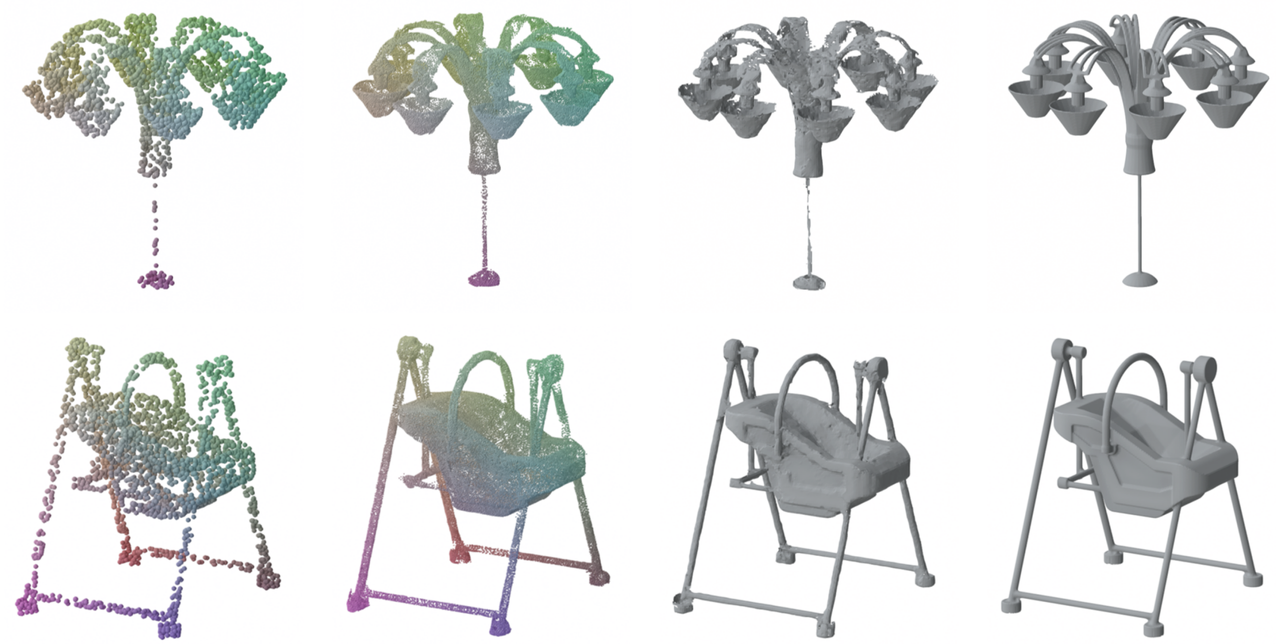


Figure 16: The network architecture of the baseline method "Regress" in Table 6 of the manuscript.



Input Upsampled Ours GT

Figure 17: Reconstruction results of complex shapes in the ShapeNet dataset.

(a) Input (b) GIFS [12] (c) Ours

Figure 18: Visual comparisons on more difficult inputs. Each subfigure, from left to right: STD=0.01, STD=0.02, visibly non-uniform, and 500 points.

Table 1: Comparisons of Quadratic and Cubic Polynomials.

Method	CD (↓) ($\times 10^{-2}$)	P2F (↓) ($\times 10^{-3}$)
Quadratic	0.245	0.512
Cubic	0.257	0.522

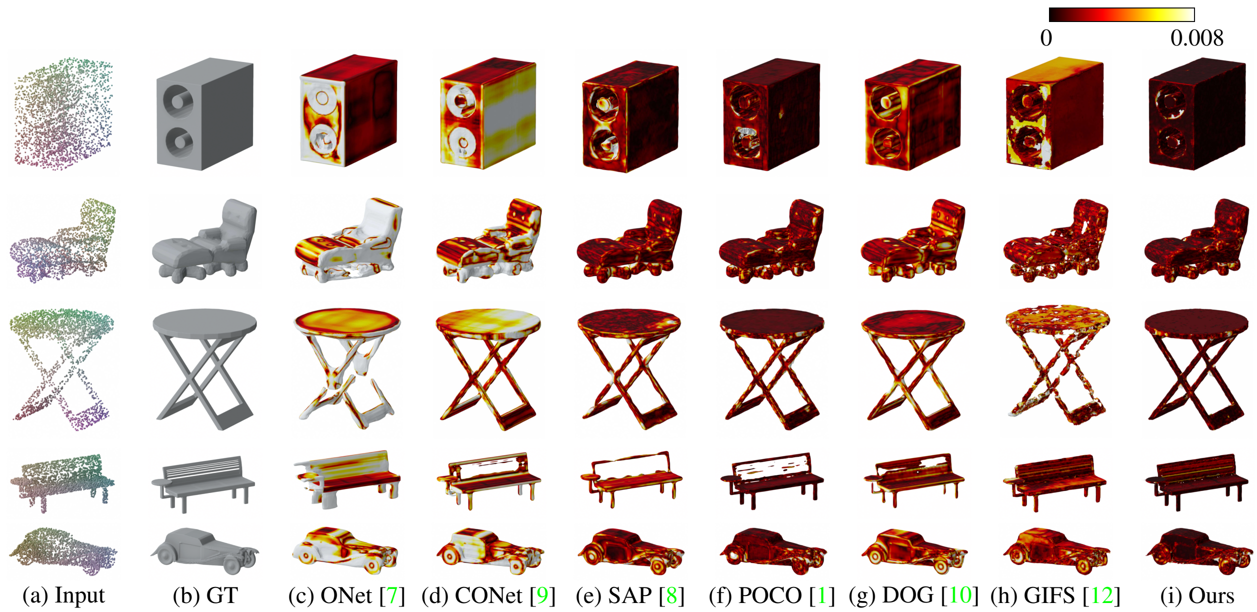


Figure 19: Error Maps of the shapes in the ShapeNet dataset [3].

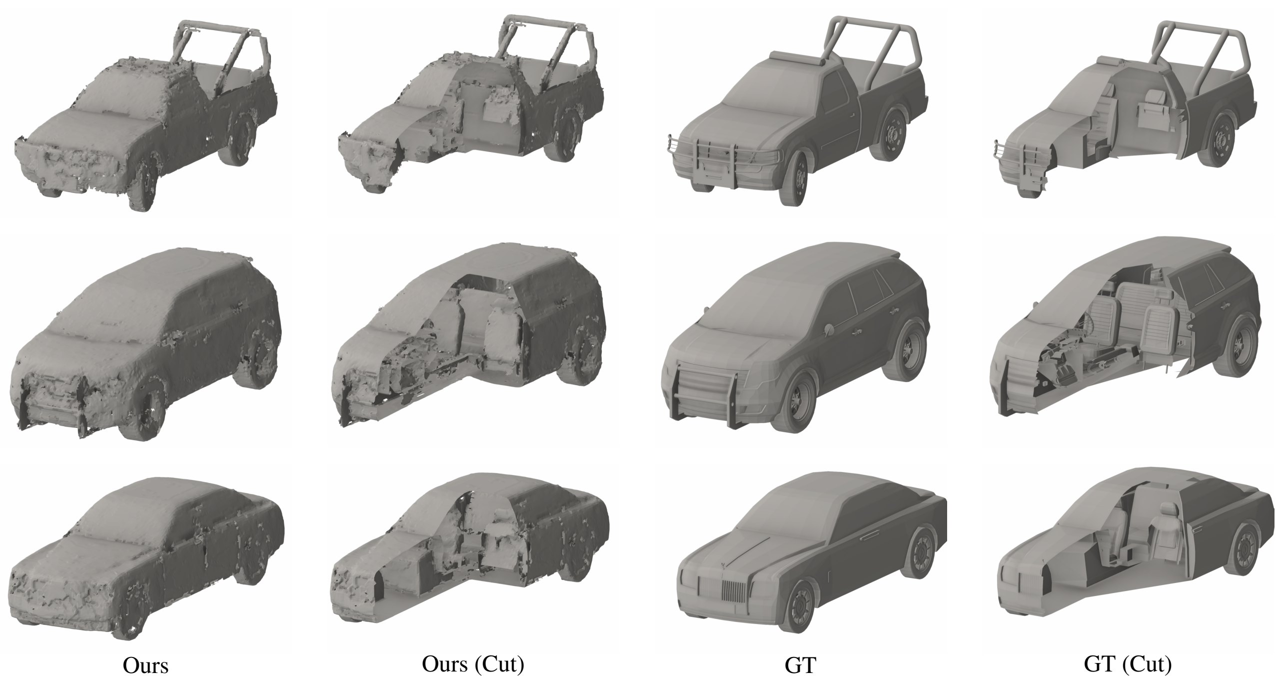


Figure 20: More results of reconstructed shapes with multi-layer structures.

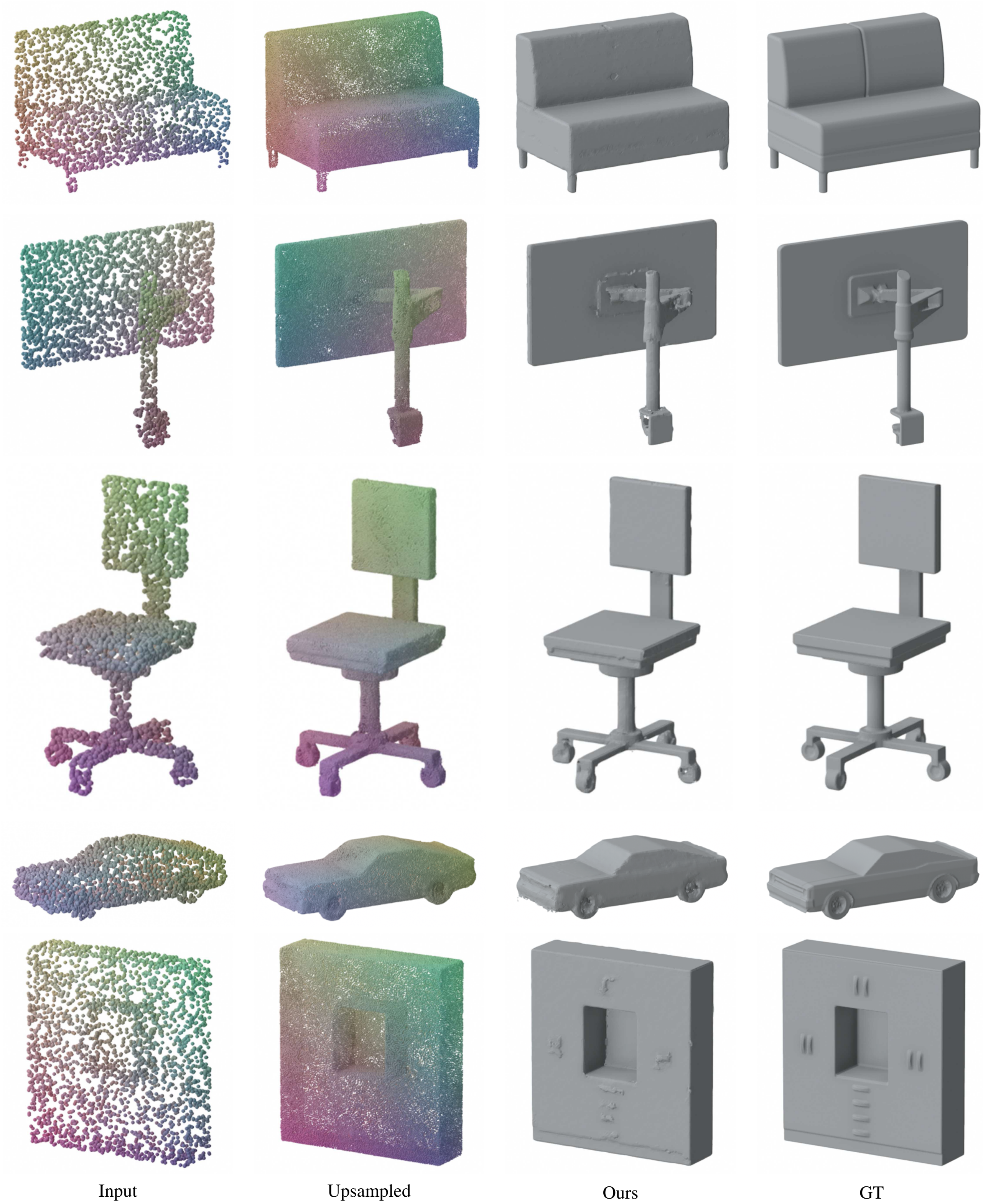


Figure 21: More results of reconstructed water-tight shapes.

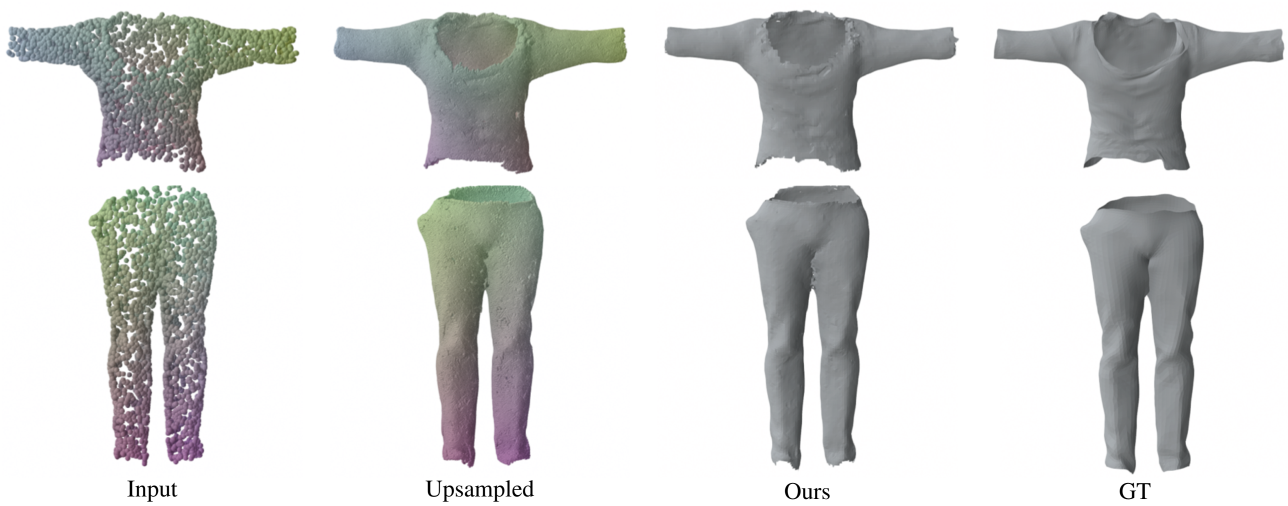


Figure 22: More results of reconstructed open shapes.

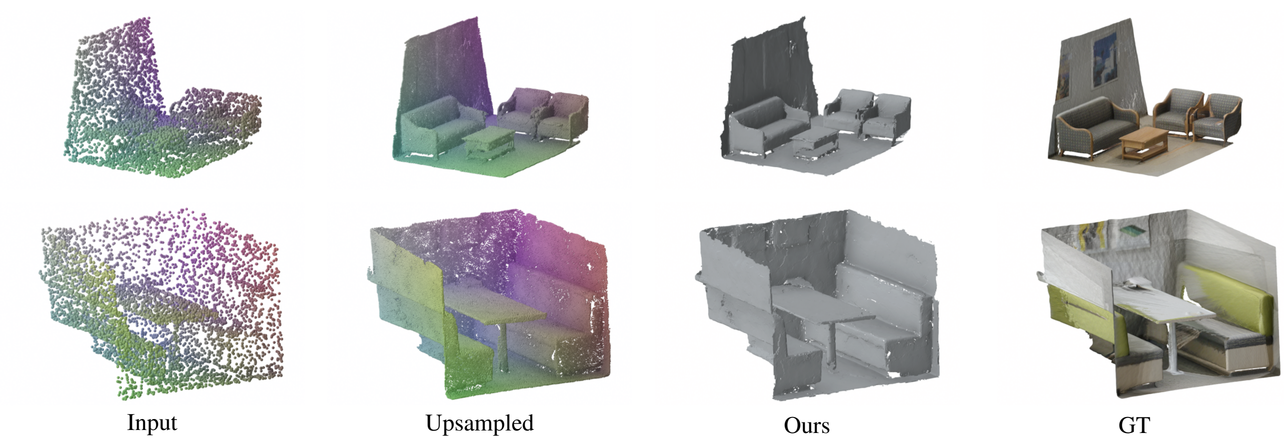


Figure 23: More results of reconstructed scene-level shapes.

References

[1] Alexandre Boulch and Renaud Marlet. Poco: Point convolution for surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6302–6314, June 2022. 2, 3, 5

[2] Robert Bridson. Fast poisson disk sampling in arbitrary dimensions. In *ACM SIGGRAPH 2007 sketches*, pages 22–es. 2007. 2

[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*, pages 1–xxx, 2015. 5

[4] Julian Chibane, Gerard Pons-Moll, et al. Neural unsigned distance fields for implicit function learning. *Advances in Neural Information Processing Systems*, 33:21638–21652, 2020. 3

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[6] Shi-Lin Liu, Hao-Xiang Guo, Hao Pan, Peng-Shuai Wang, Xin Tong, and Yang Liu. Deep implicit moving least-squares functions for 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1788–1797, June 2021. 1, 2

[7] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, June 2019. 2, 3, 5

[8] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. *Advances in Neural Information Processing Systems*, 34:13032–13044, 2021. 2, 3, 5

[9] 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. 2, 3, 5

[10] Peng-Shuai Wang, Yang Liu, and Xin Tong. Dual octree graph networks for learning adaptive volumetric shape representations. *ACM Transactions on Graphics*, 41(4):1–15, 2022. 1, 2, 3, 5

[11] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics*, 38(5):1–12, 2019. 1

[12] Jianglong Ye, Yuntao Chen, Naiyan Wang, and Xiaolong Wang. Gifs: Neural implicit function for general shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12829–12839, June 2022. 2, 3, 4, 5