

---

# MultiPull: Detailing Signed Distance Functions by Pulling Multi-Level Queries at Multi-Step

---

Takeshi Noda<sup>1\*</sup> Chao Chen<sup>1\*</sup> Weiqi Zhang<sup>1</sup> Xinhai Liu<sup>2</sup>  
Yu-Shen Liu<sup>1†</sup> Zhizhong Han<sup>3</sup>

<sup>1</sup>School of Software, Tsinghua University, Beijing, China

<sup>2</sup>Tencent, Hunyuan, Beijing, China

<sup>3</sup>Department of Computer Science, Wayne State University, Detroit, USA  
yeth21@mails.tsinghua.edu.cn chencha19@tsinghua.org.cn  
zwq23@mails.tsinghua.edu.cn adlerxhliu@tencent.com  
liuyushen@tsinghua.edu.cn h312h@wayne.edu

## Abstract

Reconstructing a continuous surface from a raw 3D point cloud is a challenging task. Recent methods usually train neural networks to overfit on single point clouds to infer signed distance functions (SDFs). However, neural networks tend to smooth local details due to the lack of ground truth signed distances or normals, which limits the performance of overfitting-based methods in reconstruction tasks. To resolve this issue, we propose a novel method, named MultiPull, to learn multi-scale implicit fields from raw point clouds by optimizing accurate SDFs from coarse to fine. We achieve this by mapping 3D query points into a set of frequency features, which makes it possible to leverage multi-level features during optimization. Meanwhile, we introduce optimization constraints from the perspective of spatial distance and normal consistency, which play a key role in point cloud reconstruction based on multi-scale optimization strategies. Our experiments on widely used object and scene benchmarks demonstrate that our method outperforms the state-of-the-art methods in surface reconstruction. Project page: <https://takeshie.github.io/MultiPull>

## 1 Introduction

Reconstructing surfaces from 3D point clouds is an important task in computer vision. It is widely used in various real-world scenarios such as autonomous driving, 3D scanning and other downstream applications. Recently, using neural networks to learn signed distance functions from 3D point clouds has made huge progress [1, 2, 3, 4, 5, 6, 7, 8]. An SDF represents a surface as the zero-level set of a 3D continuous field, and the surface can be further extracted using the marching cubes algorithm [9]. In supervised methods [10, 11, 12, 13], a continuous field is learned using signed distance supervision. Some methods employ multi-level representations [14, 15], such as Fourier layers and level of detail (LOD) [16, 17], to learn detailed geometry. However, these methods require 3D supervision, including ground truth signed distances or point normals, calculated on a watertight manifold. To address this issue, several unsupervised methods [18, 19, 20, 21, 22] were proposed to directly infer an SDF by overfitting neural networks on a single point cloud without requiring ground truth signed distances and point normals. They usually need various strategies, such as geometric constraints [18, 19, 20] and consistency constraints [22, 23], for smoother and more completed signed distance field. However, the raw point cloud is a highly discrete approximation of the surface, learning

---

\*Equal contribution. †The corresponding author is Yu-Shen Liu.

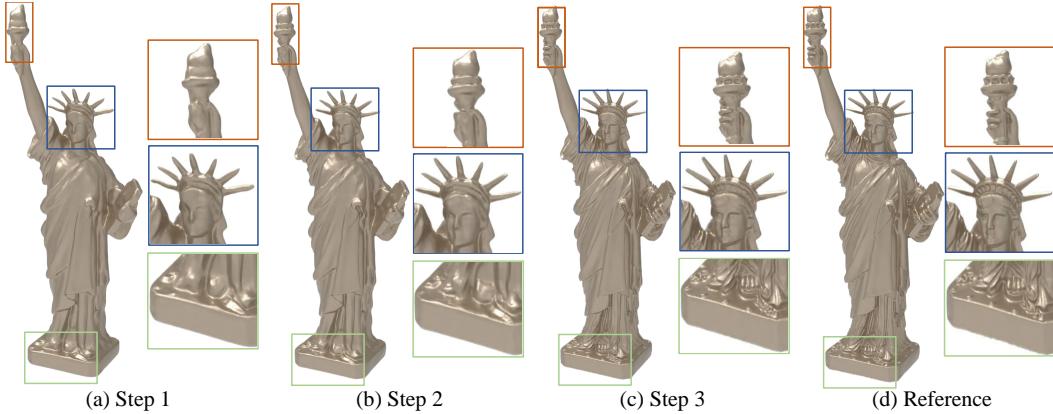


Figure 1: Visualization of the 3D shape reconstruction. In (a), (b) and (c), SDFs are learned from a point cloud by optimizing multi-level query points at multi-step. At each step, we optimize query points at one level with frequency features at this specific level as conditions. This enables the network to progressively recover coarse-to-fine geometry details.

SDFs directly from the point cloud is often inaccurate and highly ambiguous. This makes it hard for the network to learn accurate SDFs on local details, resulting in over-smooth reconstruction.

To address this issue, we propose *MultiPull*, to learn an accurate SDF with multi-scale frequency features. It enables network to predict SDF from coarse to fine, significantly enhancing the accuracy of the predictions. Furthermore, to optimize the SDF at different scales simultaneously, we introduce constraints on the pulling process. Specifically, given query points sampled around 3D space as input, we use a Fourier transform network to represent them as a set of Fourier features. Next, we design a network that can leverage multi-scale Fourier features to learn an SDF fields from coarse to fine. To optimize the signed distance fields with multi-scale features, we introduce a loss function based on gradient consistency and distance awareness. Compared with Level of Detail (LOD) methods [16, 17, 24], we can optimize the signed distance fields effectively without a need of signed distance supervision, recovering more accurate geometric details. Evaluations on widely used benchmarks show that our method outperforms the state-of-the-art methods. Our contribution can be summarized as follows.

- We propose a novel framework that can directly learn SDFs with details from raw point clouds, progressing from coarse to fine. This provides a new perspective for recovering 3D geometry details.
- We introduce a multi-level loss function based on gradient consistency and distance awareness, enabling the network to geometry details.
- Our method outperforms state-of-the-art methods in surface reconstruction in terms of accuracy under widely used benchmarks.

## 2 Related Work

Classic methods for geometric modeling [25, 26, 27, 28] have attempted to analyze the geometric modeling of objects, which do not require large-scale datasets. With the advent of extensive and intricate 3D datasets like ShapeNet [29] and ABC [30], learning-based methods have achieved significant advancements [18, 12, 31, 32, 22, 33, 34, 35, 36, 37, 38]. These approaches learn implicit representations from various inputs, including multi-view images [39, 40, 41, 42, 43, 44], point clouds [45, 46, 47], and voxels [48, 49, 50].

**Learning Implicit Functions with Supervision.** Supervised methods have made significant progress in recent years. These methods leverage deep learning networks to learn priors from datasets or use real data for supervision [10, 11, 51, 52, 53, 54] to improve surface reconstruction performance. Some supervised approaches use signed distances and point normals as supervision, or leverage occupancy labels to guide the network’s learning process. In order to improve the generalization ability of neural networks and learn more geometric details, some studies learn geometry prior of shapes through supervised learning.

**Learning Implicit Functions with Conditions.** To alleviate the dependence on supervised information, recent studies focus on unsupervised implicit reconstruction methods. These methods do not require pretrained priors during optimization. For example, NeuralPull (NP) [21] learns SDF by pulling query points in nearby space onto the underlying surface, which relies on the gradient field of the network. CAP [55] further complements this by forming a dense surface by additionally sampling dense query points. GridPull [23] generalizes this learning method to the grid, by pulling the query point using interpolated signed distances on the grid. In addition, some studies explore surface reconstruction more deeply and propose innovative methods, such as utilizing differentiable Poisson solutions [56], or learning signed [57, 58, 19, 51] or unsigned functions [59, 55] with priors. However, inferring implicit functions without 3D supervision requires a lengthy convergence process, which limits the performance of unsupervised methods on large-scale point cloud datasets. To address this, we propose a fitting-based frequency feature learning strategy that efficiently learns implicit fields without the need for additional supervision.

**Learning Implicit Functions with LOD.** Level-Of-Detail (LOD) models [16, 17, 24] are used to simplify code complexity and refine surface details through the architecture of multi-level outputs. Previous studies have explored multi-scale architectures in various reconstruction tasks. For example, NGLOD [16] uses octree-based feature volumes to represent implicit surfaces, which can adapt to shapes with multiple discrete levels of detail and enable continuous level-of-detail switching through SDF interpolation. MFLOD [17] applies Fourier layers to LOD, which can offer better feasibility in Fourier analysis. However, it is difficult to optimize multi-level features simultaneously to learn 3D shapes. To address this issue, we propose a novel strategy to optimize multi-level frequency features, allowing the network to progressively learn geometric details from coarse to fine.

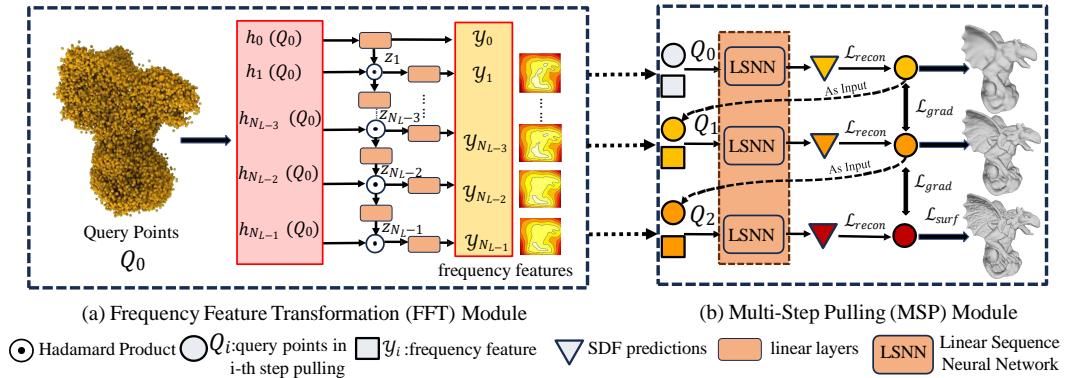


Figure 2: Overview of our method: (a) Frequency Feature Transformation (FFT) module and (b) Multi-Step Pulling (MSP) module. In (a), we learn Fourier bases  $h_i(Q)$  from query points  $Q$  using the Fourier layer and obtain multi-level frequency features  $y_i$  through Hadamard product. In (b), using multi-level frequency features from (a) and a linear network **LSNN** with shared parameters, we calculate the distance(D) of  $Q_i$  to its corresponding surface target point  $Q_t$  to predict a more accurate surface. We visualize the predicted SDF distribution map corresponding to the frequency features in (a) and the reconstruction from each step of SDF predictions on the right side of (b).

### 3 Method

**Overview.** The overview of MultiPull is shown in Fig. 2. We design a neural network to learn an implicit function  $f$  from a single 3D point cloud by progressively pulling a set of query points  $Q_0$  onto the underlying surface, where  $Q_0$  is randomly sampled around the raw point cloud  $S$ . Our network mainly consists of two parts as follows.

(1) The Frequency Feature Transformation (FFT) Module (Fig. 2(a)) aims to convert the query points  $Q_0$  into a set of multi-level frequency features  $Y = \{y_i, i \in [0, N_L - 1]\}$ . The key insight for introducing frequency features lies in a flexible control of the degree of details. (2) The Multi-Step Pulling (MSP) Module (Fig. 2(b)) is designed to predict  $f$  with coarse-to-fine details under the guidance of frequency features  $Y$ . At the  $i$ -th step, we pull  $Q_i$  to  $Q_{i+1}$  using the predicted signed distances  $s_i = f(Q_i, y_i)$  and the gradients at  $Q_i$ , according to its feature  $y_i$ . To this end, we constrain query points to be as close to their nearest neighbor point on  $S$ .

### 3.1 Frequency Feature Transformation (FFT) Module

We introduce a neural network to learn frequency features  $Y$  from point clouds. The network manipulates input  $Q_0$  through several linear layers to obtain an initial input  $z_0$  and a set of Fourier basis  $h_i(Q_0)$ ,  $i \in [0, N_L - 1]$ , formulated as follows.

$$\begin{cases} h_i(Q_0) = \sin(\omega_i Q_0 + \phi_i), \\ z_0 = h_0(Q_0), \end{cases} \quad (1)$$

where  $\omega_i$  and  $\phi_i$  are the parameters of the network, and  $N_L$  is the number of layers of the network.

To effectively represent the expression of the raw input in the frequency space, we choose the sine function as the activation function and employ the Hadamard product to compute the intermediate frequency feature output. Since the Hadamard product allows the representation of frequency components as the product of two feature inputs, denoted as  $a$  and  $b$ , which can be formulated as:

$$\sin(a) \sin(b) = \frac{1}{2}(\sin(a + b - \frac{\pi}{2}) + \sin(a - b + \frac{\pi}{2})). \quad (2)$$

Through Eq. (2), we can calculate the frequency component  $z_i$  of  $h_i(Q_0)$ , and then obtain the output  $y_i$  of the  $i$ -th layer, formulated as:

$$\begin{cases} z_i = h_i(Q_0) \odot (W_i z_{i-1} + b_i), i \in [1, N_L - 1], \\ y_i = W_i z_i + b_i, \end{cases} \quad (3)$$

where  $\odot$  indicates the Hadamard product,  $W_i, b_i$  are parameters of the network.

Frequency networks based on the Multiplication Filter Network (MFN) [14] typically employ uniform or fixed-weight initialization for network parameters in practice. This approach overlooks the issue of gradient vanishing in deep network layers during the training process, leading to underfitting and making the network overly sensitive to hyperparameter changes. To address this challenge, we propose a new initialization scheme that thoroughly considers the impact of network propagation, aiming at ensuring a uniform distribution of initial parameters. Specifically, we dynamically adjust initial weights, which can be formulated as:

$$\Psi_i = \sqrt{\eta \times \sin(i\pi/N_L)}, i \in [1, N_L - 1], \quad (4)$$

where  $N_L$  and  $\eta$  are the number of layers and the parameters of the network, respectively. We leverage the standard deviation as the initialization range to ensure that the parameters in Eq. (3) are within a reasonable range. As shown in Fig. 3, we compared the parameter distributions of different linear layers. The initialization scheme based on MFN results in gradient vanishing and small activations in deeper linear layers. In contrast, our initialization scheme ensures that the parameters of each linear layer follow a standard normal distribution.

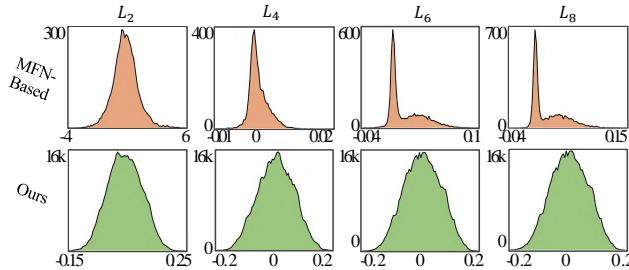


Figure 3: Comparison of parameter distributions of different linear layers especially in  $(L_2, L_4, L_6, L_8)$ . We show the different initialization strategies on the results of the reconstruction task and the visualization effects in Appendix B.

### 3.2 Multi-Step Pulling (MSP) Module

In Fig. 2(b), we demonstrate our idea of learning an accurate implicit function  $f$  with multiple frequency features. Given a set of frequency feature  $Y$ , we use frequency features  $y_i$  in  $Y$  as the

input along with query points  $Q_i$  for the MSP module. We follow NP[21] to construct initial query points and calculate the stride and direction of  $Q_i$  at  $i$ -th step for pulling it to the target surface point. Furthermore, we use the direction of the gradient as  $\nabla f(Q_i, y_i)$  and signed distance  $f(Q_i, y_i)$  for the pulling, where  $\nabla f(Q_i, y_i)$  represents the fastest increase in signed distance in 3D space, pointing away from the surface direction. Therefore,  $Q_i = Q_{i-1} - f(Q_{i-1}, y_{i-1}) \cdot \nabla f(Q_{i-1}, y_{i-1}) / \|\nabla f(Q_{i-1}, y_{i-1})\|_2$ . For each step of pulling the query points  $Q_i$ , it corresponds to a nearest point  $q_i$  on the surface, and the distance between query points and surface points can be described as  $D_i = \|Q_i - q_i\|_2^2$ . Based on this, we initiate the optimization by pulling query points  $Q_i$  the target points  $q_i$  progressively. Therefore, we can obtain the combined loss  $\mathcal{L}_{\text{pull}}$  under optimal conditions:

$$\mathcal{L}_{\text{pull}} = \sum_{i=1}^I D_i, i \in [1, I], \quad (5)$$

where  $I$  is the step of moving operation. However, optimizing all query points accurately through this equation alone is challenging when merely constraining surface points. This is because the query points  $Q_i$  may be located across multiple spatial scales with inconsistent gradient directions, indicating that simultaneous optimization becomes challenging. Consequently, some outlier points may not be effectively optimized. Additionally, for sampling points near target points, some surface constraints are required to enable the network to accurately predict their corresponding zero level-set to avoid optimization errors. Therefore, we will further advance Eq. (5) from the perspectives of distance constraints, gradient consistency, surface constraints in Sec. 3.3 to enhance network performance.

### 3.3 Loss Function

**Distance-Aware Constraint.** Inspired by FOCAL [60], we design a novel constraint with distance-aware attention weights  $\alpha$  to ensure that the network pays more attention to the optimization of underfitting query points in space and optimizes the SDFs simultaneously. This allows query points at different distances from the surface to be optimized properly, and assigns higher attention weights for outlier and underlying points:

$$\begin{cases} \alpha_1, \alpha_2 = \text{softmax}(D_1, D_2), \\ \mathcal{L}_{\text{recon}} = \alpha_1 D_1 + \alpha_2^\gamma D_2 + D_3, \end{cases} \quad (6)$$

where  $\alpha_1$  and  $\alpha_2$  are calculated from  $D_1, D_2$  by the softmax activation,  $\gamma$  is a scaling coefficient we set to 2 by default. Here, we only consider 3 steps, which is a trade-off between performance and efficiency.

**Consistent Gradients.** We additionally introduce consistency constraints in the gradient direction. This loss encourages neighboring level sets to keep parallel, which reduces the artifacts off the surface and smooths the surface. We add a cosine gradient consistency loss function to encourage the gradient direction at the query points to keep consistent with the gradient direction at its target point on the surface, which aims to improve the continuity of the gradient during the multi-step pulling. We use  $Q_1, Q_2$  and  $Q_3$  to represent the query points that have been continuously optimized by the multiple steps. We take the one with the lowest similarity score to measure the overall similarity.

$$\begin{cases} L_\nabla(Q_i) = \cos(\nabla f(Q_i, y_i), \nabla f(Q_0, y_0)), \\ \mathcal{L}_{\text{grad}} = 1 - \min\{L_\nabla(Q_1), L_\nabla(Q_2), L_\nabla(Q_3)\}, \end{cases} \quad (7)$$

where  $L_\nabla(Q_i)$  represents the loss of cosine similarity between query points  $Q$  and target surface points  $q$ .

**Surface Constraint.** We introduce the surface constraint for the implicit function  $f$ , aiming to assist the network in approaching the zero level-set on the surface at final step. Hence, we constrain the  $f(Q_I, y_I)$  approaches zero at the final step:

$$\mathcal{L}_{\text{surf}} = \|f(Q_I, y_I)\|. \quad (8)$$

**Joint Loss Function.** Overall, we learn the SDFs by minimizing the following loss function  $\mathcal{L}$ .

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{grad}} + \delta \mathcal{L}_{\text{surf}}, \quad (9)$$

where  $\beta$  and  $\delta$  are balance weights. In the subsequent ablation experiments, we validated the effectiveness of different loss functions.

## 4 Experiments

In this section, we evaluate the performance of MultiPull in surface reconstruction by conducting numerical and visual comparisons with state-of-the-art methods on both synthetic and real-scan datasets. Specifically, in Sec. 4.1, we experiment on synthetic shape datasets with diverse topological structures. Furthermore, in Sec. 4.2, we report our results across various scales on real large-scale scene datasets. Meanwhile, we consider FAMOUS as the verification dataset in the ablation studies to compare the effectiveness of each module in MultiPull in Sec. 4.3.

### 4.1 Surface Reconstruction for Shapes

**Datasets and Metrics.** For the single shape surface reconstruction task, we perform evaluation on multiple datasets including ShapeNet [29], FAMOUS [10], Surface Reconstruction Benchmark (SRB) [45] Thingi10K [61] and D-FAUST [62]. We conduct validation experiments on 8 subcategories within the ShapeNet dataset, while the remaining datasets are experimented on the complete dataset. For metric comparison, we leverage L1 and L2 Chamfer Distance  $CD_{L1}$  and  $CD_{L2}$ , Normal Consistency (NC), and F-Score as evaluation metrics.

**ShapeNet.** We evaluate our approach on the ShapeNet[29] according to the experimental settings of GP [23]. We compared our methods with methods including ATLAS [63], DSDF [51], NP [21], PCP [64], GP [23], as shown in Tab. 1. We report  $CD_{L2}$ , NC and F-Score metrics for ShapeNet, where we randomly sample 10,000 points on the reconstructed object surface for evaluation. MultiPull outperforms the state-of-the-art methods. Compare to previous gradient-based methods in Fig. 4, our method performs better by revealing more local details of these complex structures. We provide detailed results in Appendix C.

Table 1: Reconstruction accuracy on ShapeNet in terms of  $CD_{L2}$ , NC and F-Score with thresholds of 0.002 and 0.004.

Methods	$CD_{L2} \times 100$	NC	$F\text{-Score}^{0.002}$	$F\text{-Score}^{0.004}$
ATLAS	1.368	0.695	0.062	0.158
DSDF	0.766	0.884	0.212	0.717
NP	0.038	0.939	0.961	0.976
PCP	0.0136	0.9590	0.9871	0.9899
GP	0.0086	0.9723	0.9896	0.9923
Ours	<b>0.0075</b>	<b>0.9737</b>	<b>0.9906</b>	<b>0.9932</b>

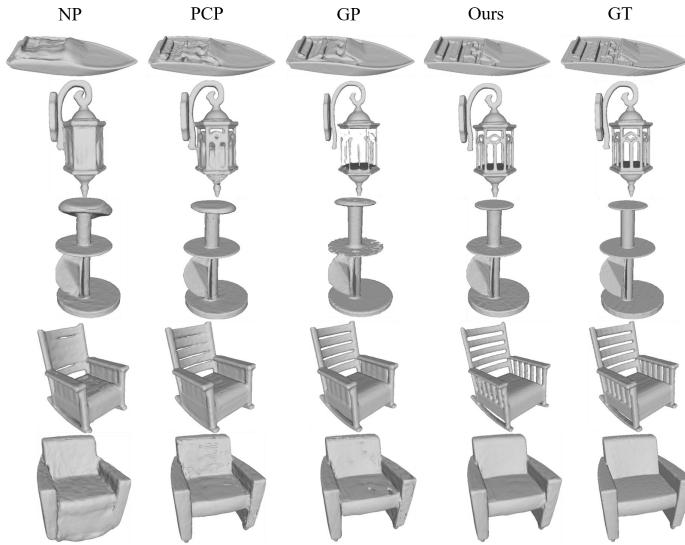


Figure 4: Visual comparison of reconstructions on ShapeNet.

**FAMOUS.** We evaluate the performance of our method on the FAMOUS dataset according to the experimental settings of PCP [64] and NP [21]. Our method demonstrates superiority over recent

approaches, including GP [23], PCP [64], GenSDF [65], FGC [66], NP [21], and IGR [57]. As shown in Tab. 2, we compared the recent methods using  $CD_{L2}$  and NC metrics, and our method exhibits outstanding performance. To demonstrate the effectiveness of our method in reconstruction accuracy, we visualize the error-map for comparison in Fig. 5. Compare to the the state-of-art methods, our method has better overall reconstruction accuracy (bluer).

Table 2: Reconstruction accuracy on FA-MOUS in terms of  $CD_{L2}$  and NC.

Methods	$CD_{L2} \times 100$	NC
IGR	1.65	0.911
GenSDF	0.668	0.909
NP	0.220	0.914
FGC	0.055	0.933
PCP	0.044	0.933
GP	0.040	0.945
Ours	<b>0.035</b>	<b>0.953</b>

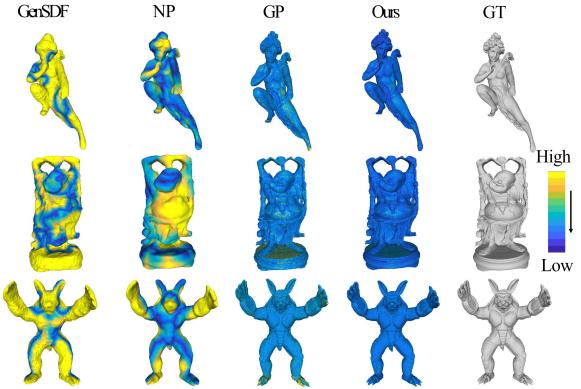


Figure 5: Visual comparison of error maps on FA-MOUS.

Table 3: Reconstruction accuracy on SRB in terms of  $CD_{L1}$  and F-Score with a threshold of 0.01.

Methods	$CD_{L2} \times 100$	NC
P2M	0.116	64.8
SAP	0.076	83.0
NP	0.106	79.7
BACON	0.089	82.7
CAP	0.073	84.5
GP	0.070	85.1
Ours	<b>0.068</b>	<b>85.7</b>

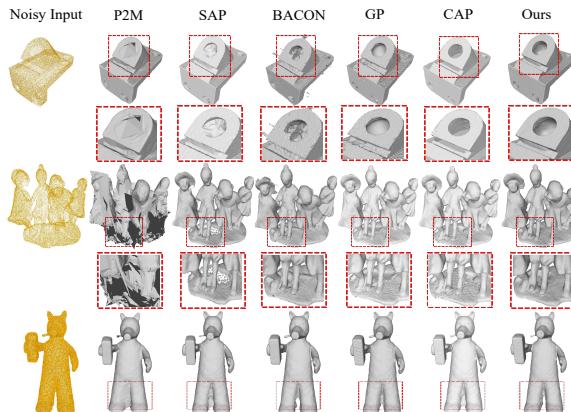


Figure 6: Visual comparison on SRB.

**SRB.** We validate our method on the real scanned dataset SRB, following the experimental settings of VisCo [67] and GP [23]. In Tab. 3, we compared our approach with recent methods including P2M [68], SAP [56], NP [21], BACON [15], CAP [55], GP [23]. We use  $CD_{L1}$  and F-Score to evaluate performance , and we surpass all others in terms of these metrics. As depicted in Fig. 6, our method excels in reconstructing more complete and smoother surfaces.

**D-FAUST.** We evaluate our method on the D-FAUST dataset with SAP [56] settings. As indicated in Tab. 4, we compared our approach with recent methods including IGR [57], SAP [56], GP [23]. Our method excels in  $CD_{L1}$ , F-Score and NC. As illustrated in Fig. 7, compared to other methods, our approach demonstrates superior accuracy in recovering human body shapes.

**Thingi10K.** We assess the performance of our approach on the Thingi10K dataset, following the experimental setup of SAP [56]. We compared our approach with recent methods including IGR [57], SAP [56], BACON [15], GP [23]. As indicated in Tab. 5, our method surpasses existing methods across in  $CD_{L1}$ , F-Score and NC metrics. As illustrated in Fig. 8, our method can reconstruct surfaces with more accurate details.

Table 4: Reconstruction accuracy under D-FAUST in terms of  $CD_{L1}$  and F-Score with a threshold of 0.01.

Methods	$CD_{L1}$	$F\text{-Score}^{0.01}$	NC
IGR	0.235	0.805	0.911
SAP	0.043	0.966	0.959
GP	0.015	0.975	0.978
Ours	<b>0.009</b>	<b>0.986</b>	<b>0.988</b>

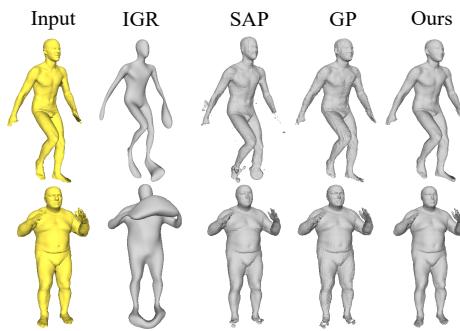


Figure 7: Visual comparison on D-FAUST.

Table 5: Reconstruction accuracy under Thingi10K in terms of  $CD_{L1}$  and F-Score with a threshold of 0.01.

Methods	$CD_{L1}$	$F\text{-Score}^{0.01}$	NC
IGR	0.440	0.505	0.692
SAP	0.054	0.940	0.947
BACON	0.053	0.946	0.961
GP	0.051	0.948	0.965
Ours	<b>0.048</b>	<b>0.953</b>	<b>0.968</b>

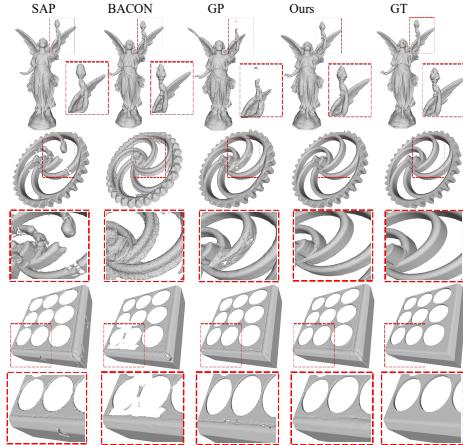


Figure 8: Visual comparison on Thingi10K.

## 4.2 Surface Reconstruction for Real-Scan Scenes

**Datasets and Metrics.** For the scene reconstruction task, we validate our method on the 3DScape [69] and KITTI [70] datasets to assess the performance on large-scale datasets. We keep the same evaluation metrics as those used for shape reconstruction in Sec. 4.1.

**3DScape.** In accordance with the experimental settings of PCP [64], we compared our approach with recent methods including ConvOcc [39], NP [21], PCP [64] and GP [23]. We report the evaluation results of  $CD_{L1}$ ,  $CD_{L2}$  and NC, and compared our method with the latest approaches listed in Tab. 6. As illustrated in Fig. 9, our method outperforms prior-based methods and overfitting based methods.

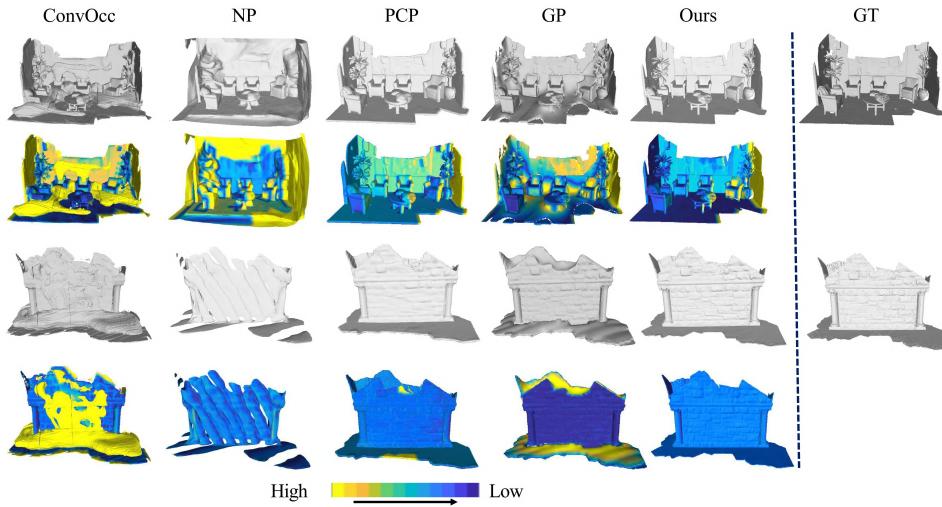


Figure 9: Visual comparison of CD error maps on 3DScape.

Table 6: Reconstruction accuracy on 3DScene in terms of  $CD_{L1}$ ,  $CD_{L2}$  and NC.

Methods	$CD_{L2} \times 100$	$CD_{L1}$	NC
ConvOcc	13.32	0.049	0.752
NP	8.350	0.0194	0.713
PCP	0.11	0.007	0.886
GP	0.10	<b>0.006</b>	0.903
Ours	<b>0.094</b>	<b>0.006</b>	<b>0.918</b>

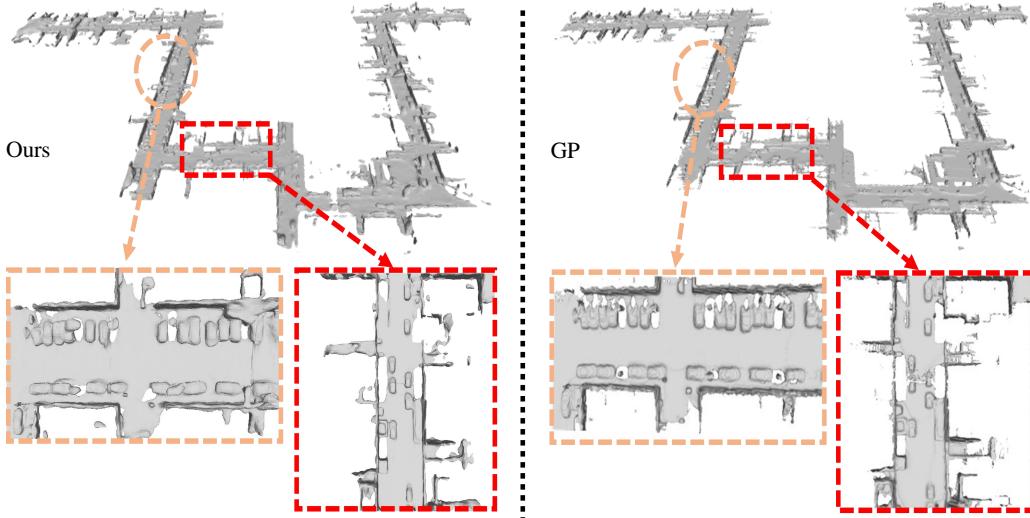


Figure 10: Visual comparison on KITTI.

**KITTI.** We validate our method on the large-scale scanned point cloud dataset KITTI [70], which contains 13.8 million points. As shown in Fig. 10, our approach is capable of reconstructing more complete and accurate surfaces compared to the GP method [23]. GP struggles to reconstruct continuous surfaces such as walls and streets, whereas our method achieves a more detailed reconstruction of objects at various scales in real scanned scenes. It demonstrates that our method is robust when handling point cloud with various scales.

### 4.3 Ablation Experiments

**Effect of Frequency Layers.** We denote the  $j$ -th layer of the frequency network as  $L_j$ , a specific combination of frequency feature layers can be formulated as  $\{L_i, L_j, L_k\}$ , where  $\{i, j, k\} \in [1, N_L - 1]$ . We evaluate the effectiveness of the frequency transformer layers in Tab. 7 with  $CD_{L2}$  and NC, replacing the frequency network with linear layers results in a decrease in the performance of the  $CD_{L2}$  and NC metrics. The performance of using only one layer( $L_4$ ) surpasses linear layers. With the increase of the frequency layers,  $\{L_4, L_6, L_8\}$  produces best results.

Table 7: Effect of frequency features.

Layer	$CD_{L2} \times 100$	NC
Linear	0.042	0.920
$L_4$	0.040	0.926
$L_4, L_6$	0.037	0.933
$L_4, L_6, L_8$	<b>0.036</b>	<b>0.948</b>

**Effect of MSP Module.** We report comparisons with different features in Tab. 8. The 'Layer' column denotes the combination of frequency features obtained by the FFT module. For instance,  $\{L_4, L_6, L_8\}$  represent the frequency features from the 4<sup>th</sup>, 6<sup>th</sup>, and 8<sup>th</sup> layers guiding the pulling of the query point in the MSP network, respectively. We find that the accuracy of the network increases with the number of steps. After considering both performance metrics and time efficiency, we have set Step=3 by default.

Table 8: Effect of MSP Module.

Step	$CD_{L2} \times 100$	NC
1	0.040	0.926
2	0.037	0.933
3	0.036	0.948
4	0.036	0.942
5	<b>0.0357</b>	<b>0.955</b>

**Effect of Loss Functions.** We compared  $CD_{L2}$  metric under different loss strategies in Tab. 9. As shown in the table, Weighting query points at different scales effectively enhances reconstruction accuracy and the reconstruction loss allows the network to obtain a complete shape with local details. Furthermore, The gradient loss improves the surface continuity of the object. And the surface supervision loss facilitates the learning of more precise zero-level sets, which also improves the accuracy.

Table 9: Effect of loss functions.

Loss	$CD_{L2} \times 100$	NC
$\mathcal{L}_{\text{pull}}$	0.0443	0.937
$\mathcal{L}_{\text{recon}}$	0.0383	0.946
$\mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{sim}}$	0.0367	0.948
$\mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{sim}} + \mathcal{L}_{\text{sdf}}$	<b>0.0352</b>	<b>0.954</b>

**Effect of Different Levels of Noise.** We evaluate the reconstruction performance of our method on the Famous dataset under two levels of noise: Mid-Level and Max-Level noise. As shown in Tab. 10, our method outperforms the majority of approaches even in the presence of noisy inputs.

Table 10: Effect of different levels of noise.

Noise level	NP	PCP	GP	Ours
Mid-Noise	0.280	0.071	0.044	<b>0.037</b>
Max-Noise	0.310	0.298	0.060	<b>0.058</b>

## 5 Conclusion

We propose a novel method to learn detailed SDFs by pulling queries onto the surface at multi-step. We leverage the multi-level features to predict signed distances, which recovers high frequency details. Through optimization, our method is able to gradually restore the coarse-to-fine structure of reconstructed objects, thereby revealing more geometry details. Visual and numerical comparisons show that our approach demonstrates competitive performance over the state-of-the-art methods.

## 6 Acknowledgment

This work was supported by National Key R&D Program of China (2022YFC3800600), and the National Natural Science Foundation of China (62272263, 62072268), and in part by Tsinghua-Kuaishou Institute of Future Media Data.

## References

- [1] Matthew Tancik, Vincent Casser, Xincheng Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-NeRF: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022.
- [2] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021.
- [3] Zhizhong Han, Chao Chen, Yu-Shen Liu, and Matthias Zwicker. Drwr: A differentiable renderer without rendering for unsupervised 3D structure learning from silhouette images. In *International Conference on Machine Learning*, 2020.
- [4] Chao Chen, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Unsupervised learning of fine structure generation for 3D point clouds by 2D projections matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12466–12477, 2021.
- [5] Junsheng Zhou, Baorui Ma, Shujian Li, Yu-Shen Liu, and Zhizhong Han. Learning a more continuous zero level set in unsigned distance fields through level set projection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.
- [6] Zixiong Wang, Yunxiao Zhang, Rui Xu, Fan Zhang, Peng-Shuai Wang, Shuangmin Chen, Shiqing Xin, Wenping Wang, and Changhe Tu. Neural-singular-hessian: Implicit neural representation of unoriented point clouds by enforcing singular hessian. *ACM Transactions on Graphics (TOG)*, 42(6):1–14, 2023.
- [7] Fei Hou, Chiyu Wang, Wencheng Wang, Hong Qin, Chen Qian, and Ying He. Iterative poisson surface reconstruction (ipsr) for unoriented points. *arXiv preprint arXiv:2209.09510*, 2022.
- [8] Pengchong Hu and Zhizhong Han. Learning neural implicit through volume rendering with attentive depth fusion priors. In *Advances in Neural Information Processing Systems*, 2023.
- [9] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 21(4):163–169, 1987.
- [10] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J Mitra, and Michael Wimmer. Points2Surf learning implicit surfaces from point clouds. In *European Conference on Computer Vision*, pages 108–124. Springer, 2020.
- [11] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, Thomas Funkhouser, et al. Local implicit grid representations for 3D scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6001–6010, 2020.
- [12] Julien NP Martel, David B Lindell, Connor Z Lin, Eric R Chan, Marco Monteiro, and Gordon Wetzstein. ACORN: adaptive coordinate networks for neural scene representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–13, 2021.
- [13] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J Guibas. Curriculum deepsdf. In *European Conference on Computer Vision*, pages 51–67. Springer, 2020.
- [14] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J Zico Kolter. Multiplicative filter networks. In *International Conference on Learning Representations*, 2020.
- [15] David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. BACON: Band-limited coordinate networks for multiscale scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16252–16262, 2022.
- [16] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021.
- [17] Yishun Dou, Zhong Zheng, Qiaoqiao Jin, and Bingbing Ni. Multiplicative fourier level of detail. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1808–1817, 2023.
- [18] Matan Atzmon and Yaron Lipman. SAL: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2565–2574, 2020.

- [19] Matan Atzmon and Yaron Lipman. SALD: Sign agnostic learning with derivatives. In *International Conference on Learning Representations*, 2020.
- [20] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3D shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4857–4866, 2020.
- [21] Baorui Ma, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Neural-Pull: Learning signed distance function from point clouds by learning to pull space onto surface. In *International Conference on Machine Learning*, pages 7246–7257. PMLR, 2021.
- [22] Baorui Ma, Junsheng Zhou, Yu-Shen Liu, and Zhizhong Han. Towards better gradient consistency for neural signed distance functions via level set alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17724–17734, 2023.
- [23] Chao Chen, Yu-Shen Liu, and Zhizhong Han. GridPull: Towards scalability in learning implicit representations from 3D point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18322–18334, 2023.
- [24] Danhang Tang, Mingsong Dou, Peter Lincoln, Philip Davidson, Kaiwen Guo, Jonathan Taylor, Sean Fanello, Cem Keskin, Adarsh Kowdle, Sofien Bouaziz, et al. Real-time compression and streaming of 4d performances. *ACM Transactions on Graphics (TOG)*, 37(6):1–11, 2018.
- [25] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.
- [26] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(3):1–13, 2013.
- [27] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018.
- [28] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22(3):463–470, 2003.
- [29] ShapeNet. <https://www.shapenet.org/>.
- [30] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9601–9611, 2019.
- [31] Konstantinos Rematas, Ricardo Martin-Brualla, and Vittorio Ferrari. Sharf: Shape-conditioned radiance fields from a single view. In *International Conference on Machine Learning*, 2021.
- [32] Baorui Ma, Yu-Shen Liu, and Zhizhong Han. Learning signed distance functions from noisy 3d point clouds via noise to noise mapping. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [33] Shujuan Li, Junsheng Zhou, Baorui Ma, Yu-Shen Liu, and Zhizhong Han. Neaf: Learning neural angle fields for point normal estimation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 1396–1404, 2023.
- [34] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.
- [35] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [36] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2023.
- [37] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions on Graphics (TOG)*, 42(4):1–16, 2023.

- [38] Siyu Ren, Junhui Hou, Xiaodong Chen, Ying He, and Wenping Wang. Geoudf: Surface reconstruction from 3d point clouds via geometry-guided distance representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14214–14224, 2023.
- [39] 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.
- [40] Shichen Liu, Shunsuke Saito, Weikai Chen, and Hao Li. Learning to infer implicit surfaces without 3D supervision. In *Advances in Neural Information Processing Systems*, 2019.
- [41] Lior Yariv, Jitao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Advances in Neural Information Processing Systems*, 2021.
- [42] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020.
- [43] Liang Han, Junsheng Zhou, Yu-Shen Liu, and Zhizhong Han. Binocular-guided 3D gaussian splatting with view consistency for sparse view synthesis. In *Advances in Neural Information Processing Systems*, 2024.
- [44] Wenyuan Zhang, Yu-Shen Liu, and Zhizhong Han. Neural signed distance function inference through splatting 3d gaussians pulled on zero-level set. In *Advances in Neural Information Processing Systems*, 2024.
- [45] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019.
- [46] Minghua Liu, Xiaoshuai Zhang, and Hao Su. Meshing point clouds with predicted intrinsic-extrinsic ratio guidance. In *European Conference on Computer vision*, 2020.
- [47] Zhenxing Mi, Yiming Luo, and Wenbing Tao. SSRNet: Scalable 3D surface reconstruction network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 970–979, 2020.
- [48] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Improved direct voxel grid optimization for radiance fields reconstruction. *arXiv preprint arXiv:2206.05085*, 2022.
- [49] N. Sedaghat, M. Zolfaghari, E. Amiri, and T. Brox. Orientation-boosted voxel nets for 3D object recognition. In *British Machine Vision Conference*, 2017.
- [50] Hai Li, Xingrui Yang, Hongjia Zhai, Yuqian Liu, Hujun Bao, and Guofeng Zhang. Vox-Surf: Voxel-based implicit surface representation. *CoRR*, abs/2208.10925, 2022.
- [51] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.
- [52] Gal Metzer, Rana Hanocka, Denis Zorin, Raja Giryes, Daniele Panozzo, and Daniel Cohen-Or. Orienting point clouds with dipole propagation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021.
- [53] Rui Xu, Zhiyang Dou, Ningna Wang, Shiqing Xin, Shuangmin Chen, Mingyan Jiang, Xiaohu Guo, Wenping Wang, and Changhe Tu. Globally consistent normal orientation for point clouds by regularizing the winding-number field. *ACM Transactions on Graphics (TOG)*, 42(4):1–15, 2023.
- [54] Qing Li, Huifang Feng, Kanle Shi, Yue Gao, Yi Fang, Yu-Shen Liu, and Zhizhong Han. Shs-net: Learning signed hyper surfaces for oriented normal estimation of point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13591–13600, 2023.
- [55] Junsheng Zhou, Baorui Ma, Liu Yu-Shen, Fang Yi, and Han Zhizhong. Learning consistency-aware unsigned distance functions progressively from raw point clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [56] Songyou Peng, Chiyu “Max” Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. In *Advances in Neural Information Processing Systems*, 2021.
- [57] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, pages 3789–3799. PMLR, 2020.

- [58] Wenbin Zhao, Jiabao Lei, Yuxin Wen, Jianguo Zhang, and Kui Jia. Sign-agnostic implicit learning of surface self-similarities for shape modeling and reconstruction from raw point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10256–10265, 2021.
- [59] Yu-Tao Liu, Li Wang, Jie Yang, Weikai Chen, Xiaoxu Meng, Bo Yang, and Lin Gao. Neudf: Leaning neural unsigned distance fields with volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 237–247, 2023.
- [60] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.
- [61] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016.
- [62] 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.
- [63] Qian Yu, Chengzhan Yang, and Hui Wei. Part-wise atlasnet for 3d point cloud reconstruction from a single image. *Knowledge-Based Systems*, 242:108395, 2022.
- [64] Baorui Ma, Yu-Shen Liu, Matthias Zwicker, and Zhizhong Han. Surface reconstruction from point clouds by learning predictive context priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [65] Gene Chou, Ilya Chugunov, and Felix Heide. GenSDF: Two-stage learning of generalizable signed distance functions. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2022.
- [66] Mulin Yu and Florent Lafarge. Finding good configurations of planar primitives in unorganized point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6367–6376, June 2022.
- [67] Albert Pumarola, Artsiom Sanakoyeu, Lior Yariv, Ali Thabet, and Yaron Lipman. Visco grids: Surface reconstruction with viscosity and coarea grids. In *Advances in Neural Information Processing Systems*, 2022.
- [68] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2Mesh: a self-prior for deformable meshes. *ACM Transactions on Graphics (TOG)*, 39(4):126–1, 2020.
- [69] Qian-Yi Zhou and Vladlen Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (TOG)*, 32(4):1–8, 2013.
- [70] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.
- [71] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022.

## A Implementation Details.

Our network consists of two main parts: frequency feature transformation and multi-step pulling modules in Fig 2 (a) and (b), respectively. For frequency feature transformation, we transform the raw point cloud into frequency features  $M$ , where  $M$  is initialized to 256. Same for the multi-step pulling module, we train a linear sequence neural network (**LSNN**) with shared parameters and we fix intermediate layer output dimension to 512. In the construction of query points, we establish the corresponding pairs between query points and their nearest points on surfaces. Specifically, we follow NP [21] to construct 40 queries for each point of the point cloud, the construction of these query points follows a Gaussian distribution. During the reconstruction process, we use the Marching Cubes algorithm [27] to extract the mesh surface.

During the training process, we do optimization in 40,000 iterations, with an average time of 24 minutes for single-object reconstruction. We utilize a single NVIDIA RTX-3090 GPU for both training and testing.

## B Additional Experiments

**Effect of Frequency Features.** To further validate the superiority of frequency features, we exclude multi-step pulling, and only use single-layer frequency feature for performance verification against linear layers. We note the frequency feature in the  $i$ -th layer as  $L_i$ . We compared the  $CD_{L2}$  and NC of specific layers( $L_2, L_4, L_6, L_8$ ) with the linear layers. As shown in Tab. 11 the performance of frequency features at different layers is superior to the linear layers, and with an increase in the number of layers, higher-level frequency conditions enhance the network’s performance.

Table 11: Effect of frequency features.

Layer	$CD_{L2} \times 100$	NC
Linear	0.042	0.920
$L_4$	0.040	0.926
$L_6$	0.038	0.931
$L_8$	<b>0.037</b>	<b>0.935</b>

**Effect of Initialization Strategies.** We compared our initialization strategy with random initialization and MFN-based method (BACON [15]) as example. We compared the metrics of these initialization methods in Tab. 12, which shows that combining random or BACON initialization with our approach does not yield satisfactory results. To further demonstrate the advantages of our initialization method, we visually compared SDF with random initialization and BACON initialization strategies. As shown in the Fig. 11, our method significantly outperforms other initialization methods in terms of convergence speed. In addition, our reconstruction results also indicate that a reasonable initialization method can enable the network to learn more accurate signed distance field. We compared the results with the same iterations and the final results under the default settings for different methods (Final) in Fig. 12, these failed reconstructions based on MFN demonstrate the instability of parameter initialization.

Table 12: Effect of initialization strategies.

Initialization	$CD_{L2} \times 100$	NC
Random	0.042	0.938
BACON	0.038	0.946
Ours	<b>0.035</b>	<b>0.950</b>

**Effect of Parameters on Networks.** We compared the parameter quantities of the methods listed in Tab. 13 below. It shows that the parameter number of PCP[64] is the largest among all the three methods, while NP[21] has the least parameters. To further investigate the performance of networks with the similar amount of parameters, we increase the parameters of NP and MultiPull to match PCP. The comparison in the Tab. 14 indicates that both NP and MultiPull show the improved performance. This demonstrates that more parameters are beneficial to improve the performance, but

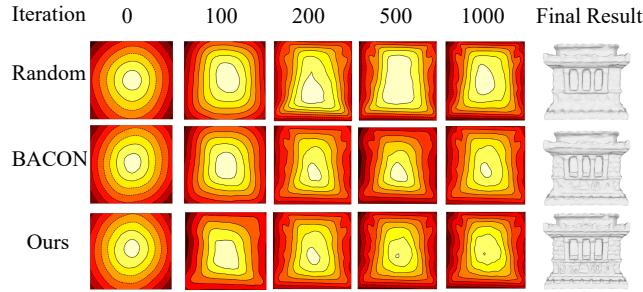


Figure 11: Comparison of signed distances in optimization

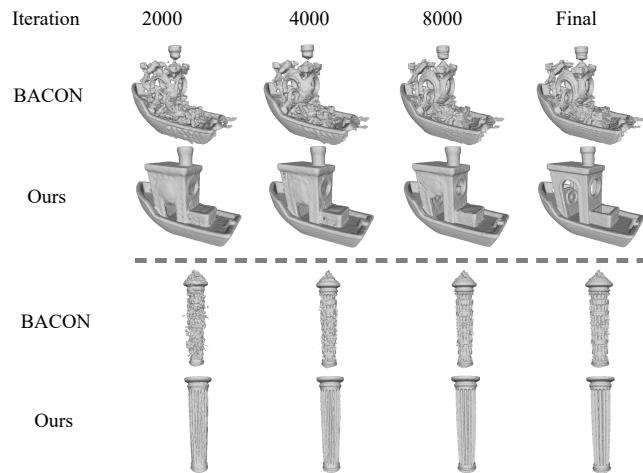


Figure 12: Comparison of the initialization strategies in optimization.

our performance comes from our novel methods rather than more parameters. Meanwhile, we further evaluate inference time with NP and PCP. As shown in Tb. 15 the local based method (PCP) has the longer inference time than the global based method, with NP having the fastest inference time.

Table 13: Comparison at different parameter levels.

Method	Parameters	$CD_{L2} \times 100$	NC
NP	1843708	0.220	0.914
PCP	<b>7894022</b>	0.044	0.933
Ours	2648094	<b>0.035</b>	0.953

Table 14: Comparison at a uniform parameter levels.

Method	Parameters	$CD_{L2} \times 100$	NC
NP	7907441	0.1966	0.917
PCP	<b>7894022</b>	0.044	0.933
Ours	7856620	<b>0.0317</b>	0.957

Table 15: Comparison of Inference Time.

	PCP	NP	Ours
Inference Time(min)	0.17	<b>0.138</b>	0.141

**Design of  $\mathcal{L}_{\text{grad}}$ .** We further discuss the design of  $\mathcal{L}_{\text{grad}}$  and effectiveness on performance. We use minimum (min) as the baseline and compare it with using the average (avg). As shown in Tab. 16, using  $\mathcal{L}_{\text{grad}}(\text{avg})$  to calculate the similarity of query points at different time steps results in a slight increase in CD error. In contrast,  $\mathcal{L}_{\text{grad}}(\text{min})$  achieves a similar level of similarity but better performance in CD metrics. Therefore, we calculate minimum of the gradient similarities as a constraint to prevent significant deviations in the moving direction during training, making the network more sensitive to changes in gradient direction.

Table 16: Comparison at a uniform parameter levels.

	w/o $\mathcal{L}_{\text{grad}}$	$\mathcal{L}_{\text{grad}}(\text{avg})$	$\mathcal{L}_{\text{grad}}(\text{min})$
$\text{CD}_{L2} \times 100$	0.0388	0.0383	<b>0.0352</b>
NC	0.945	0.950	<b>0.954</b>

**Feature Comparison.** We combine MSP with the linear layers and the traditional feature learning encoder PointMLP to explore the effectiveness of MSP. We present the results of combining different feature learning networks with MSP methods in Tab. 17. We denote the single moving operation in MSP as Pull and use multiple feature learning networks as baselines. Due to the lack of an FFT module, the same features are used for multiple offsets in linear+MSP and PointMLP + MSP. As shown in Tab. 17, the combination of different feature extraction networks and MSP achieved better results in terms of both CD and NC metrics. We further demonstrate the effectiveness of MSP in Fig. 13 and Fig. 14 in the PDF. PointMLP / linear+MSP can generate finer local details compared to PointMLP / linear + Pull.

Table 17: Comparison of Reconstruction Accuracy in  $\text{CD}_{L2} \times 100$

	Iteration	10K	20K	40K
Linear+Pull	0.078	0.061	<b>0.055</b>	
PointMLP+Pull	0.064	0.045	<b>0.037</b>	
Linear+MSP	0.073	0.058	<b>0.041</b>	
PointMLP+MSP	<b>0.053</b>	<b>0.042</b>	<b>0.037</b>	

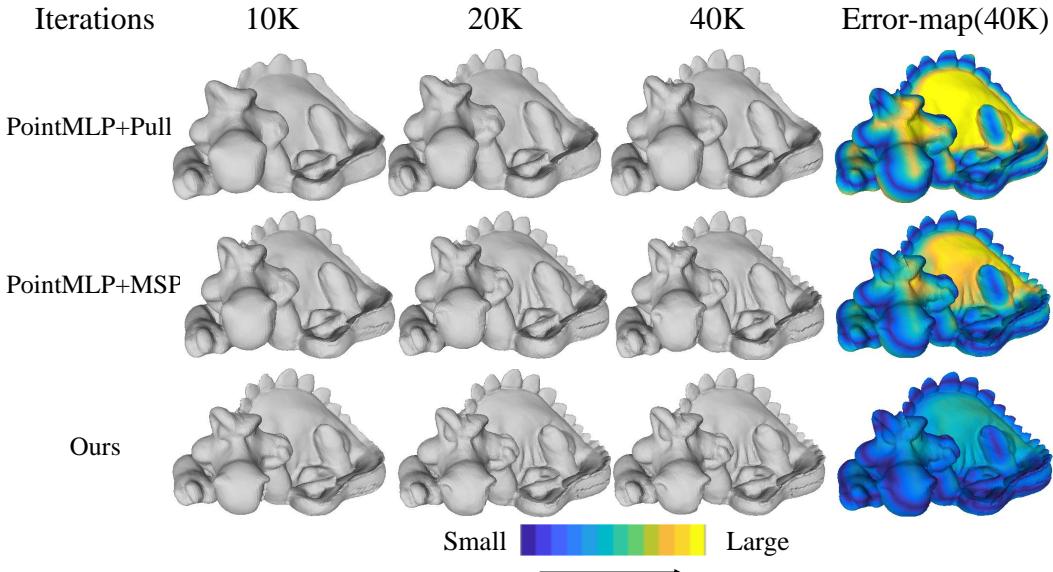


Figure 13: Comparison of different feature encoders and MSP module on FAMOUS dataset.

**Default setting of Step.** We set Step to 3 by default for two reasons: (1) MSP can bring partial accuracy improvement when learning more steps, but more network parameters are also needed. We

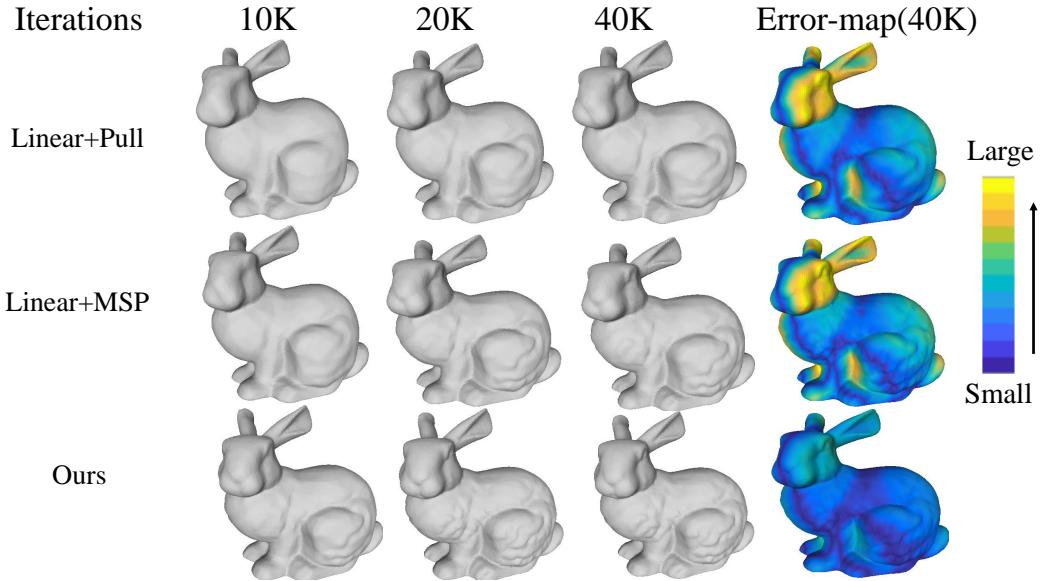


Figure 14: Comparison of different feature encoders and MSP module on FAMOUS dataset.

present comparisons under different steps in Tab. 8. (2) Although deeper frequency features can represent more comprehensive information, combining frequency features from different layers can achieve better results. We denote the combination of the frequency. We conduct the experiment according to the default settings, and the results are shown in Tab. 18.

Table 18: Comparison of  $CD_{L2} \times 100$ , NC under different feature combination.

	$L_4, L_4, L_4$	$L_6, L_6, L_6$	$L_8, L_8, L_8$	$L_4, L_6, L_8$
$CD_{L2} \times 100$	0.0417	0.0397	0.0360	<b>0.0343</b>
NC	0.922	0.927	0.938	<b>0.948</b>

## C Additional Results

**Comparison Details for ShapeNet.** The complete comparison under all the five scenes of the ShapeNet dataset. The results are shown in Tab. 19 to Tab. 21. We use Chamfer Distance ( $CD_{L1}$ ,  $CD_{L2}$ ) and NC as evaluation metrics.

Table 19: Reconstruction accuracy on ShapeNet in terms of  $CD_{L2}$

Class	ATLAS	DSDF	NP	PCP	GP	Ours
Display	1.094	0.317	0.039	0.0087	0.0082	<b>0.0074</b>
Lamp	1.988	0.955	0.080	0.0380	0.0347	<b>0.0301</b>
Airplane	1.011	1.043	0.008	0.0065	0.0007	<b>0.0006</b>
Cabinet	1.611	0.921	0.026	0.0153	0.0112	<b>0.0105</b>
Vessel	0.997	1.254	0.022	0.0079	0.0033	<b>0.0028</b>
Table	1.311	0.660	0.060	0.0131	0.0052	<b>0.0047</b>
Chair	1.575	0.483	0.054	0.0110	0.0043	<b>0.0036</b>
Sofa	1.307	0.496	0.012	0.0086	0.0015	<b>0.0009</b>
Mean	1.368	0.766	0.038	0.0136	0.0086	<b>0.0075</b>

Table 20: Reconstruction accuracy on ShapeNet in terms of NC.

Class	ATLAS	DSDF	NP	PCP	GP	Ours
Display	0.828	0.932	0.964	0.9775	0.9847	<b>0.9855</b>
Lamp	0.593	0.864	0.930	0.9450	0.9693	<b>0.9710</b>
Airplane	0.737	0.872	0.947	0.9490	0.9614	<b>0.9633</b>
Cabinet	0.682	0.872	0.930	0.9600	0.9689	<b>0.9693</b>
Vessel	0.671	0.841	0.941	0.9546	0.9667	<b>0.9671</b>
Table	0.783	0.901	0.908	0.9595	0.9755	<b>0.9758</b>
Chair	0.638	0.886	0.937	0.9580	0.9733	<b>0.9757</b>
Sofa	0.633	0.906	0.951	0.9680	0.9792	<b>0.9822</b>
Mean	0.695	0.884	0.939	0.9590	0.9723	<b>0.9737</b>

Table 21: Reconstruction accuracy on ShapeNet in terms of F- Score with thresholds of 0.002 and 0.004.

	F-Score <sup>0.002</sup>						F-Score <sup>0.004</sup>					
	ATLAS	DSDF	NP	PCP	GP	Ours	ATLAS	DSDF	NP	PCP	GP	Ours
Display	0.071	0.632	0.989	0.9939	0.9963	<b>0.9966</b>	0.179	0.787	0.991	0.9958	0.9963	<b>0.9980</b>
Lamp	0.029	0.268	0.891	0.9382	0.9455	<b>0.9473</b>	0.077	0.478	0.924	0.9402	0.9538	<b>0.9561</b>
Airplane	0.070	0.350	0.996	0.9942	0.9976	<b>0.9989</b>	0.179	0.566	0.997	0.9972	0.9989	<b>0.9991</b>
Cabinet	0.077	0.573	0.980	0.9888	0.9901	<b>0.9913</b>	0.195	0.694	0.989	0.9939	0.9946	<b>0.9969</b>
Vessel	0.058	0.323	0.985	0.9935	0.9956	<b>0.9962</b>	0.153	0.509	0.990	0.9958	0.9972	<b>0.9979</b>
Table	0.080	0.577	0.922	0.9969	0.9977	<b>0.9982</b>	0.195	0.743	0.973	0.9958	<b>0.9990</b>	0.9988
Chair	0.050	0.447	0.954	0.9970	0.9979	<b>0.9980</b>	0.134	0.665	0.969	0.9991	0.9990	<b>0.9994</b>
Sofa	0.058	0.577	0.968	0.9943	0.9974	<b>0.9981</b>	0.153	0.734	0.974	0.9987	<b>0.9992</b>	<b>0.9992</b>
Mean	0.062	0.212	0.961	0.9871	0.9896	<b>0.9906</b>	0.158	0.717	0.976	0.9899	0.9923	<b>0.9932</b>

**Comparison Details for 3DScene.** We also provide detailed metrics for single scenes in 3DScene dataset. We evaluate it by  $CD_{L1}$ ,  $CD_{L2}$  and NC. As shown in the Tab. 22, our approach achieves the best performance across all scenes.

Table 22:  $CD_{L1}$ ,  $CD_{L2} \times 100$  and NC comparison under 3DScene.

		ConvOcc	NP	PCP	GP	Ours
Burghers	$CD_{L2} \times 100$	26.69	1.76	0.267	0.246	<b>0.228</b>
	$CD_{L1}$	0.077	0.010	<b>0.008</b>	<b>0.008</b>	<b>0.008</b>
	NC	0.865	0.883	0.914	0.926	<b>0.934</b>
Lounge	$CD_{L2} \times 100$	8.68	39.71	0.061	0.055	<b>0.048</b>
	$CD_{L1}$	0.042	0.059	0.006	<b>0.005</b>	<b>0.005</b>
	NC	0.857	0.857	0.928	0.922	<b>0.949</b>
Copyroom	$CD_{L2} \times 100$	10.99	0.051	0.076	0.069	<b>0.064</b>
	$CD_{L1}$	0.045	0.011	0.007	<b>0.006</b>	<b>0.006</b>
	NC	0.848	0.884	0.918	0.929	<b>0.923</b>
Stonewall	$CD_{L2} \times 100$	19.12	0.063	0.061	0.058	<b>0.043</b>
	$CD_{L1}$	0.066	0.007	0.0065	0.006	<b>0.005</b>
	NC	0.866	0.868	0.888	0.893	<b>0.936</b>
Totepole	$CD_{L2} \times 100$	1.16	0.19	0.10	0.093	<b>0.087</b>
	$CD_{L1}$	0.016	0.010	0.008	<b>0.007</b>	<b>0.007</b>
	NC	0.325	0.765	0.784	0.847	<b>0.851</b>

**Computational Complexity.** We report our computational complexity in Tab. 23, we present numerical comparisons with the latest overfitting-based methods, including NP and PCP, using different point counts, such as 20K and 40K. The benchmark rounds for both NP and PCP are set at 40K. NP does not require learning priors, resulting in the highest operational efficiency. PCP needs to learn priors, which requires additional time. To achieve more refined results, we dedicate extra time to learning the frequency features of point clouds and computing the sampling point strides. Consequently, our speed is slower compared to NP. However, it is noteworthy that our method outperforms PCP and operates faster even without using local priors.

Table 23: Comparison of computational complexity.

Time/GPU Memory	20K	40K
NP	<b>12min/1.5G</b>	<b>15min/2.3G</b>
PCP	14min/1.9G	18min/2.7G
Ours	13min/1.8G	16min/2.5G

## D Limitation

We propose a method that approximates the accurate signed distance field through multi-step optimization, achieving more precise results. However, there is still room for further optimization in terms of time and computational efficiency as shown in Tab. 8 and Tab. 14. In future work, we will continue to explore how to integrate multi-resolution (such as NGLOD [16] and Instant-NGP [71]) features effectively to balance computational efficiency and accuracy.

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: Our main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We analyze the limitations of our method in the Appendix. D

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results. method in the main paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the detailed information in reproducing our methods in Sec. 3-4 of the main paper and the appendix. We also provide a demonstration code of our method in the supplementary materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [\[Yes\]](#)

Justification: We provide our demonstration code as a part of our supplementary materials. We will release the source code, data and instructions upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: We provide the training and testing details in the experiment section (Sec.4) and the Appendix .A.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[No\]](#)

Justification: We report the average performance as the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer “Yes” if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification: The computer resources needed to reproduce the experiments are provided in the Appendix .A.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [\[Yes\]](#)

Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: We discuss the applications and potential impacts of our method in the introduction and contribution sections.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: **[No]**

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: **[Yes]**

Justification: We use the open-sourced datasets under their licenses.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

### 13. New Assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

### 14. Crowdsourcing and Research with Human Subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

### 15. Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.