A. Appendix

In Section A.1 we give further details about our evaluation, in particular the datasets that we used, the methods we compared (and how we compared to them) and the metrics that we used.

In Section A.2 we give more insights into the octree. We discuss the videos that are also provided in our supplement, and then we explain our 2D example from the main paper in more detail, with more diagrams and annotations. In particular we give intuition into what our energy function does, and how our move making algorithm efficiently minimises it.

In Section A.3 we provide extended results and visualisations. For ShapeNet we give the metric breakdown per shape class, and visualize a shape from each shape class. For SRB we give the metric breakdown per shape, and visualize each shape. We also give a depth ablation on SRB, analogous to depth ablation on ShapeNet in the main paper.

In Section A.4 we provide the implementation details for all stages of our method, including hyperparamters.

A.1. Datasets, Methods and Evaluation metrics

A.1.1 Datasets

ShapeNet We use the preprocessing and split of Neural Splines [17], which consists of 13 shape classes with 20 shapes in each shape class. The shapes use Occupancy Networks's [10] preprocessing to make them watertight via TSDF fusion. They also provide the uniform surface points (with normals) for Chamfer Distance, as well as ground truth occupancy for 100k points randomly sampled in the space for computing the IoU metric.

Surface Reconstruction Benchmark (SRB) we use the version of the original dataset [4] provided by DGP [16]. The dataset provides a 'scan' point cloud (with normals) that simulate noisy range scanning with missing regions and misalignment, and a 'ground truth' point cloud that densely covers the surface without missing regions.

A.1.2 Other Methods

For SAL, IGR, SIREN (and SIREN wo n), FFN [15], Biharmonic RBF [5], SVR [13], NSP and DiGS (and DiGS + n) we report results from the papers of NSP [17] and DiGS [3], and note that all these methods have code available. In particular we were able to reproduce the results (or within a decent tolerance) for all these methods.

For SAP [12], we obtained the results ourselves using the code provided by the authors. To be consistent with their paper, we use their hierarchical setting not the uniform setting, which greatly reduces the time taken. While SAP has results on both ShapeNet and SRB, their ShapeNet results are for the learning setting only (not the optimisation setting which corresponds to our setting) and their SRB results are with different metrics (we use the metrics that are most associated with the dataset, which come from the paper that currently released the data [16]).

For SPSR [8] and normal estimation + SPSR, we used the industry-standard implementation in Open3D [18]. In particular, SPSR is performed with depth 9, and normal estimation is done by plane fitting, and then the orientation is made consistent by using a minimum spanning tree. Note that the results with the Open3D implementation are much better than the results reported for research code of SPSR in NSP [17] and SAP [12].

A.1.3 Metrics

We follow the evaluation procedure of DiGS [3], who use different metrics for the two datasets in order to stay consistent with the literature on those datasets.

For the ShapeNet dataset, they report the squared Chamfer to be consistent with previous works [11, 17], which is a distance between two point clouds (thus measuring predicted surface). They also report the volumetric Intersection over Union (IoU) of the interior of the shapes as per Occupancy Netowrks [10] who released the preprocessed version of the dataset used, which is consistent with [17]. This metric compares the underlying occupancy (interior) predicted by the method.

For the SRB dataset, they use the Chamfer (d_C) and Hausdorff (d_H) distances which is consistent with the literature on SRB [7, 9, 16, 17]. These metrics only compare the accuracy of the predicted surface of the shape.

They define the squared Chamfer distance used in DeepSDF [11] as

$$d_C^{sq}(\chi_1, \chi_2) = d_{\vec{C}}^{sq}(\chi_1, \chi_2) + d_{\vec{C}}^{sq}(\chi_2, \chi_1) \tag{1}$$

$$d_{\vec{C}}^{sq}(\chi_1, \chi_2) = \frac{1}{|\chi_1|} \sum_{x_1 \in \chi_1} \min_{x_2 \in \chi_2} \|x_1 - x_2\|_2^2 . \tag{2}$$

$$\tag{3}$$

Note that the distances are in the same scaling as the input/ground truth point cloud, they have not applied any scaling before calculating distances.

For the volumetric IoU, following Occupancy Networks [10] they obtain unbiased estimates of the occupied (interior) volume of the shapes by evaluating ath the 100k points randomly sampled in the space provided by the dataset. Thus given the known occupancy of the ground truth mesh $O_{GT}(x) \in \{0, 1\}$ (which is provided in the dataset) and the predicted SDF $\Phi(x) \in \mathbb{R}$ for a points

$x \in \chi$, the occupancy of the SDF is given by

$$O_\Phi(x) = \begin{cases} 1 & \Phi(x) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

and the IoU is given by

$$IoU_\chi(O_{GT}, O_\Phi) = \frac{\sum_{x \in \chi} O_{GT}(x) \text{ and } O_\Phi(x)}{\sum_{x \in \chi} O_{GT}(x) \text{ or } O_\Phi(x)}. \tag{5}$$

They define the Chamfer and Hausdorff distances as

$$d_C(\chi_1, \chi_2) = \frac{1}{2}(d_{\vec{C}}(\chi_1, \chi_2) + d_{\vec{C}}(\chi_2, \chi_1)) \tag{6}$$

$$d_H(\chi_1, \chi_2) = \max(d_{\vec{H}}(\chi_1, \chi_2), d_{\vec{H}}(\chi_2, \chi_1)) \tag{7}$$

$$\tag{8}$$

where

$$d_{\vec{C}}(\chi_1, \chi_2) = \frac{1}{|\chi_1|} \sum_{x_1 \in \chi_1} \min_{x_2 \in \chi_2} \|x_1 - x_2\|_2 \tag{9}$$

$$d_{\vec{H}}(\chi_1, \chi_2) = \max_{x_1 \in \chi_1} \min_{x_2 \in \chi_2} \|x_1 - x_2\|_2 \tag{10}$$

are the one directional Chamfer distance and one directional Hausdorff distance respectively.

Note that the distances are in the same scaling as the input (scan) and ground truth point cloud, they have not applied any scaling before calculating distances.

A.2. Octree Energy Function and Move-Making Algorithm: Further Explanation

A.2.1 Animation: Octree algorithm in 3D

With our supplementary material we also provide videos showing our octree algorithm acting on point clouds from our dataset. In these videos, surface leaves are yellow and inside non-surface leaves are blue, just like our 2D visualizations. We show each accepted move of the octree algorithm being made.

A.2.2 Explanation of our 2D example

We now give more intuition into what the proposed energy function and move-making algorithm on the octree are doing using the 2D example shown in Figure **??** of the main paper. We further expanded upon it in Figure 2 for more stages and depths, and annotate key stages that we discuss further here in Figure 1. These are best viewed when zoomed in, especially for the larger depths or for the annotations.

In this 2D example we use the following values for our energy function hyperparameters: $\gamma^{(0)} = 0.5$, $\gamma^{(1)} = 3$, $\eta^{(0)} = 0.25$, $\eta^{(1)} = 1$ and $\lambda << 1$. Note that

- As $\gamma^{(0)} = 0.5$, we want each surface leaf to have an outside count of 0.5 neighbors, where non-surface outside neighbors (grey in Figure **??**) count as 1 outside neighbor and surface neighbors (yellow in Figure **??**) count as 0.25 of an outside neighbor (as $\eta^{(0)} = 0.25$).
- As $\gamma^{(1)} = 3$, we want each surface leaf to have an inside count of 3 neighbors, where non-surface inside neighbors (blue in Figure **??**) count as 1 inside neighbor and surface neighbors (yellow in Figure **??**) count as 1 inside neighbor (as $\eta^{(1)} = 1$).
- Neighbors in this context include diagonals.
- The energy function penalises by the amount of inside/outside neighbors not satisfied for each surface leaf.
- As $\lambda << 1$, satisfying surface leaf inside/outside counts is more important than decreasing surface area between inside and outside leafs on the border.

In Figure **??** (A) we have the initially created octree at depth 3, and its initialised labels. The first thing to note is that the shape is made to fit within a sphere of radius 1, while the domain is a cube of radius 1.1 (each edge is of length 2.2). As a result, there is a significant region of space surrounding the shape. When the octree is built to depth 3, in this case (and most cases) none of the surface leaves are on the boundary of the domain, allowing the initialisation to have the whole boundary (or most of the boundary in the worst case) as outside, while all other non-surface leaves are inside.

In Figure **??** (B) (see also Figure 2 (B)) we have applied the $Grow(\mathcal{O})$ function. First all leaves next to the surface leaves have been expanded to the same depth as the surface leaves. Then all inside nodes on the border who have more than half their neighbors as outside are set as outside. This is repeated until there are no such neighbors left, and in case there is an inside leaf on the border not at the maximum depth, we also expand such leaves.

In Figure **??** (C) (see also Figure 2 (C)) we have applied the $makeMove(\mathcal{O})$ function. However as there are no inside leaves on the border, the there were no moves available to consider, and thus no changes.

In Figure **??** (D) (see also Figure 2 (D)) we have expanded the surface leafs to reach depth 4, applied the $Grow(\mathcal{O})$ function and applied the move making (see Figure 2 for a visualization of these steps).

In Figure **??** (E) we have expanded to reach depth 5, and applied the $Grow(\mathcal{O})$ function. Note that every inside leaf on the border has at least 4 neighbors that are not outside (otherwise the $Grow$ function would have changed them), and they are at the maximum depth (the same depth as the surface leaves).

In Figure **??** (F) we apply moving making with a maximum move set size of $s = 1$, which means that our in our moves we can only consider changing one inside leaf at a time. We provide a new version of the figure, showing the leaves that have been changed with green dots, in Figure 1 (E)/(F).

In all cases, the surface leaves had the required surface leaf counts (for both inside and outside) both before and after the changes. Instead, the moves decreased the surface area of the border between inside and outside leafs. We can see that we cannot decrease the border in any more single changes, unless we violate that surface leaves have an inside count of 3, which we point out with red dots (the inside leaf) and red arrows (the relevant surface leaf).

There are also large inside neighborhoods where if we removed all of it, we could decrease the border without violating the inside counts, and possibly helping the outside counts. These are indicated with thin green (and some of the red) arrows in Figure 1 (F). Note that if we changed the leaves in that neighborhood one at a time, we would not be increasing the border size at each step, which is why we needed to consider multi-leaf moves.

In Figure **??** (G) we change multiple leaves at the same time. Note that we build the moves we are considering by starting from an inside leaf on the border, and adding inside neighbors, all the while considering whether the current addition makes it a better move or not. Hence the iterative move addition is somewhat like a path, so we have drawn a few possible move sets with thin arrows in Figure 1 (F).

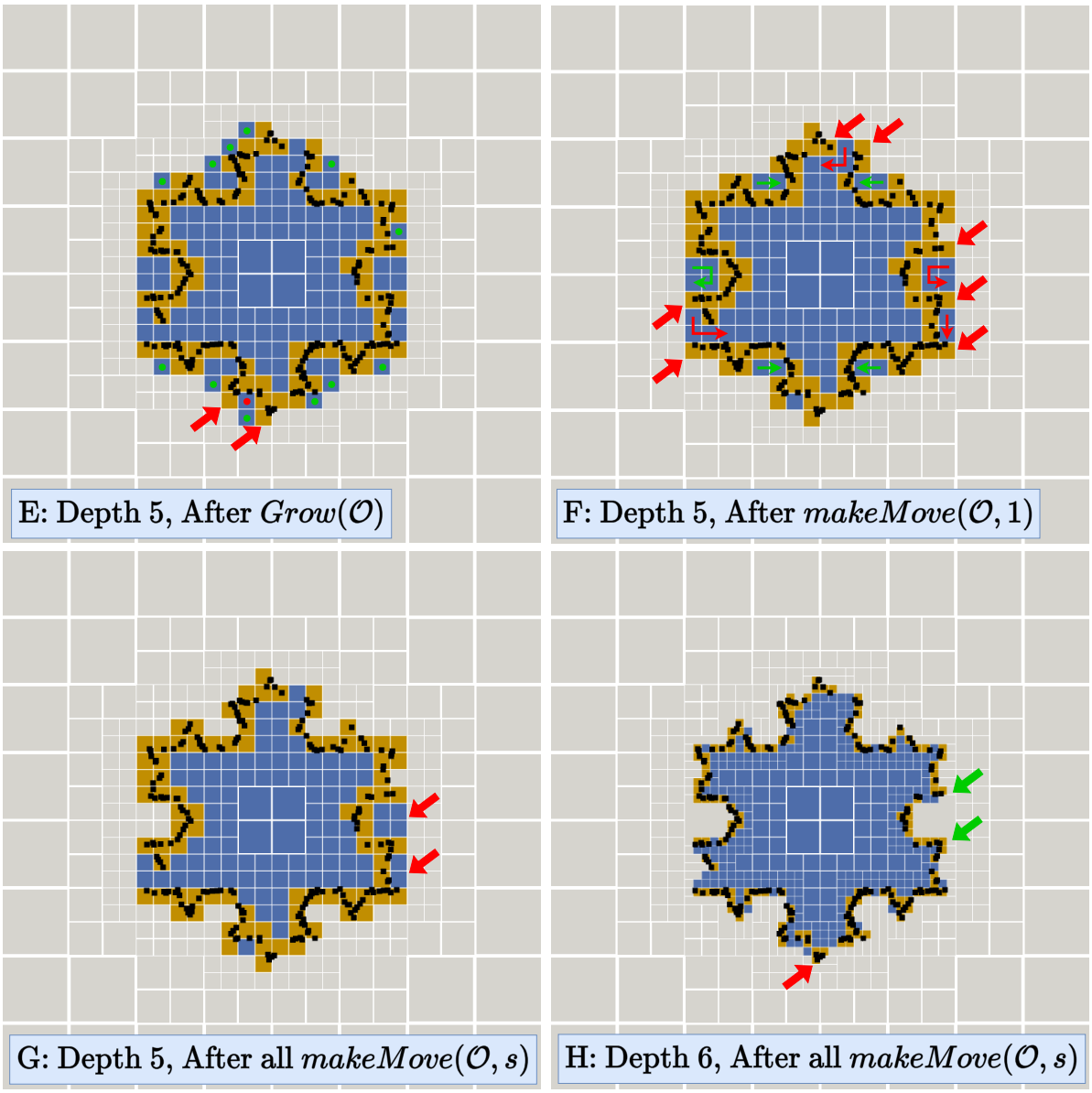

Figure 1. Our algorithm demonstrated on a 2D quadtree example. This expands upon Figure **??** and shows stage (E)-(H) with annotations. The black dots are our input point cloud, the yellow squares are surface leaves, and the blue and gray squares are non-surface leaves labelled as inside and outside respectively.

Now we can see changes to big concave regions between (F) and (G), where larger move set decreased the border and possibly satisfied some outside counts. These are due to successful multi-leaf moves, which we indicate by by thin green arrows in (F).

We show some moves that failed with thin red arrows. We also point out the surface leaves whose inside count would have been violated by those moves in larger red arrows. We can see however that the ones that should be changed eventually do get changed in (H).

In Figure **??** (H) we have expanded, applied the $Grow(\mathcal{O})$ function and applied the move making. We can see now that the regions we discussed before have been fixed, the new surface leaves in the regions indicated by the green arrows now have an inside count of 3, though we still need to keep and inside leaf next to the surface leaf in the region indicated by the red arrow.

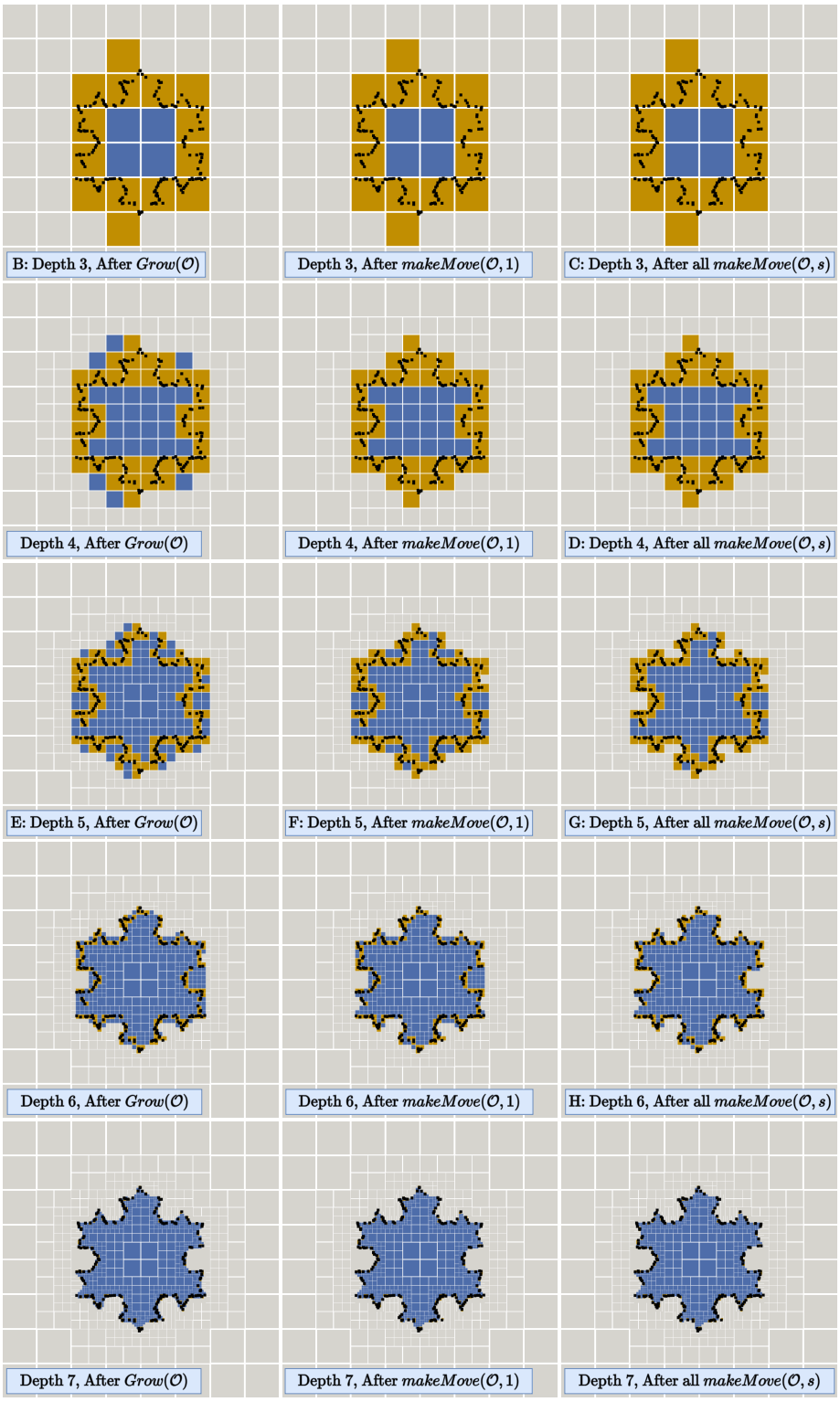

Figure 2. Our algorithm demonstrated on a 2D quadtree example. This expandas upon Figure **??** and shows stage (B)-(H) from there. The black dots are our input point cloud, the yellow squares are surface leaves, and the blue and gray squares are non-surface leaves labelled as inside and outside respectively.

A.3. Full shape/category metric breakdowns

A.3.1 ShapeNet Performance

Table 1 and Table 2 shows the performance of OG-NGLOD and OG-SIREN (both at depth 7) compared to previous methods (including those using normals) on each of the 13 classes in the ShapeNet split. For squared Chamfer distance, the best performing method is usually one of the octree guided methods, with SAP sometimes outperforming, and for IoU, the octree guided methods usually have the best mean IoU and often have the best median IoU, again sometimes being outperformed by SAP. However, apart from mean IoU on *sofa*, the octree guided methods are always very competitive, while for *chair*, *lamp* and *table* on both metrics and *bench* on IoU, the octree guided methods significantly outperform the other methods. Overall, the octree guided methods have a significantly better mean on both metrics, while median is similar to both DiGS and SAP, indicating that DiGS and SAP have many more shapes with very poor performance than the octree guided methods. Interestingly the performance is often better for these shapes compared to methods that use normal information.

Visualizations of the reconstruction of a shape from each category can be seen in Figure 3, Figure 4 and Figure 5.

A.3.2 SRB Performance

Table 3 shows the performance of OG-NGLOD and OG-SIREN (both at depth 7) compared to previous methods (including those using normals) on the five shapes of the SRB dataset. Our methods are quite competitive to the best methods (PHASE+FF, DiGS and SAP), with OG-SIREN achieving the second best average performance after DiGS, though it usually does not get the best performance on individual shapes. Like DiGS, our methods are competitive with most methods that use normal information.

Visualizations of the reconstruction of each shape can be seen in Figure 6 and Figure 7.

We also provide a depth ablation of our model on SRB in Table 4.

A.3.3 Comparison to SPSR

We have included results from Open3D's [18] industry-standard implementation of SPSR [8] and normal estimation. These are in blue for both datasets and not directly compared to for bold numbers as they are highly optimised for both speed and accuracy, for comparison the results on the research code of SPSR are given in both NSP [17] and SAP [12] where they do quite poorly. We can see that for ShapeNet, where we have quite uniform sampling of the shape with no noise, when normals are provided the method does close to perfect on all classes. However when normals are not given and need to be estimate, the performance drops significantly, with the performance being extremely poor on *airplane*, *bench* and *lamp*. These are the shapes where there are thin surfaces, which are difficult for normal estimation to deal with.

On SRB, where there are a lot of defects in the input point cloud, with ground truth normals SPSR performs similar to the worst performing method, though all methods do quite well. In particular it does very well on *Lord Quas* which is sampled quite well everywhere except for one large missing region. When normals need to be estimated, SPSR does extremely poorly. In particular it does well on *Lord Quas*, somewhat badly on *Anchor*, and extremely poorly on the rest.

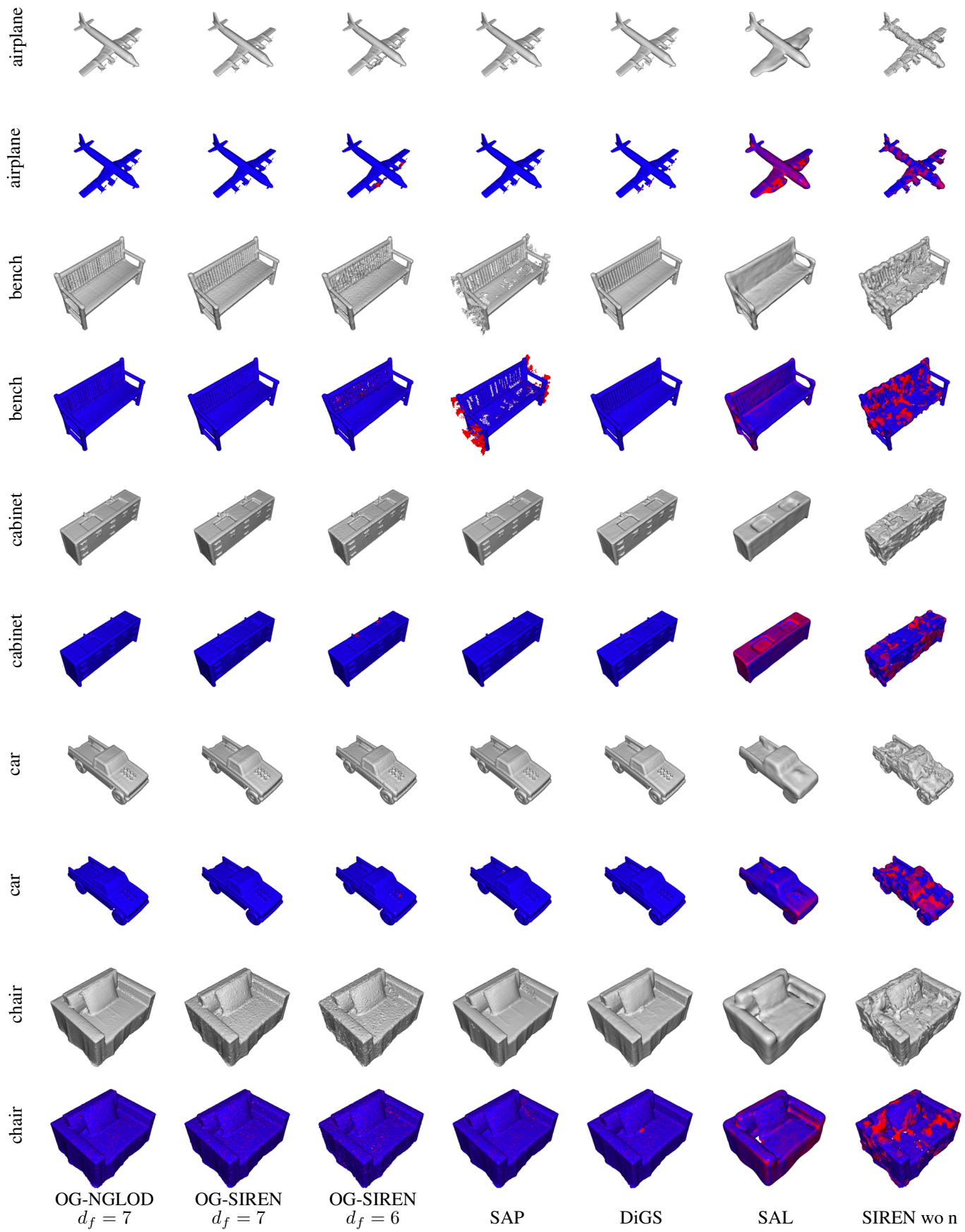

Figure 3. Qualitative results on ShapeNet. A single example for each class has been chosen, and both the reconstructed mesh and the reconstructed mesh colored by distance to input point cloud (more red indicates high distance) are shown. See Section A.3.1 for a discussion on performance.

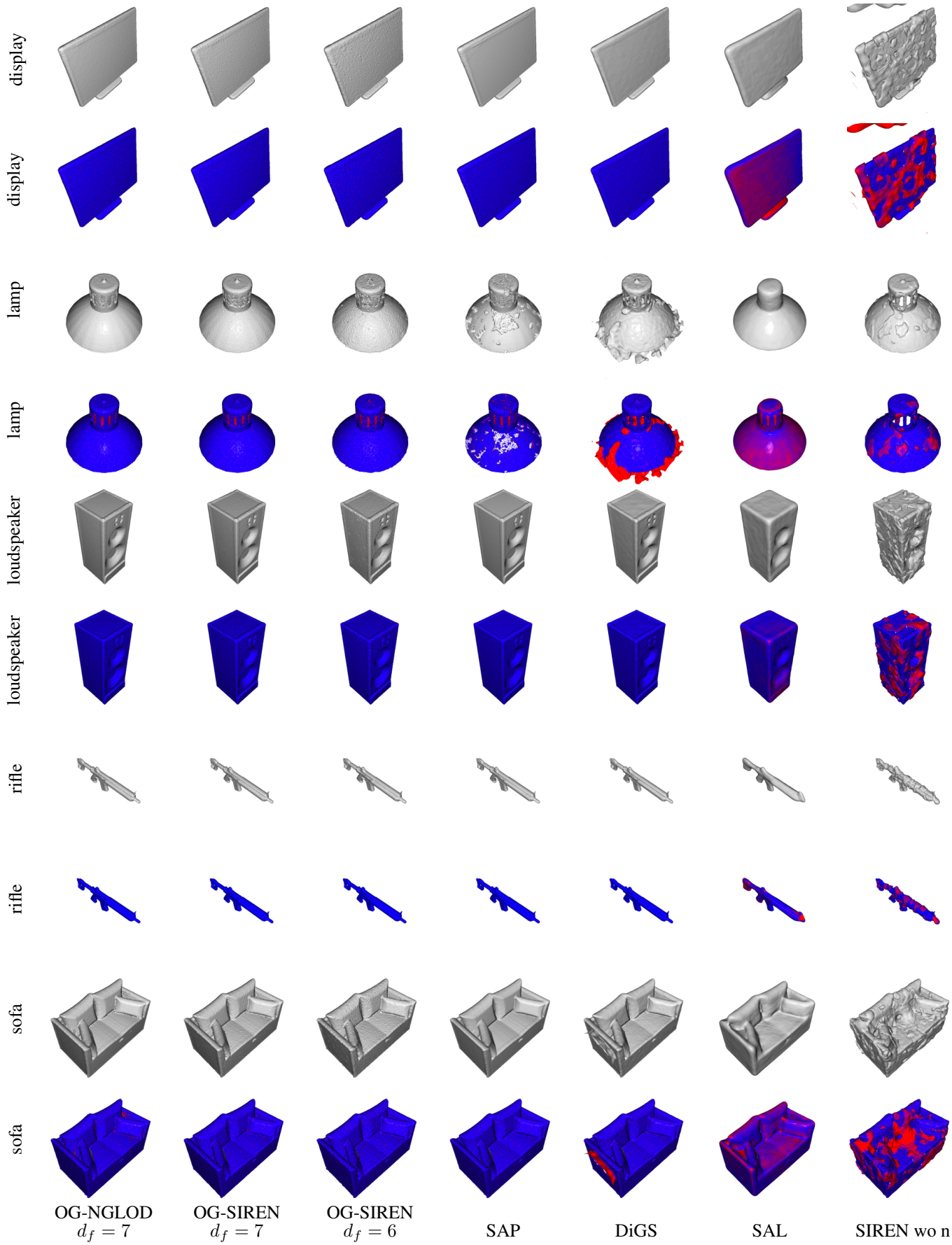

Figure 4. Qualitative results on ShapeNet. A single example for each class has been chosen, and both the reconstructed mesh and the reconstructed mesh colored by distance to input point cloud (more red indicates high distance) are shown. See Section A.3.1 for a discussion on performance.

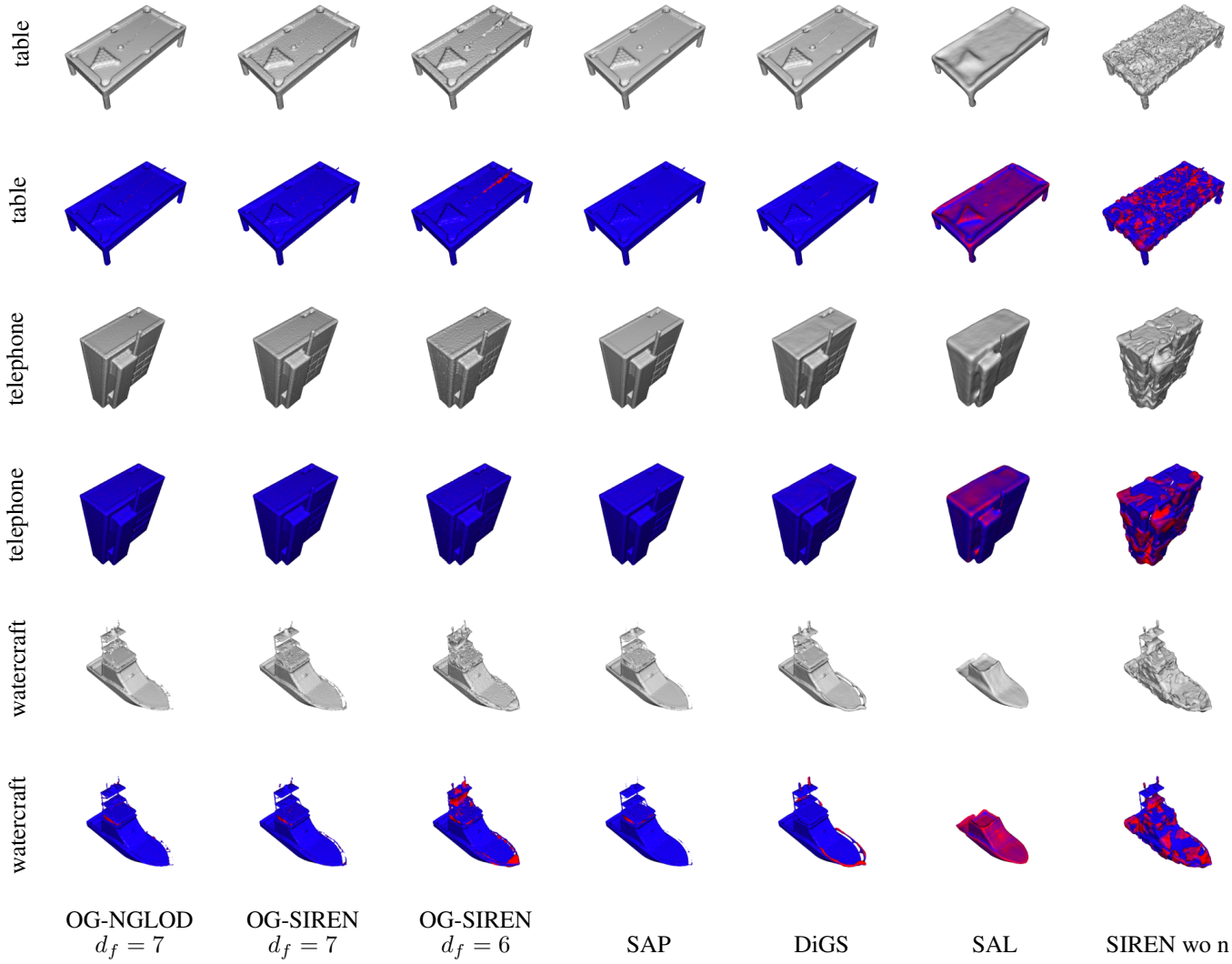

Figure 5. Qualitative results on ShapeNet. A single example for each class has been chosen, and both the reconstructed mesh and the reconstructed mesh colored by distance to input point cloud (more red indicates high distance) are shown. See Section A.3.1 for a discussion on performance.

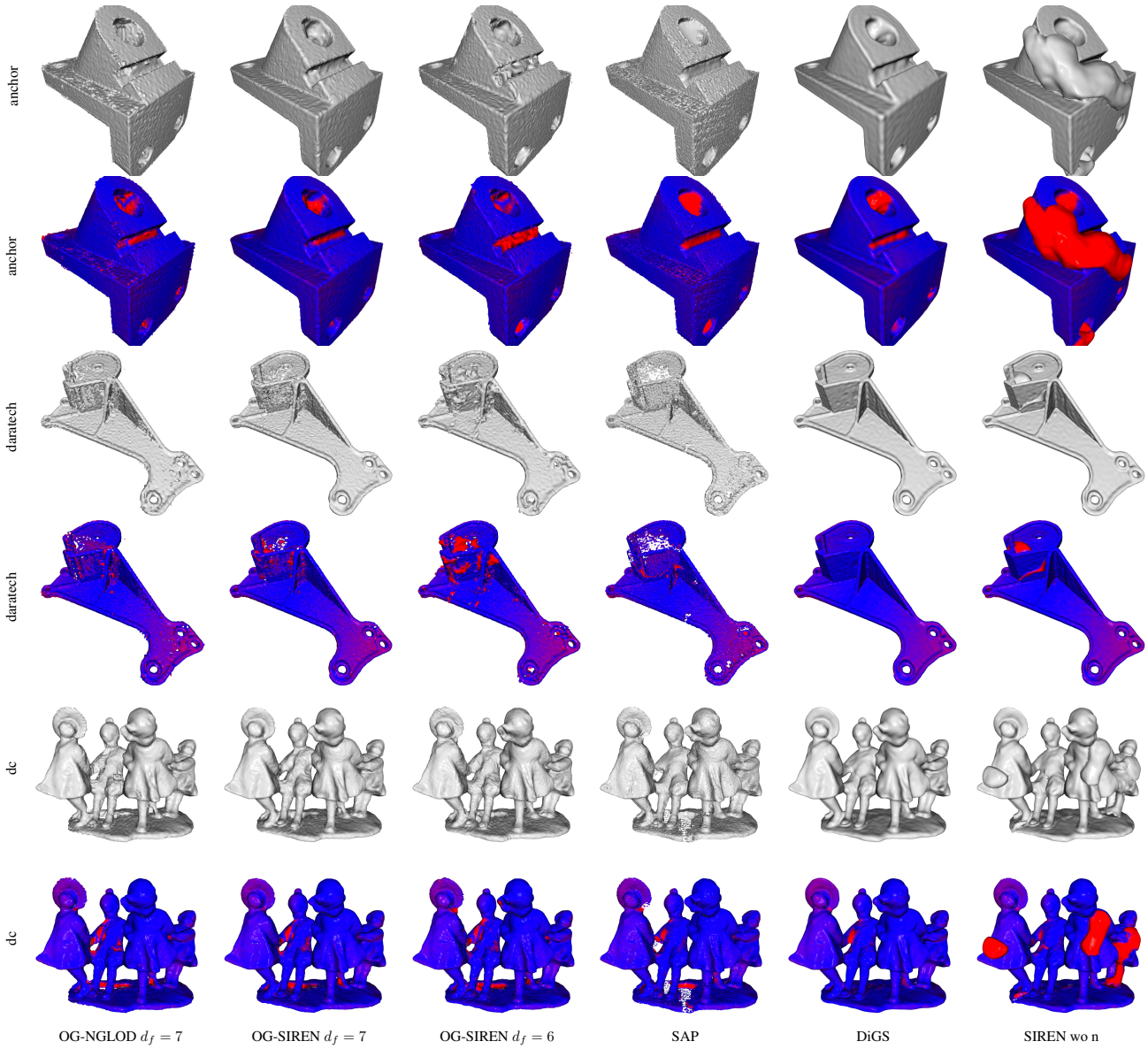

Figure 6. Qualitative results on SRB. For each shape, both the reconstructed mesh the the same mesh colored by distance to input point cloud (more red indicates high distance) are shown. See Section A.3.2 for a discussion on performance.

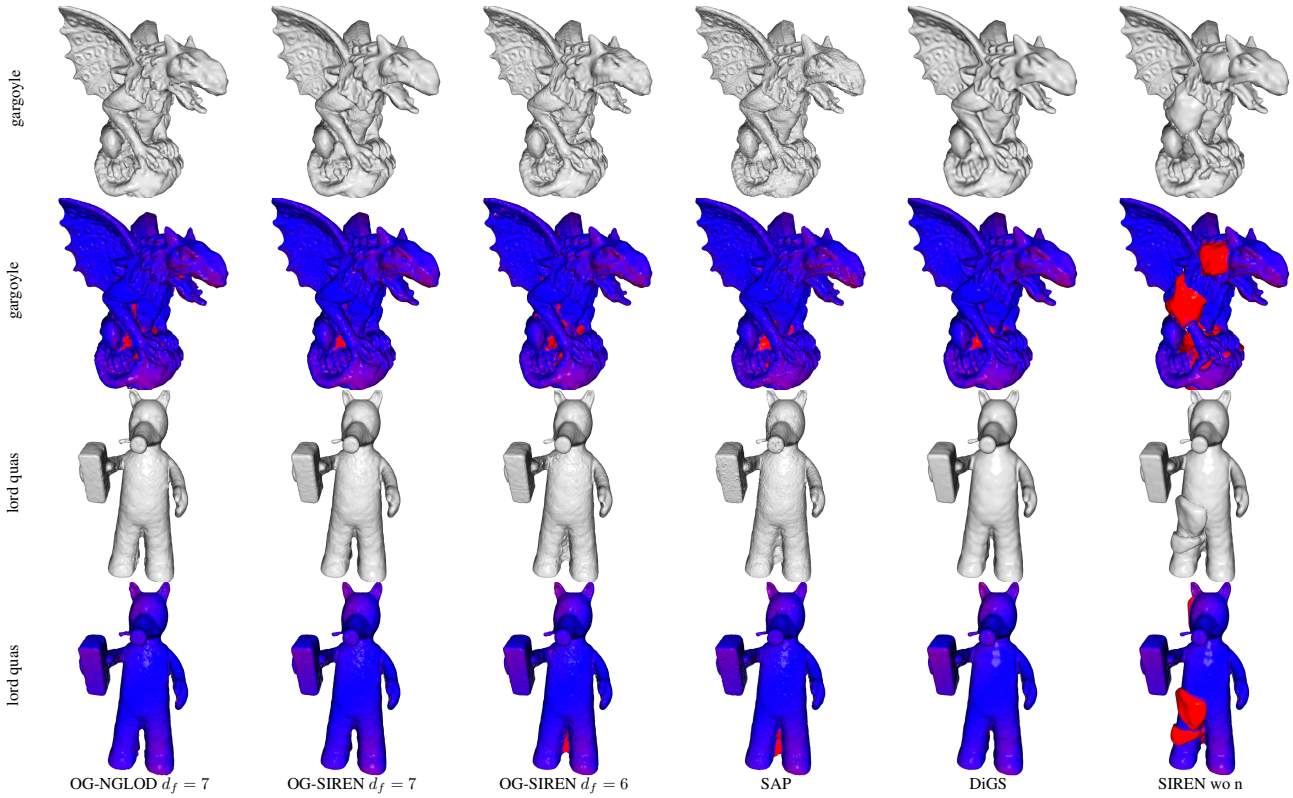

Figure 7. Qualitative results on SRB. For each shape, both the reconstructed mesh the the same mesh colored by distance to input point cloud (more red indicates high distance) are shown. See Section A.3.2 for a discussion on performance.

Squared Chamfer

Method	All			airplane			bench			cabinet			car		
	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std
SPSR [8] ††	5.44e-5	1.97e-5	5.06e-4	9.10e-6	8.74e-6	2.69e-6	2.08e-5	1.89e-5	1.01e-5	3.56e-5	3.56e-5	1.41e-5	2.41e-5	2.23e-5	8.41e-6
IGR [7] †	6.66e-4	1.07e-4	4.69e-3	3.04e-4	1.74e-4	3.47e-4	4.48e-4	2.58e-4	4.33e-4	1.56e-4	9.39e-5	1.23e-4	2.60e-4	2.82e-4	9.80e-5
SIREN [14] †	1.03e-4	5.28e-5	1.93e-4	4.15e-5	3.87e-5	8.57e-6	9.63e-5	8.12e-5	5.41e-5	1.51e-4	6.69e-5	1.77e-4	1.39e-4	9.07e-5	1.03e-4
NSP [17] †	**5.36e-5**	4.06e-5	3.64e-5	3.55e-5	3.44e-5	2.45e-6	5.66e-5	4.82e-5	2.09e-5	**6.98e-5**	4.69e-5	4.34e-5	**8.21e-5**	7.18e-5	3.60e-5
DiGS + n [3] †	2.74e-4	**2.32e-5**	9.90e-4	**1.05e-5**	**9.29e-6**	3.93e-6	**3.11e-5**	**2.17e-5**	3.98e-5	6.92e-4	**4.28e-5**	1.10e-3	3.96e-4	**3.87e-5**	1.52e-3
Nrml Est. + SPSR [8] ††	3.76e-3	4.38e-5	1.14e-2	2.97e-3	2.60e-3	3.20e-3	7.76e-3	2.52e-3	9.23e-3	5.35e-4	4.12e-5	1.11e-3	3.19e-3	6.95e-5	6.20e-3
SIREN wo n [14] ††	3.08e-4	2.58e-4	3.26e-4	2.42e-4	2.50e-4	5.92e-5	1.93e-4	1.67e-4	9.09e-5	3.16e-4	2.72e-4	1.72e-4	2.67e-4	2.58e-4	4.78e-5
SAL [1] †	1.14e-3	2.11e-4	3.63e-3	5.98e-4	2.38e-4	9.22e-4	3.55e-4	1.71e-4	4.26e-4	2.81e-4	1.86e-4	1.81e-4	4.51e-4	2.74e-4	4.36e-4
DiGS [3] †	1.32e-4	2.55e-5	4.73e-4	1.32e-5	1.01e-5	7.56e-6	7.26e-5	2.21e-5	1.74e-4	4.07e-4	4.45e-5	9.25e-4	7.89e-5	3.97e-5	1.10e-4
SAP [12] ††	4.09e-4	2.47e-5	2.36e-3	1.28e-5	**9.00e-6**	9.40e-6	6.68e-5	2.44e-5	1.04e-4	7.51e-5	3.70e-5	1.69e-4	**3.14e-5**	**2.32e-5**	2.38e-5
OG-SIREN (ours)	**3.75e-5**	**2.17e-5**	8.24e-5	**9.61e-6**	9.07e-6	3.00e-6	**5.21e-5**	**2.11e-5**	1.27e-4	**4.19e-5**	**3.74e-5**	2.37e-5	4.01e-5	2.45e-5	4.83e-5
OG-NGLOD (ours)	5.07e-5	2.57e-5	9.39e-5	1.02e-5	9.61e-6	3.12e-6	5.69e-5	2.12e-5	1.41e-4	9.22e-5	7.35e-5	5.57e-5	4.74e-5	3.62e-5	4.24e-5

Method	chair			display			lamp			loudspeaker			rifle		
	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std
SPSR [8] ††	3.11e-5	2.37e-5	1.73e-5	2.78e-5	2.35e-5	1.08e-5	1.38e-5	1.19e-5	1.00e-5	4.42e-5	4.28e-5	2.15e-5	4.03e-6	3.53e-6	1.60e-6
IGR [7] †	9.25e-4	9.88e-5	3.11e-3	9.99e-5	7.49e-5	8.44e-5	1.72e-3	1.28e-4	6.24e-3	3.77e-3	1.15e-4	1.49e-2	9.62e-5	5.29e-5	1.25e-4
SIREN [14] †	1.05e-4	6.34e-5	1.18e-4	6.98e-5	5.68e-5	3.86e-5	6.26e-5	5.07e-5	3.35e-5	2.77e-4	6.88e-5	5.54e-4	3.62e-5	3.50e-5	4.03e-6
NSP [17] †	**5.62e-5**	4.21e-5	4.32e-5	**4.36e-5**	3.99e-5	1.28e-5	4.19e-5	3.91e-5	1.00e-5	**8.41e-5**	**4.54e-5**	7.54e-5	3.26e-5	3.15e-5	2.79e-6
DiGS + n [3] †	8.55e-5	**2.43e-5**	1.43e-4	8.67e-4	**2.52e-5**	2.45e-3	**3.34e-5**	**1.70e-5**	4.80e-5	1.05e-3	7.13e-4	1.14e-3	**4.80e-6**	**4.73e-6**	1.74e-6
Nrml Est. + SPSR [8] ††	3.67e-3	3.51e-5	1.05e-2	1.39e-3	8.51e-5	4.65e-3	1.64e-2	1.59e-3	3.05e-2	3.45e-4	4.83e-5	6.36e-4	3.53e-5	3.89e-6	1.04e-4
SIREN wo n [14] ††	2.63e-4	2.60e-4	1.31e-4	2.49e-4	2.20e-4	8.45e-5	6.10e-4	3.49e-4	1.04e-3	3.29e-4	3.04e-4	1.31e-4	5.44e-4	5.56e-4	1.44e-4
SAL [1] †	1.28e-3	2.92e-4	2.05e-3	2.56e-4	8.86e-5	4.99e-4	5.86e-3	1.29e-3	9.35e-3	4.04e-4	2.63e-4	4.50e-4	2.18e-3	1.15e-4	5.17e-3
DiGS [3] †	3.72e-4	2.73e-5	1.05e-3	**3.16e-5**	2.53e-5	2.32e-5	1.70e-4	2.18e-5	3.96e-4	1.18e-4	6.18e-5	2.15e-4	9.10e-6	5.26e-6	1.03e-5
SAP [12] ††	1.34e-3	4.78e-5	3.34e-3	4.13e-5	2.48e-5	5.66e-5	1.23e-3	4.40e-5	2.45e-3	7.46e-5	4.46e-5	1.16e-4	5.38e-4	5.03e-6	1.83e-3
OG-SIREN (ours)	**3.39e-5**	2.65e-5	1.84e-5	3.51e-5	**2.39e-5**	3.31e-5	**5.69e-5**	**1.22e-5**	1.49e-4	**5.16e-5**	**4.31e-5**	3.39e-5	**4.37e-6**	**4.01e-6**	1.60e-6
OG-NGLOD (ours)	3.68e-5	**2.62e-5**	2.17e-5	4.16e-5	2.72e-5	3.93e-5	8.84e-5	3.37e-5	1.62e-4	6.52e-5	4.60e-5	4.08e-5	5.29e-6	5.41e-6	1.52e-6

Method	sofa			table			telephone			watercraft		
	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std
SPSR [8] ††	3.05e-5	2.58e-5	1.23e-5	4.37e-4	2.99e-5	1.78e-3	1.72e-5	1.68e-5	4.26e-6	1.20e-5	1.07e-5	6.04e-6
IGR [7] †	2.86e-4	1.02e-4	5.30e-4	3.40e-4	1.95e-4	3.33e-4	1.03e-4	4.43e-5	1.54e-4	1.47e-4	1.12e-4	1.23e-4
SIREN [14] †	7.88e-5	6.99e-5	3.90e-5	1.92e-4	8.32e-5	2.32e-4	3.88e-5	3.58e-5	9.64e-6	5.57e-5	4.21e-5	2.95e-5
NSP [17] †	**5.11e-5**	4.80e-5	1.24e-5	**6.60e-5**	4.88e-5	4.17e-5	**3.34e-5**	3.19e-5	3.60e-6	4.41e-5	3.84e-5	1.42e-5
DiGS + n [3] †	6.83e-5	**2.77e-5**	9.39e-5	1.68e-4	**3.26e-5**	3.50e-4	1.15e-4	**1.75e-5**	3.05e-4	**2.77e-5**	**1.57e-5**	3.30e-5
Nrml Est. + SPSR [8] ††	4.59e-3	5.03e-5	8.75e-3	2.98e-3	5.49e-5	4.98e-3	1.23e-3	1.69e-5	3.64e-3	3.79e-3	8.69e-5	1.20e-2
SIREN wo n [14] ††	2.72e-4	2.66e-4	6.74e-5	2.29e-4	2.38e-4	8.40e-5	2.10e-4	1.86e-4	6.60e-5	2.97e-4	2.43e-4	1.26e-4
SAL [1] †	3.75e-4	1.93e-4	4.31e-4	1.82e-3	5.10e-4	4.31e-3	1.04e-4	6.81e-5	7.99e-5	8.08e-4	2.06e-4	1.75e-3
DiGS [3] †	5.76e-5	3.27e-5	5.39e-5	2.94e-4	2.98e-5	6.76e-4	1.77e-5	1.74e-5	4.49e-6	6.10e-5	2.43e-5	9.03e-5
SAP [12] ††	**3.21e-5**	**2.59e-5**	1.44e-5	1.85e-3	3.53e-5	6.85e-3	**1.71e-5**	**1.68e-5**	4.24e-6	**1.47e-5**	**1.27e-5**	8.62e-6
OG-SIREN (ours)	7.20e-5	2.63e-5	1.69e-4	**5.32e-5**	**2.91e-5**	1.03e-4	1.76e-5	1.70e-5	4.36e-6	1.88e-5	1.32e-5	1.68e-5
OG-NGLOD (ours)	7.68e-5	3.49e-5	1.69e-4	6.66e-5	3.20e-5	1.30e-4	2.84e-5	1.71e-5	3.50e-5	4.27e-5	2.23e-5	6.28e-5

Table 1. Extended results for surface reconstruction on ShapeNet [6]. For each shape class, and all shapes together, we report the squared Chamfer distance (first three tables) to the ground truth mesh. Methods above the line use ground truth normal information, and methods below do not. †: results reported from NSP [17] or DiGS [3]. ††: Methods run using provided code.

IoU

Method	All			airplane			bench			cabinet			car		
	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std
SPSR [8] ††	0.9926	0.9956	0.0105	0.9945	0.9949	0.0031	0.9793	0.9859	0.0251	0.9960	0.9972	0.0027	0.9936	0.9963	0.0063
IGR [7] †	0.8102	0.8480	0.1519	0.7851	0.8193	0.0977	0.5812	0.5923	0.2487	0.8709	0.8857	0.0924	0.8026	0.8664	0.1300
SIREN [14] †	0.8268	0.9097	0.2329	0.8045	0.9080	0.2696	0.6109	0.7442	0.3258	0.8706	0.9263	0.1621	0.8036	0.9241	0.2753
NSP [17] †	0.8973	0.9230	0.0871	0.8165	0.8998	0.1551	0.7872	0.8370	0.1236	**0.9274**	0.9291	0.0422	0.8954	0.9288	0.0740
DiGS + n [3] †	**0.9200**	**0.9774**	0.1992	**0.9693**	**0.9718**	0.0151	**0.9428**	**0.9655**	0.0644	0.8323	**0.9867**	0.3076	**0.9147**	**0.9754**	0.2126
Nrml Est. + SPSR [8] ††	0.7187	0.9761	0.3767	0.4923	0.3342	0.3190	0.4654	0.2679	0.4373	0.8413	0.9952	0.2547	0.7367	0.9182	0.3297
SIREN wo n [14] ††	0.3085	0.2952	0.2014	0.2248	0.1735	0.1103	0.4020	0.4231	0.1953	0.3014	0.2564	0.1275	0.3336	0.3030	0.0997
SAL [1] †	0.4030	0.3944	0.2722	0.1908	0.1693	0.0955	0.2260	0.2311	0.1401	0.6923	0.7224	0.1637	0.6261	0.6526	0.1525
DiGS [3] †	0.9390	0.9764	0.1262	0.9613	0.9577	0.0164	0.9061	0.9536	0.1413	0.9261	0.9853	0.2137	0.9455	0.9765	0.0699
SAP [12] ††	0.9118	**0.9923**	0.2002	0.9720	0.9829	0.0274	0.8537	0.9241	0.2057	**0.9829**	**0.9976**	0.0339	0.9396	**0.9945**	0.1389
OG-SIREN (ours)	**0.9615**	0.9871	0.1048	**0.9828**	**0.9858**	0.0096	**0.9256**	**0.9665**	0.1440	0.9742	0.9951	0.0448	**0.9504**	0.9899	0.1122
OG-NGLOD (ours)	0.9593	0.9870	0.1057	0.9782	0.9789	0.0095	0.9218	0.9632	0.1515	0.9722	0.9944	0.0484	0.9472	0.9891	0.1157

Method	chair			display			lamp			loudspeaker			rifle		
	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std
SPSR [8] ††	0.9908	0.9916	0.0053	0.9955	0.9966	0.0025	0.9915	0.9934	0.0077	0.9968	0.9980	0.0025	0.9915	0.9950	0.0143
IGR [7] †	0.8049	0.8320	0.1022	0.8741	0.8917	0.0533	0.7865	0.8259	0.1318	0.8867	0.9324	0.1017	0.8279	0.8267	0.0542
SIREN [14] †	0.8721	0.8807	0.0495	0.9014	0.9146	0.0440	0.8392	0.8995	0.2025	0.8458	0.9618	0.2404	0.7329	0.9132	0.3662
NSP [17] †	0.8841	0.9034	0.0825	**0.9309**	0.9316	0.0251	**0.9037**	0.9178	0.0512	**0.9323**	0.9627	0.0599	0.9299	0.9313	0.0215
DiGS + n [3] †	**0.9719**	**0.9759**	0.0140	0.8367	**0.9855**	0.3485	0.9024	**0.9637**	0.1991	0.8798	**0.9747**	0.2424	**0.9569**	**0.9571**	0.0207
Nrml Est. + SPSR [8] ††	0.7363	0.9878	0.3894	0.8406	0.9642	0.2765	0.4661	0.2104	0.4467	0.8235	0.9959	0.3001	0.9573	0.9894	0.0945
SIREN wo n [14] ††	0.4208	0.3748	0.2322	0.3566	0.3123	0.1790	0.3055	0.2573	0.2598	0.2229	0.1724	0.1575	0.0265	0.0092	0.0554
SAL [1] †	0.2589	0.1491	0.2213	0.5067	0.5801	0.2474	0.1689	0.0698	0.1994	0.6702	0.7264	0.1976	0.2835	0.2821	0.1530
DiGS [3] †	0.9082	0.9650	0.1523	0.9839	0.9886	0.0102	0.8776	0.9646	0.1943	0.9632	0.9851	0.0978	0.9486	0.9567	0.0281
SAP [12] ††	0.8500	0.9709	0.2436	0.9682	**0.9961**	0.1065	0.6684	0.9111	0.3529	0.9536	**0.9966**	0.1089	0.8713	**0.9848**	0.2493
OG-SIREN (ours)	0.9778	0.9825	0.0153	0.9839	0.9898	0.0212	**0.8934**	**0.9832**	0.2287	**0.9693**	0.9930	0.0921	**0.9814**	0.9843	0.0106
OG-NGLOD (ours)	**0.9792**	**0.9856**	0.0189	**0.9859**	0.9919	0.0234	0.8860	0.9776	0.2204	0.9681	0.9953	0.0902	0.9659	0.9737	0.0255

Method	sofa			table			telephone			watercraft		
	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std	Mean	Median	Std
SPSR [8] ††	0.9943	0.9952	0.0032	0.9885	0.9907	0.0121	0.9961	0.9967	0.0020	0.9960	0.9963	0.0032
IGR [7] †	0.8891	0.9139	0.0708	0.6852	0.7260	0.2004	0.9148	0.9372	0.0639	0.8146	0.8445	0.0931
SIREN [14] †	0.9251	0.9411	0.0390	0.7280	0.8058	0.2089	0.9427	0.9514	0.0310	0.8722	0.9279	0.1990
NSP [17] †	0.9387	0.9473	0.0264	0.8414	0.8427	0.0534	**0.9569**	0.9625	0.0260	0.9207	0.9231	0.0402
DiGS + n [3] †	**0.9624**	**0.9859**	0.0696	**0.9284**	**0.9784**	0.1743	0.8880	**0.9855**	0.2935	**0.9747**	**0.9789**	0.0168
Nrml Est. + SPSR [8] ††	0.6774	0.9486	0.3804	0.6018	0.9816	0.4558	0.9033	0.9965	0.2799	0.8014	0.9006	0.2944
SIREN wo n [14] ††	0.3397	0.3444	0.1206	0.3797	0.3603	0.1528	0.3778	0.3806	0.2590	0.3190	0.3007	0.1877
SAL [1] †	0.4844	0.4530	0.1404	0.0965	0.0320	0.1502	0.6025	0.6704	0.2203	0.4170	0.4728	0.2367
DiGS [3] †	0.9572	0.9807	0.0896	0.8943	0.9720	0.1996	0.9854	0.9876	0.0071	0.9522	0.9735	0.0504
SAP [12] ††	**0.9748**	**0.9959**	0.0785	0.8492	0.9844	0.2810	**0.9967**	**0.9971**	0.0028	0.9725	**0.9947**	0.0440
OG-SIREN (ours)	0.9480	0.9907	0.1183	0.9401	0.9762	0.1469	0.9922	0.9930	0.0029	0.9806	0.9861	0.0186
OG-NGLOD (ours)	0.9468	0.9922	0.1240	**0.9452**	**0.9856**	0.1470	0.9941	0.9948	0.0037	**0.9807**	0.9883	0.0198

Table 2. Extended results for surface reconstruction on ShapeNet [6]. For each shape class, and all shapes together, we report the IoU compared to the ground truth mesh. Methods above the line use ground truth normal information, and methods below do not. †: results reported from NSP [17] or DiGS [3]. ††: Methods run using provided code.

	Mean		Anchor				Daratech				DC				Gargoyle				Lord Quas			
	GT		GT		Scans		GT		Scans		GT		Scans		GT		Scans		GT		Scans	
Method	d_C	d_H	d_C	d_H	$d_{\vec{C}}$	$d_{\vec{H}}$	d_C	d_H	$d_{\vec{C}}$	$d_{\vec{H}}$	d_C	d_H	$d_{\vec{C}}$	$d_{\vec{H}}$	d_C	d_H	$d_{\vec{C}}$	$d_{\vec{H}}$	d_C	d_H	$d_{\vec{C}}$	$d_{\vec{H}}$
SPSR [8] ††	0.21	4.69	0.33	7.62	0.11	1.99	0.26	6.62	0.09	1.81	0.17	2.79	0.07	2.75	0.18	4.60	0.08	2.10	0.12	1.83	0.06	1.03
DGP [16] †	0.21	5.18	0.33	8.82	**0.08**	2.79	0.20	3.14	**0.04**	1.89	0.18	4.31	**0.04**	**2.53**	0.21	5.98	**0.06**	3.41	0.14	3.67	**0.04**	2.03
IGR [7] †	0.19	2.99	0.23	4.71	0.12	1.32	0.25	4.01	0.14	1.59	0.17	2.22	0.09	2.61	0.18	**2.85**	0.10	1.29	0.12	1.17	0.07	0.98
SIREN [14] †	0.19	3.86	0.31	7.32	0.11	1.23	0.21	4.74	0.09	1.85	0.15	2.37	0.07	2.71	0.17	4.26	0.09	**0.82**	0.12	**0.62**	0.08	0.81
NSP [17] †	0.17	2.85	0.22	4.65	0.11	**1.11**	0.21	4.35	0.08	**1.14**	**0.14**	**1.35**	0.06	2.75	**0.16**	3.20	0.08	2.75	0.12	0.69	0.05	**0.62**
PHASE [9] †	**0.16**	**2.77**	**0.21**	**4.29**	0.09	1.23	**0.18**	**2.92**	0.08	1.80	0.15	2.52	0.05	2.78	**0.16**	3.14	0.07	1.09	**0.11**	0.96	0.04	0.96
DiGS + n [3] †	0.18	3.55	0.28	5.71	0.11	1.14	0.21	5.02	0.09	1.75	0.15	2.13	0.06	2.74	**0.16**	3.81	0.09	0.90	0.12	1.10	0.06	0.77
Nrml Est. + SPSR [8] ††	1.25	22.59	0.35	7.65	0.11	3.49	3.12	48.91	0.13	2.87	1.84	32.73	0.09	2.77	0.79	21.77	0.10	4.29	0.13	1.89	0.06	1.00
IGR wo n [7] †	1.38	16.33	0.45	7.45	0.17	4.55	4.90	42.15	0.70	3.68	0.63	10.35	0.14	3.44	0.77	17.46	0.18	2.04	0.16	4.22	0.08	1.14
SIREN wo n [7] †	0.42	7.67	0.72	10.98	0.11	1.27	0.21	4.37	0.09	1.78	0.34	6.27	0.06	2.71	0.46	7.76	0.08	**0.68**	0.35	8.96	0.06	0.65
SAL [7] †	0.36	7.47	0.42	7.21	0.17	4.67	0.62	13.21	0.11	2.15	0.18	3.06	0.08	2.82	0.45	9.74	0.21	3.84	0.13	4.14	0.07	4.04
IGR+FF [7] †	0.96	11.06	0.72	9.48	0.24	8.89	2.48	19.60	0.74	4.23	0.86	10.30	0.28	3.98	0.26	5.24	0.18	2.93	0.49	10.7	0.14	3.71
PHASE+FF [9] †	0.22	4.96	**0.29**	7.43	**0.09**	1.49	0.35	7.24	**0.08**	**1.21**	0.19	4.65	**0.05**	2.78	**0.17**	4.79	**0.07**	1.58	**0.11**	**0.71**	**0.05**	0.74
DiGS [3] †	**0.19**	**3.52**	**0.29**	7.19	0.11	**1.17**	**0.20**	3.72	0.09	1.80	**0.15**	**1.70**	0.07	2.75	**0.17**	**4.10**	0.09	0.92	0.12	0.91	0.06	0.70
SAP [12] ††	0.21	4.51	0.33	7.82	0.10	1.62	0.23	3.07	0.09	1.40	0.18	3.31	0.07	**2.60**	0.19	5.06	0.08	1.13	0.14	3.30	0.06	1.11
Our OG-SIREN	0.20	4.06	**0.29**	7.56	0.12	1.36	0.23	**2.89**	0.10	1.03	0.17	2.68	0.08	2.71	0.19	5.01	0.10	1.48	0.13	2.14	0.06	**0.64**
Our OG-NGLOD	0.22	6.03	**0.29**	**7.17**	0.12	1.20	0.23	3.85	0.11	1.35	0.18	6.74	0.07	2.72	0.20	5.61	0.09	1.02	0.19	6.78	0.06	**0.64**
Our OG-NGLOD + H	0.19	2.97	0.25	5.12	0.12	1.07	0.22	2.67	0.10	1.76	0.18	2.29	0.07	1.10	0.17	2.57	0.09	1.15	0.13	2.17	0.06	0.82
Our OG-NGLOD + GT	0.17	2.44	0.23	4.45	0.12	1.18	0.17	2.98	0.10	1.67	0.15	1.41	0.08	2.77	0.17	2.61	0.10	0.77	0.12	0.76	0.06	0.79

Table 3. Results on the Surface Reconstruction Benchmark using Chamfer d_C, Hausdorff distance d_H. We compare methods with normal supervision above the line and without normal supervision below the line, while the last two results are with extra supervision (Human and Ground Truth). The *GT* column reports the two sided distances between the reconstruction and the dense, uniform ground truth. The *Scans* column reports the one sided distances ($d_{\vec{C}}, d_{\vec{H}}$) between the reconstruction and the simulated scans, which gives a measure of the reconstruction's overfit to the noisy input. †: results reported from NSP [17], DiGS [3] or PHASE [9].

	Final Mesh		Octree Build+Label Time (s)		
Depth	$d_C \downarrow$	$d_H \downarrow$	Mean	Median	Std
5	0.40	5.59	3.3	2.4	1.6
6	0.21	3.97	16.7	17.2	7.3
7	0.20	4.06	79.4	87.5	30.6

Table 4. Octree depth ablations on SRB using OG-SIREN.

.

A.4. Further Implementation Details

Energy function We use a fairly small λ ($\lambda = 10^{-3}$) as want to ensure that the surface property is maintained, and then minimise surface area subject to that. We use $\gamma^{(0)} = 2$, $\gamma^{(1)} = 10$, $\eta^{(0)} = 0.25$ and $\eta^{(1)} = 1$ (see Section A.2 for what these mean in relation to our 2D example).

Move Making Algorithm We implement our algorithm in Cython [2], and compile the individual move making functions to C++. We start at an initial depth of $d_i = 3$ and end with a final depth of $d_f = 7$ (so $k = 4$). For move making, we implement both a function for the energy of a given labelling on the whole octree, and a cost difference function for the change in energy of changing one label given the rest of the labelling. The latter allows us to keep track of the change in energy when changing labels one at a time. Our set of move sizes are $C = [1, 2, 10, 1000]$. For multi-leaf ($s > 1$) moves we have additional stopping criteria when traversing potential moves from an initial inside leaf on the border: if the energy has only increased in the last 100 moves, or if the energy has increased to more than 100 compared to before the first change, we terminate the move early.

SDF Loss Function For $\mathcal{L}_4$ we sample 30 points inside each surface leaf, and 10 points inside each non-surface leaf. If this is more than half of our batch size b, we randomly choose $b/4$ from each of these. The remaining points are randomly sampled from a 256^3 grid of the domain and used for $\mathcal{L}_3$. $\mathcal{L}_1$ uses all of the points from the initial point cloud χ irrespective of b, and $\mathcal{L}_2$ uses the points that are used for both $\mathcal{L}_2$ and $\mathcal{L}_3$.

We vary the scaling of our losses during training. Details of this can be seen in the Training Details section.

Architecture Details Our SIREN architectures has 4 layers with 256 units in each layer. Note that we are only using the SIREN architecture, not the SIREN losses (which include requiring ground truth normals), so OG-SIREN is the method SIREN without normals ("SIREN wo") with adding on supervision from the octree. Our NGLOD architectures has one LOD of depth 7 (129^3 parameter vectors), with the encoding parameter vectors each being 32 dimensional. The small ReLU end network is a one hidden layer ReLU network with a hidden dimension of size 128.

Training Details For OG-NLGOD we use a batch size of 400k, and run for 300 iterations. We start of with $(\lambda_1, \lambda_2, \lambda_3, \lambda_4) = (3e3, 5e1, 5e4, 1e2)$, and change λ_3, λ_4 to 1e4,1e1 at 100 iterations and 1e2,1e-1 at 200 iterations. The learning rate is set to 1e-3 for the first 200 iterations and 5e-4 for the remaining iterations.

For OG-SIREN we use a batch size of 100k (which is as much as we can fit in our GPU), and run for 600 iterations. We use the same values of the λ_i as OG-NGLOD and change λ_3, λ_4 identically. The learning rate is set to 1e-3 for the first 200 iterations, 5e-4 for the next 200 iterations and 5e-5 for the remaining iterations.

References

[1] Matan Atzmon and Yaron Lipman. SAL: Sign Agnostic Learning of shapes from raw data. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2565–2574, 2020. 12, 13

[2] Stefan Behnel, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, and Kurt Smith. Cython: The best of both worlds. *Computing in Science & Engineering*, 13(2):31–39, 2011. 15

[3] Yizhak Ben-Shabat, Chamin Hewa Koneputugodage, and Stephen Gould. Digs: Divergence guided shape implicit neural representation for unoriented point clouds. *arXiv preprint arXiv:2106.10811*, 2021. 1, 12, 13, 14

[4] Matthew Berger, Joshua A Levine, Luis Gustavo Nonato, Gabriel Taubin, and Claudio T Silva. A benchmark for surface reconstruction. *ACM Trans. on Graphics (ToG)*, 32:1–17, 2013. 1

[5] Jonathan C Carr, Richard K Beatson, Jon B Cherrie, Tim J Mitchell, W Richard Fright, Bruce C McCallum, and Tim R Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proc. of the Conference on Computer Graphics and Interactive Techniques*, pages 67–76, 2001. 1

[6] 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. 12, 13

[7] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit Geometric Regularization for learning shapes. In *Proc. of the International Conference on Machine Learning (ICML)*, volume 119, pages 3789–3799. PMLR, 2020. 1, 12, 13, 14

[8] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. on Graphics (ToG)*, 32:1–13, 2013. 1, 6, 12, 13, 14

[9] Yaron Lipman. Phase transitions, distance functions, and implicit neural representations. *arXiv preprint arXiv:2106.07689*, 2021. 1, 14

[10] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4460–4470, 2019. 1

[11] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, 2019. 1

[12] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape As Points: A differentiable poisson solver. *arXiv preprint arXiv:2106.03452*, 2021. 1, 6, 12, 13, 14

[13] Bernhard Schölkopf, Joachim Giesen, and Simon Spalinger. Kernel methods for implicit surface modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, NIPS'04, page 1193–1200, Cambridge, MA, USA, 2004. MIT Press. 1

[14] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020. 12, 13, 14

[15] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[16] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep Geometric Prior for surface reconstruction. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10130–10139, 2019. 1, 14

[17] Francis Williams, Matthew Trager, Joan Bruna, and Denis Zorin. Neural Splines: Fitting 3D surfaces with infinitely-wide neural networks. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9949–9958, 2021. 1, 6, 12, 13, 14

[18] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 1, 6