6 Supplementary Material

6.1 Training Setup

Optimization. In all of our experiments, we train the MLPs in PyTorch using the Adam optimizer with the default parameters. We supervise the network based on the mean squared error loss between reconstructed color and ground truth color. We adopt the cosine annealing learning rate scheduler, with initial learning rate as 10^{-4} and final learning rate as 10^{-7}. We randomly shuffled all the training samples and divided them into batches of size equal to 1,024. To ensure reproducibility, we set the random seed as 0.
On a machine with NVIDIA GeForce RTX 2080 TI, training 10 epochs takes around 4 minutes for the SDF and OF condition, and 13 minutes for PRIF.

Network Architecture. We set all networks to have 10 layers with residual connections, where the 8 intermediate layers having a dimension size of 512×512. We apply layer normalization after all MLP layers except the final one. When we further extend the network to model color or background mask, we simply invoke a second network with the same architecture, as network compression rate or total parameter count is not the main focus of the current work.

6.2 Runtime Discussions

Rendering Speed. The formulation of PRIF inherently allows us to bypass the inefficient multi-sample issue of the previous methods that adopt level-set functional representations. To more quantitatively measure the speed-up, we measure the number of network evaluations (Queries) and the actual time to render a 512×512 depth image from a trained neural representation on the same machine with an NVIDIA GeForce RTX 2080 Ti graphics card. We first consider

Method	Steps	Queries	Time (s)
DeepSDF + Sphere Tracing	100	2,453,700	1.01070
DeepSDF + DIST Acceleration	100	382,325	0.30539
PRIF	**1**	**262,144**	**0.00578**

Table 4: Quantitative results on the cost to render a 512×512 depth image from trained neural shape representations. *Queries* refer to the number of network evaluations, and *Steps* refer to the maximum sphere tracing steps specified in the actual implementation. *Steps* = 100 is in line with the original DIST implementation from Liu *et al.* [28].

a baseline method where a trained DeepSDF network is used to perform naive sphere tracing. We then consider an improved baseline where the DeepSDF-based

sphere tracing is sped up by various techniques proposed by Liu *et al.* [28]. Both baselines are adopted from the implementation of Liu *et al.*, and correspond to two conditions (trivial and recursive) in their implementation.

Meshing Speed The overall meshing speed depends on *network inference* + *surface reconstruction*. We examine the trade-off by decoding the same shape with PRIF and SDF trained with networks of equivalent capacity. Our method spends 1.01s at inference, then 2.65s to obtain the triangle mesh from MeshLab (Poisson with normal estimation). The SDF-based network (similar to DeepSDF) spends 6.97s for inference, then 1.19s for Marching Cubes. We also note that many practical scenarios do not require the meshing process, such as camera pose registration and neural rendering.

6.3 More Ablations

Comparison to Plücker Coordinate. We use the original Plücker coordinate for the shape generation task for the *Car* category. Below are the results (with mean/median Chamfer distance):

Method	Car	Chair	Table	Plane	Lamp	Sofa
Plain Plücker	2.397|0.547	2.603|0.531	5.054|0.554	0.766|0.199	4.838|1.094	1.738|0.305
Proposed	**1.961|0.347**	**0.982|0.267**	**4.532|0.315**	**0.389|0.125**	**3.276|0.534**	**1.236|0.222**

Although the plain Plücker formulation works well for single shape representation (Section 5 Table 3), it performs much worse in the generative task. We find the network trained with Plücker coordinates often fails to predict correct background masks, leading to many noisy points and worse performance overall. Compared to the Plücker moment vector, our proposed reference points is better geometrically related to the surface hitpoint, which is our overarching goal of shape representation and rendering.

Varying Model complexity. We examine the effect of varying the model capacity (by changing MLP layer width) for the generation task on shapes from the *Car* category.

# Param (in Millions)	2.96	2.23	1.58	1.35	SDF (2.10)
Mean|Median CD	1.961|0.347	2.026|0.364	2.058|0.368	2.055|0.386	2.315|0.495

Results show we can decrease the number of parameters and still achieve better performance than SDF.

Different Noise Levels. We perform additional denoising experiments (as Fig. 5 in the paper) and present the impact of different noise levels on the shape representation accuracy on the *Car* category (measured by Chamfer distance):

Noise Level	0.01	0.02	0.04	0.08	0.1
Mean|Median CD	1.929|0.368	1.931|0.428	2.005|0.590	2.471|0.951	2.842|1.123

6.4 Additional Results

Fig. 11: **More Single Shape from Different Angles.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

Fig. 12: **More Single Shape from Different Angles.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

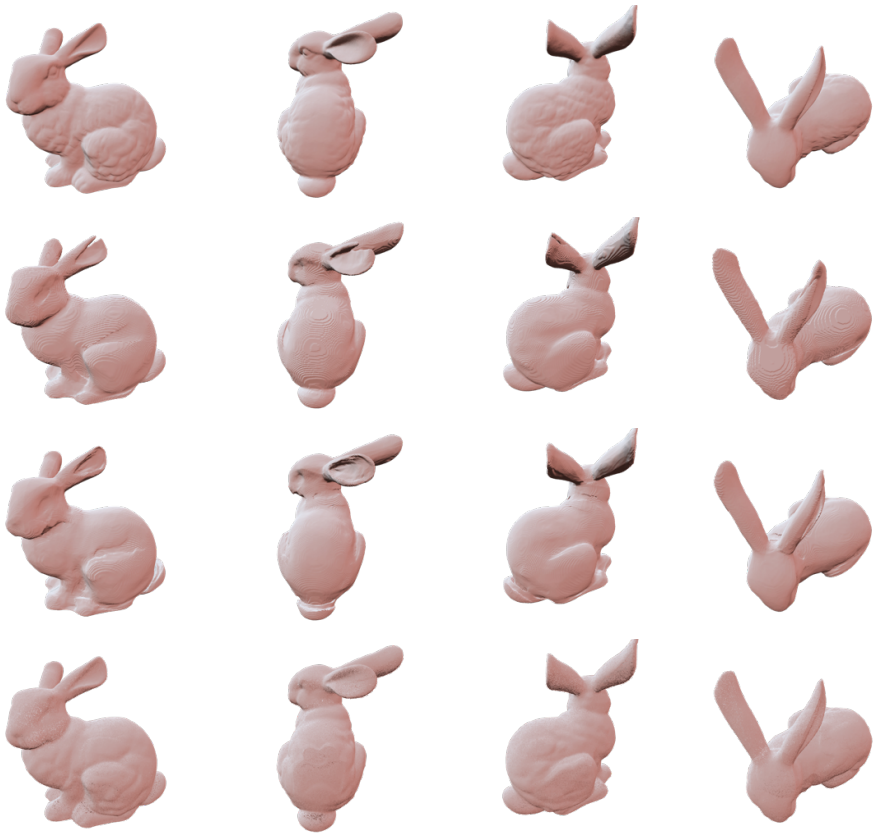

Fig. 13: **More Single Shape from Different Angles.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

Fig. 14: **More Single Shape from Different Angles.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

Fig. 15: **More Single Shape from Different Angles.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

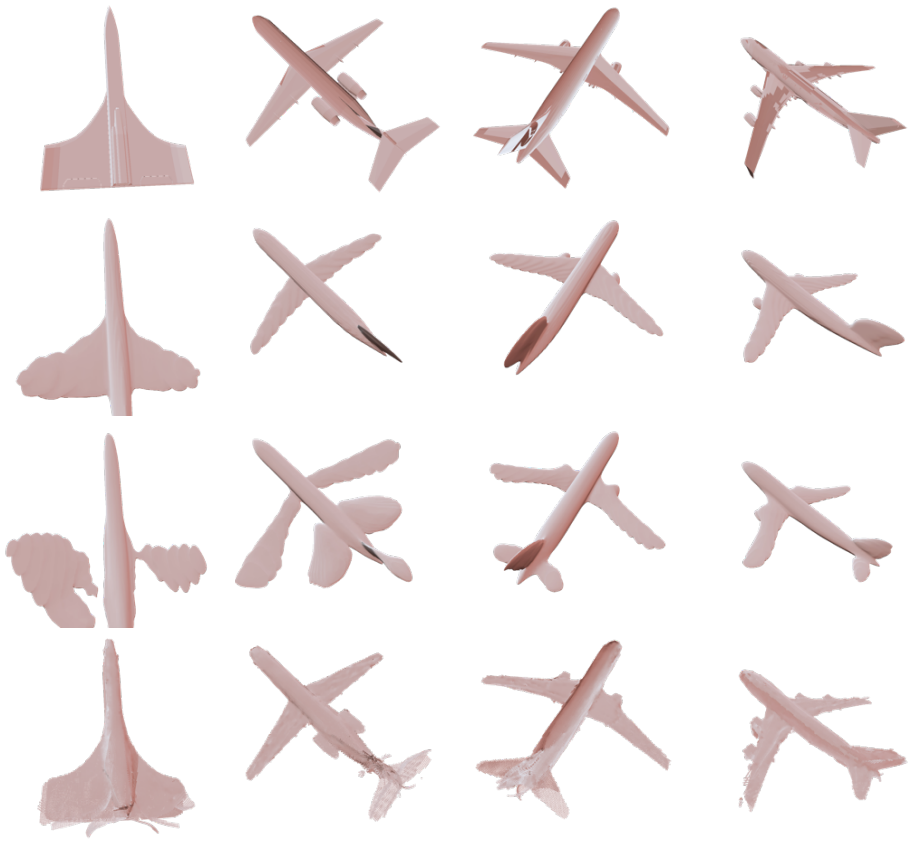

Fig. 16: **More Generative Results from Different Categories.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

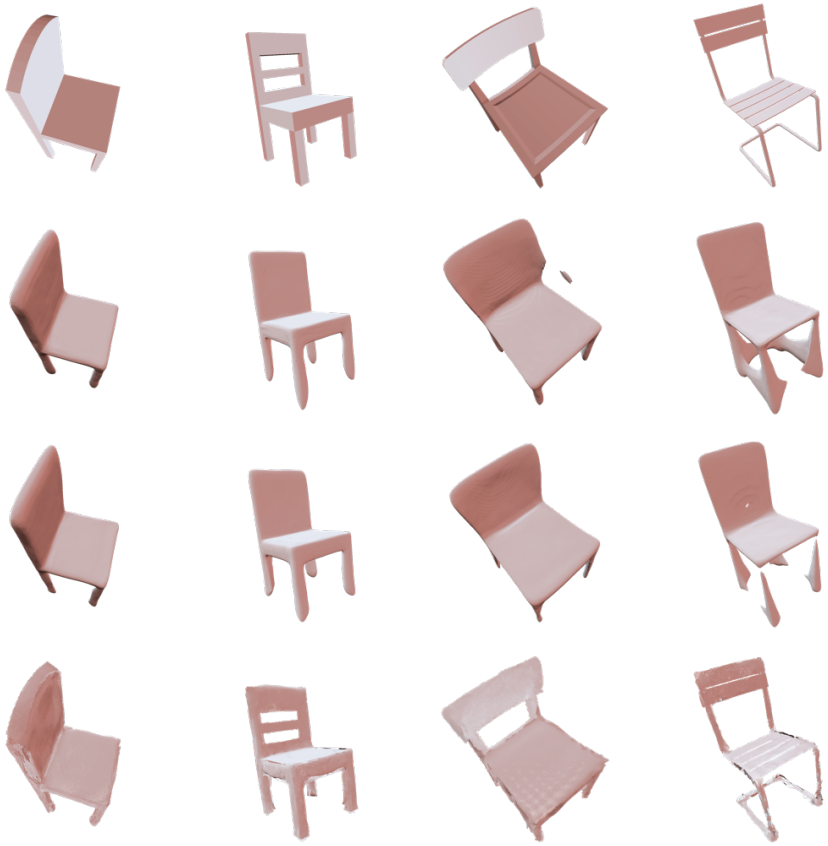

Fig. 17: **More Generative Results from Different Categories.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

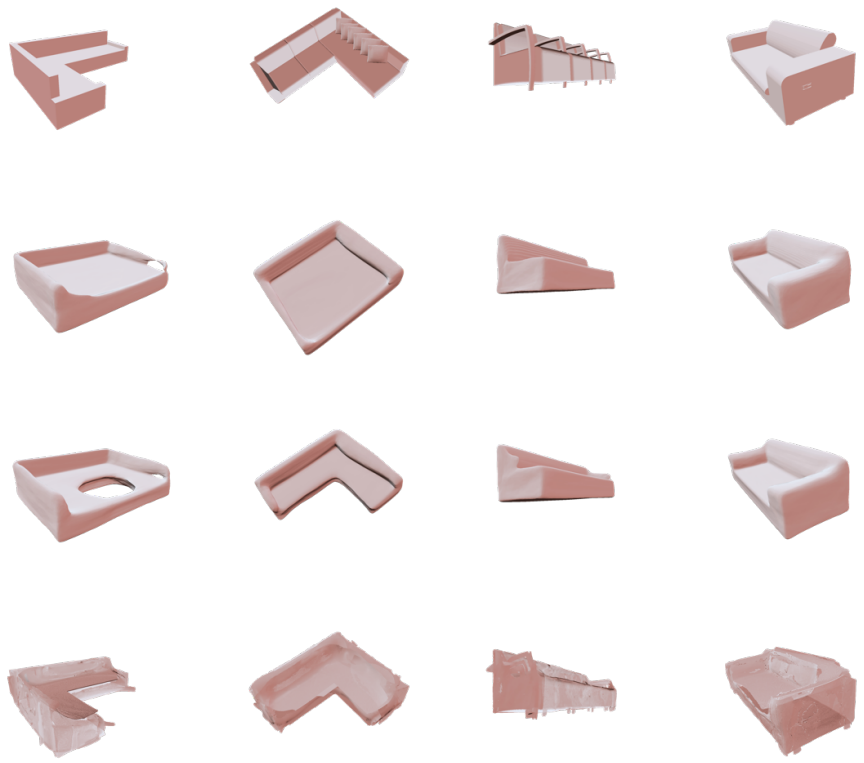

Fig. 18: **More Generative Results from Different Categories.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

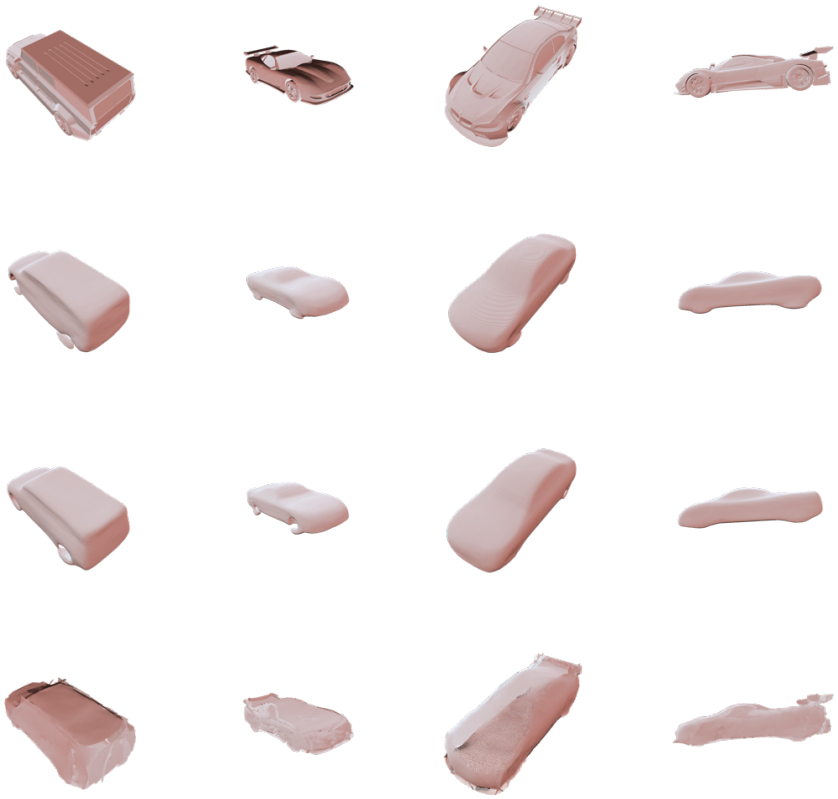

Fig. 19: **More Generative Results from Different Categories.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

Fig. 20: **More Generative Results from Different Categories.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

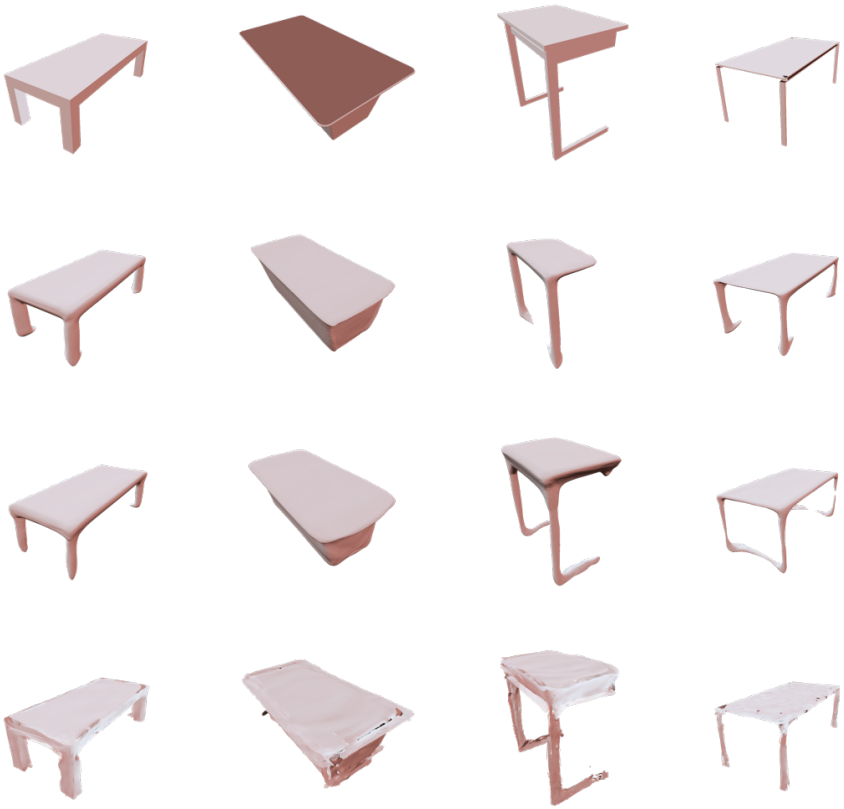

Fig. 21: **More Generative Results from Different Categories.** Results from Rows 1 to 4 correspond to **Reference**, **OF**, **SDF**, **PRIF - Mesh**.

Fig. 22: **More Color Rendering Results.** Depth (left) and Color (right) renderings from the same view point are shown.

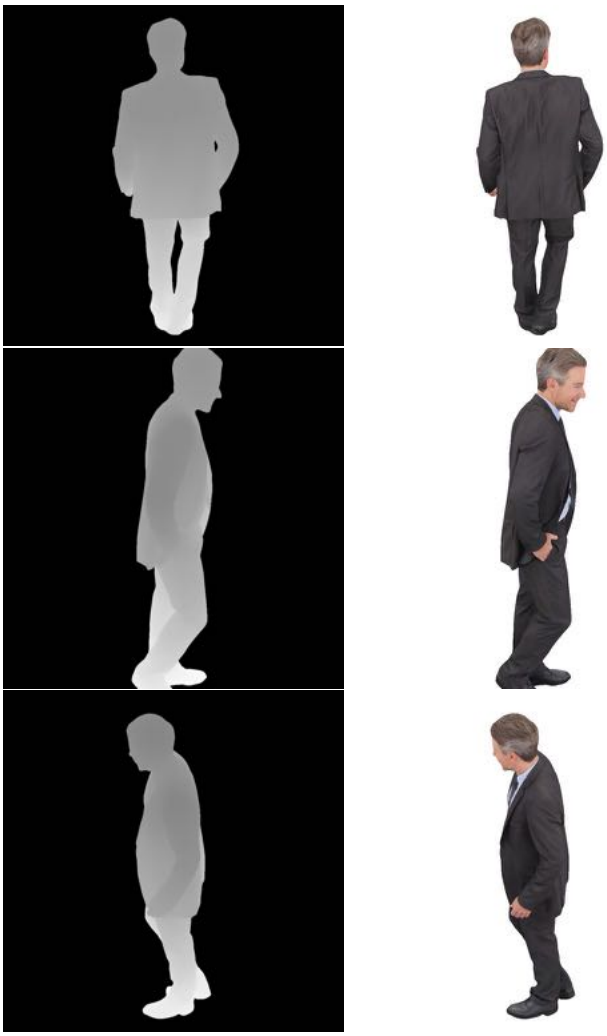

Fig. 23: **More Color Rendering.** Depth (left) and Color (right) renderings from the same view point are shown.