Surface Reconstruction from Point Clouds by Learning Predictive Context Priors

Baorui Ma¹, Yu-Shen Liu¹; Matthias Zwicker², Zhizhong Han³
¹School of Software, BNRist, Tsinghua University, Beijing, China
²Department of Computer Science, University of Maryland, College Park, USA
³Department of Computer Science, Wayne State University, Detroit, USA
mbr18@mails.tsinghua.edu.cn, liuyushen@tsinghua.edu.cn, zwicker@cs.umd.edu, h312h@wayne.edu

Abstract

Surface reconstruction from point clouds is vital for 3D computer vision. State-of-the-art methods leverage large datasets to first learn local context priors that are represented as neural network-based signed distance functions (SDFs) with some parameters encoding the local contexts. To reconstruct a surface at a specific query location at inference time, these methods then match the local reconstruction target by searching for the best match in the local prior space (by optimizing the parameters encoding the local context) at the given query location. However, this requires the local context prior to generalize to a wide variety of unseen target regions, which is hard to achieve. To resolve this issue, we introduce Predictive Context Priors by learning Predictive Queries for each specific point cloud at inference time. Specifically, we first train a local context prior using a large point cloud dataset similar to previous techniques. For surface reconstruction at inference time, however, we specialize the local context prior into our Predictive Context Prior by learning Predictive Queries, which predict adjusted spatial query locations as displacements of the original locations. This leads to a global SDF that fits the specific point cloud the best. Intuitively, the query prediction enables us to flexibly search the learned local context prior over the entire prior space, rather than being restricted to the fixed query locations, and this improves the generalizability. Our method does not require ground truth signed distances, normals, or any additional procedure of signed distance fusion across overlapping regions. Our experimental results in surface reconstruction for single shapes or complex scenes show significant improvements over the state-of-the-art under widely used benchmark.

1. Introduction

Surface reconstruction from 3D point clouds estimates continuous surfaces from 3D point clouds that can be captured by various 3D sensors. This is still a challenge even with the help of state-of-the-art deep learning models. A standard strategy [1,14,19] is to first learn a Signed Distance Function (SDF) from a point cloud [1, 19] or from ground truth signed distances [14] using a neural network, and then reconstruct a surface based on the learned SDF via marching cubes [47]. If the SDF is trained to capture a global shape prior from a global 3D shape, however, it is hard to capture local geometry details.

As a remedy, state-of-the-art methods learn local SDFs from local regions [7, 37, 73]. The global shape is usually split into overlapping [37, 73] or non-overlapping [7] parts, and the local region prior is learned as a local SDF that is represented by a neural network with some parameters encoding the geometry of local regions. The intuition behind this idea is that the local region prior will generalize to various unseen local reconstruction targets, and for surface reconstruction at inference time, its parameters can be optimized to match the reconstruction target at specific locations. However, the matching requires the learned local region prior to cover as many specific locations on target regions as possible, which dramatically limits the generalization ability of the learned local prior.

To resolve this issue, we propose to learn SDFs as a Predictive Context Prior for highly accurate surface reconstruction from point clouds, as shown in Fig. 1. Specifically, we first train a neural network to represent local SDFs of local regions across a large dataset of point clouds. This aims to capture a local context prior in a local coordinate system, similar as in previous work. Our main contribution is that during surface reconstruction at inference time, we specialize the pre-trained local context prior into a Predictive...
try, one strategy is to leverage more latent codes with arbitrary topology. To reveal more detailed geometry in representing high-resolution shapes, other representations in representing high-resolution shapes can be used.

Deep Learning based Surface Reconstruction. The state-of-the-art methods mainly represent the reconstruction target as an implicit function \( f(\mathbf{x}) \), due to advantages of SDFs or occupancy fields over other representations in representing high resolution shapes with arbitrary topology. To reveal more detailed geometry, one strategy is to leverage more latent codes \( G \). This requires to split the point cloud into different voxels, and then represent the points in each voxel as a latent code that is either extracted by a neural network \( \{37, 46\} \) or learned in an auto-decoding manner \( \{7, 37\} \). These methods need normals for each point to produce signed distances as supervision in the optimization. Given the ground truth signed distances, Points2Surf \( \{14\} \) encodes points sampled in a local patch and on the whole point cloud as a shape prior, while DeepMLS \( \{44\} \) learns to produce oriented points to approximate SDFs. Similarly, PatchNet \( \{73\} \) learns local SDFs to represent patches with explicit control over positions, orientations, and scales. Neural-pull \( \{49\} \) introduced a new way of learning SDFs by pulling nearby space onto the surface, which is achieved by predicting the SDFs and its gradient using the network. This removes the requirement of ground truth normals or signed distances. A similar idea is introduced to learn unsigned distances \( \{11\} \), but requires to move dense sampling with additional directions to form the surface. Moreover, other novel ways for surface reconstruction have been proposed, such as a differentiable formulation of Poisson solver \( \{59\} \), point convolution \( \{6\} \) and part retrieval \( \{66\} \).

Other information is also leveraged to learn implicit functions \( \{52, 68, 83\} \). Occupancy is used to capture a prior at a global level \( \{36, 68\} \) or a local level \( \{52\} \). Iso-points \( \{83\} \) tried to impose geometry-aware sampling and regularization in the learning. Moreover, implicit functions can also be learned from point clouds with additional constraints, such as geometric regularization \( \{19\} \), sign agnostic learning with a specially designed loss function \( \{1\} \), sign agnostic learning with local surface self-similarities and post-sign processing \( \{71, 85\} \), constraints on gradients \( \{2\} \) or a divergence penalty \( \{4\} \).

From a meshing perspective, surfaces can also be reconstructed by generating local connectivity with intrinsic-extrinsic metrics \( \{43\} \), Delaunay triangulation of point clouds \( \{48\} \) or inheriting connectivity from an initial mesh \( \{30\} \). With local chart parameterizations in neural networks, a local point cloud is reconstructed via fitting using the Wasserstein distance as a measure of approximation \( \{79\} \).

Deep Shape Prior. Beside the priors reviewed above, shape priors can also be captured by parameters in neural networks in shape reconstruction \( \{3, 16, 20, 21, 25, 28, 34, 65, 74, 82\} \), segmentation \( \{45, 60, 61\} \), and completion \( \{32, 33, 76, 80, 84\} \). Deep manifold prior \( \{17\} \) was introduced to reconstruct 3D shapes starting from random initializations.

3. Method

Overview. We provide an overview of our method in Fig. 2. We aim to reconstruct a surface mesh for a 3D point cloud \( G \). Our method consists of the following three stages.
For surface reconstruction at test time, we specialize the signed distance $F$ as a local SDF $F_t$ in (a). We learn a predictive context prior to reconstruct $G$ by predicting queries $q'$ associated with conditions $f'_i$ for $F_t$ in (b).

1. During training, we start by learning a local context prior as a local SDF $F$ by training a neural implicit network under a local region set $T = \{t_i, i \in [1, I]\}$. As shown in Fig. 3 (a), the neural implicit network learns $F$ as a mapping from a query point $q_i$ with its corresponding condition $f_i$ to a signed distance $s$ in a local coordinate system.

2. For surface reconstruction at test time, we specialize the local context prior into a predicitive context prior for a specific point cloud $G$. Point cloud $G$ is located in a global coordinate system without normalization. We train an additional neural query network with parameters of $\theta_3$ specially for $G$, where we keep the neural network parameters $\theta_2$ representing the learned local context prior fixed. This leads to a global SDF $F_g'$ that captures the predictive context prior which we use to reconstruct the surface of $G$.

The neural query network learns to generate predictive queries, that is, to transform a query point $q_i$ around $G$ in a global coordinate system into a point $q_i'$ in the local coordinate system that the learned local context prior covers. In addition, the neural query network also predicts the condition $f'_i$ of predictive query $q'_i$. Here, we are inspired by the idea of ResNet [31], and predict the shift $\Delta q$ from $q_i$ to $q'_i$.

$$q'_i = q_i + \Delta q.$$  

The intuition behind the neural query network is to train a network specific to point cloud $G$ that is able to manipulate the queries for the learned local context prior. This prediction is equivalent to flexibly searching for correct information from the learned local SDF $F_t$, and then combining them together to fit the point cloud $G$. This leads to a global SDF $F_g'$ that predicts the signed distance $s'$ at a query location of $q_g$ with a condition of $G$,

$$s' = F_g(q_g, G) = F(q'_i, f'_i).$$  

To remove the requirement of ground truth signed distance values or normals in training, we minimize a pulling cost introduced in Neural-pull [49] to train the local SDF $F_t$. We simultaneously optimize the parameters of $\theta_1$ in PointNet and $\theta_2$ in the neural implicit network. The intuition of the pulling cost is to pull a query $q_i$ using the predicted signed distance $s$ to its nearest neighbor $nn(q_i)$ on region $t_i$ along the direction of the gradient $\nabla s = \partial F / \partial q_i$ at $q_i$. Our objective function during training is to minimize the pulling cost $C_{pull}$, where $nn(q_i) \in t_i$.

$$\min_{\theta_1, \theta_2} \|nn(q_i) - (q_i - s \times \nabla s / \|\nabla s\|_2)\|^2.$$  

**Predictive Context Prior.** For surface reconstruction at test time, we first specialize the learned local context prior into a predicitive context prior for a specific point cloud $G$. Point cloud $G$ is located in a global coordinate system without normalization. We train an additional neural query network with parameters of $\theta_3$ specially for $G$, where we keep the neural network parameters $\theta_2$ representing the learned local context prior fixed. This leads to a global SDF $F_g'$ that captures the predictive context prior which we use to reconstruct the surface of $G$.

The neural query network learns to generate predictive queries, that is, to transform a query point $q_i$ around $G$ in a global coordinate system into a point $q_i'$ in the local coordinate system that the learned local context prior covers. In addition, the neural query network also predicts the condition $f'_i$ of predictive query $q'_i$. Here, we are inspired by the idea of ResNet [31], and predict the shift $\Delta q$ from $q_i$ to $q'_i$.

$$q'_i = q_i + \Delta q.$$  

The intuition behind the neural query network is to train a network specific to point cloud $G$ that is able to manipulate the queries for the learned local context prior. This prediction is equivalent to flexibly searching for correct information from the learned local SDF $F_t$, and then combining them together to fit the point cloud $G$. This leads to a global SDF $F_g'$ that predicts the signed distance $s'$ at a query location of $q_g$ with a condition of $G$,

$$s' = F_g(q_g, G) = F(q'_i, f'_i).$$  

Similar to Eq. (2) in training, we further optimize the parameters $\theta_3$ of the neural query network to pull the query $q_g$ in the global coordinate system to its nearest neighbor $nn(q_g)$ on point cloud $G$. We leverage the learned local SDF to produce the gradient, $\nabla s' = \partial F / \partial q'_i$. So, our objective function during testing is to minimize a pulling cost $C_{pull}'$ below, where $nn(q_g) \in G$.
Intuition and Advantages. The intuition of our predictive context prior is to leverage information at different locations queried from the learned prior in local SDF $F_{g}$. This significantly improves the generalization ability of the learned local context prior.

Reconstruction. After we learn parameters $\theta_3$ in the neural query network using Eq. (5), we keep $\theta_3$ and the parameters $\theta_2$ in the neural implicit network fixed to produce the global SDF $F_{g}$ for point cloud $G$, which is then used to reconstruct the surface using marching cubes [47].

Optimization. We conduct the optimization of Eq. (2)(training) and Eq. (5)(inference) using a similar procedure. For each point $p$ on $t_i$ or $G$, we randomly sample 40 queries $q_i$ around $p$. Due to the difference numbers of points on each local region $t_i \in T$, we randomly select 2000 $q_i$ around $t_i$, and regard their nearest neighbors $\{nn(q_i)\}$ on $t_i$ as the input to PointNet in each training epoch, where the randomness makes the local context prior more robust to noise. We perform this optimization in an overfitting manner, either on each single $G$ or multiple point clouds with one-hot vectors as the condition of each $q_i$.

Implementation details. To predict a signed distance value $s$, we use an OccNet [51] without activation functions in the last layer. The condition $f_i^l$ or $f_i$ is 512 dimensional.
The neural query network is a feed-forward network with 8 layers, where each one of the first 7 layers has 512 nodes with ReLU activation functions while the last layer has 515 nodes with linear activation functions to predict the 512 dimensional condition $f'_i$ and 3 dimensional query $q'_i$.

We separate each 3D shape or scene in the training set under each benchmark into a $6^3$ grid according to its bounding box, where the points located in each grid form a local region $t_i$ in $T$. In addition, we use the same method as Neural-pull [49] to sample 40 queries $q_i$ or $q_j$ around each point $p_i$ or $G$, respectively, where a Gaussian function $N(p, \sigma^2)$ is used to calculate the sampling probability, and $\sigma^2$ is the 50-th nearest neighbor of $p$ on $t_i$ or $G$.

**Dataset.** In surface reconstruction for 3D shapes, we evaluate our method under three datasets including ABC [40], FAMOUS [14], and a subset of ShapeNet [8]. In surface reconstruction for scenes, we report our results under two datasets including 3D Scene [86] and SceneNet [29]. Under ShapeNet and ABC, we leverage marching cubes on a $128^3$ grid to reconstruct meshes, while using a $512^3$ grid under FAMOUS, 3D Scene and SceneNet.

**Metrics.** Under the ABC and FAMOUS datasets, we randomly sample $1 \times 10^5$ points on the reconstructed mesh to compare with the input point clouds using L2-CD which keeps the same as the setting in Points2Surf [14]. We also follow MeshingPoint [43] to report our results under ShapeNet [8] in terms of L1-CD, Normal Consistency (NC) [51], and F-score [72] to evaluate the reconstruction performance, where we evaluate the distance between the $1 \times 10^5$ points sampled on the reconstructed shape and the $1 \times 10^5$ ground truth points released by OccNet [51]. The threshold $\mu$ in F-score calculation is $0.002$ which is the same as MeshingPoint [43] and Neural-pull [49].

Under the 3D Scene [86] dataset, we follow DeepLS [7] to report the error between reconstructed meshes and the ground truth mesh. The error with a unit of mm is the average distance from each reconstructed vertex to its nearest triangle on the ground truth mesh. We also produce L1-CD, L2-CD, and normal consistency to compare with others.

Under the SceneNet [29] dataset, we follow LIG [37] to report L1-CD, Normal Consistency (NC) [51], and F-score [72] under different sampling densities on the reconstructed meshes, such as 20, 100, 500 and 1000 points per $m^2$, where the threshold $\mu_s$ in F-score calculation is $0.025$ m which is the same as LIG [37].

### 4.2. Surface Reconstruction for Single Shapes

**Evaluation under ShapeNet.** We first report our numerical comparison under the ShapeNet subset by comparing with the non data-driven and the latest data-driven methods in terms of L2-CD in Tab. 1, normal consistency in Tab. 2, and F-score with a threshold of $\mu$ in Tab. 3, and $2\mu$ in Tab. 4. The compared methods include Poisson Surface Reconstruction (PSR) [39], Ball-Pivoting algorithm (BPA) [5], AtlasNet (ATLAS) [20], Deep Geometric Prior (DGP) [79], Deep Marching Cube (DMC) [42], DeepSDF (DSDF) [58], MeshP [43], Neural Unsigned Distance (NUD) [11], SALD [2], Local Implicit Grid (LIG) [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].

The reconstruction accuracy in Tab. 1 demonstrates that our method reveals the most accurate surface from point clouds even under some challenging classes, such as Lamp, Chair, and Table. Although we achieve comparable normal consistency to MeshP in Tab. 2, we do not require dense and clean point clouds as MeshP. In addition, our method outperforms all implicit function based methods including DSDF [58], NUD [11], SALD [2], LIG [37], IMNET [10], and Neural-Pull (NP) [49].
DeepSDF (DSDF) [58], AtlasNet (ATLAS) [20], PSR [39], Points2Surf (P2S) [14], IGR [19], Neural-Pull (NP) [49] and IMLS [44]. The numerical comparison shows that our method significantly outperforms the other methods. We also highlight our advantage by visually comparing with IGR, P2S, and NP under FAMOUS in Fig. 7 and under ABC in Fig. 8, where our reconstruction presents more geometry details with arbitrary topology.

We also evaluate our method under some variants of ABC and FAMOUS by adding different noise levels or changing the point density, which is released by P2S [14]. The L2-CD comparison in Tab. 7 demonstrates that our method is also good at resisting dramatic noise and density changes, and still achieves the best performance compared to the others.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DSDF</th>
<th>ATLAS</th>
<th>PSR</th>
<th>P2S</th>
<th>IGR</th>
<th>NP</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>12.51</td>
<td>4.04</td>
<td>3.29</td>
<td>2.14</td>
<td>0.72</td>
<td>0.566</td>
<td>0.488</td>
</tr>
<tr>
<td>F-max-noise</td>
<td>11.34</td>
<td>4.47</td>
<td>3.89</td>
<td>2.76</td>
<td>1.24</td>
<td>0.675</td>
<td>0.571</td>
</tr>
<tr>
<td>F-med-noise</td>
<td>9.89</td>
<td>4.54</td>
<td>1.80</td>
<td>1.51</td>
<td>0.28</td>
<td>0.798</td>
<td>0.071</td>
</tr>
<tr>
<td>F-var-noise</td>
<td>13.17</td>
<td>4.14</td>
<td>3.14</td>
<td>2.52</td>
<td>0.31</td>
<td>0.387</td>
<td>0.298</td>
</tr>
<tr>
<td>F-Sparse</td>
<td>10.41</td>
<td>4.91</td>
<td>2.17</td>
<td>1.93</td>
<td>0.84</td>
<td>-</td>
<td>0.083</td>
</tr>
<tr>
<td>F-Dense</td>
<td>9.49</td>
<td>4.35</td>
<td>1.60</td>
<td>1.33</td>
<td>0.22</td>
<td>-</td>
<td>0.087</td>
</tr>
<tr>
<td>Mean</td>
<td>11.73</td>
<td>4.30</td>
<td>3.10</td>
<td>2.23</td>
<td>0.60</td>
<td>-</td>
<td>0.266</td>
</tr>
</tbody>
</table>

Table 7. Noise and density in terms of L2-CD (×100).

4.3. Surface Reconstruction for Scenes

Evaluation under 3D Scene. We first evaluate our method by comparing with MPU [57], Convolutional OccNet (ConvOcc) [68], Local Implicit Grid (LIG) [37], Deep Local Shape (DeepLS) [7], and Neural-Pull (NP) [49] under five scenes in the 3D Scene dataset in Tab. 8. We use the official code of MPU and NP to produce their results, while using the trained ConvOcc and LIG from the author and normals of point clouds to report their results, where we do not use the post processing in LIG for fair comparison. Tab. 8 shows that our method can achieve much higher accuracy than these state-of-the-art methods in terms of different metrics, where we also do not require normals as LIG and DeepLS. The improvement over the-state-of-the-art is further demonstrated by the visual comparison in Fig. 9.

Evaluation under SceneNet. We compare our method with ConvOcc [68], LIG [37], and NP [49] under 5 classes in SceneNet. We produce the results of NP by training it using its code, while using the trained model of LIG to produce their results, where we do not leverage the post processing in LIG for fair comparison. The results in each of five classes in Tab. 9 show that our method achieves the best performances under different input point densities. Our visual comparison in Fig. 11 further shows that our method can reconstruct more detailed surfaces in complex scenes.

Reconstructions for Real Scan. We also show surface reconstruction comparison for a real scanned scene in our video and text supplementary.

4.4. Analysis and Discussion

We justify the effectiveness of each element in our network and explore the effect of some important parameters on the performance under the ABC dataset in terms of L2-CD and normal consistency.
Ablation Studies. We report ablation studies in Tab. 10. We first highlight the effectiveness of the predicted shift \( \Delta q \) by removing \( \Delta q \) from the network. We first try to directly use the query \( q \) from the global coordinate system as \( q' \). We found that the performance degenerates dramatically, as shown by “No \( \Delta q \)”. Then, we push the neural query network to predict \( q' \) directly. But the performance still goes down, as shown by “Direct \( q' \)”. These two results demonstrate the importance of \( \Delta q \) in the learning.

Moreover, we highlight the effectiveness of the predicted condition \( f'_1 \) by removing it from the output of the neural query network. We first leverage autodecoding similar as DeepSDF [58] to learn \( f'_1 \). The result of “No \( f'_1 \)” shows that this does not work well with the learnable \( \Delta q \). Then, we try to use \( f_1 \) from the trained PointNet to replace \( f'_1 \), but the result of “No \( f'_1+f_1 \)” gets worse neither. These experiments show that the learnable condition \( f'_1 \) is only effective when it is optimized together with its corresponding query \( q'_1 \).

Specializing Context Prior. We found that specializing the local context prior into the predictive context prior still affects the learning of the local context prior as an initialization, if we tune \( \theta_2 \) and \( \theta_3 \) simultaneously in Fig. 3(b). The result of “Tune \( \theta_2+\theta_3 \)” is still not satisfactory. These experiments demonstrate the importance of the specializing in leveraging the learned prior.

<table>
<thead>
<tr>
<th>No Prior</th>
<th>Tune ( \theta_2+\theta_3 )</th>
<th>Our specializing</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2-CD</td>
<td>4.04</td>
<td>2.09</td>
</tr>
<tr>
<td>Normal</td>
<td>0.9200</td>
<td>0.9446</td>
</tr>
</tbody>
</table>

Table 11. Effect of specializing under ABC. L2-CD \( \times \) 1000.

Normalizing Local Regions. We found the normalization of local regions \( t_i \) in \( T \) slightly affects the learning of the local context prior. As we mentioned before, we normalize \( t_i \) by centering and scaling it in the local coordinate system. We report the effect of centering and scaling on the performance in Tab. 12, which shows that both centering and scaling contribute to the increase of performance.

<table>
<thead>
<tr>
<th>No normalization</th>
<th>Only centering</th>
<th>Only scale</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2-CD</td>
<td>2.83</td>
<td>2.13</td>
<td>2.67</td>
</tr>
<tr>
<td>Normal</td>
<td>0.9360</td>
<td>0.9410</td>
<td>0.9380</td>
</tr>
</tbody>
</table>

Table 12. Effect of normalization under ABC. L2-CD \( \times \) 1000.
then, use each set of parts to learn the local context prior in Fig. 3 (a). The comparison shown in Tab. 13 demonstrates that it is hard to learn the prior well if the size of $t_i$ is too large, such as “t$^3$” and “4t$^3$”, since $t_i$ is too complex to learn. While it is also hard to learn some meaningful prior if the size of $t_i$ is too small, such as “S$^3$”. In addition, we found that the overlap between neighboring local regions does not contribute to the performance increasing under 6$^3$, such as “Lap” and “Self”. We also explore whether we can learn a more meaningful prior by using patch-wise $t_i$ in training. We form each $t_i$ using 1000, 2000, or 4000 neighbors in terms of geodesic distance. The results of “G1”, “G2” and “G4” show that patch-wise regions $t_i$ do not work better than the part-wise $t_i$ that we are using.

We highlight our advantage by learning the local context prior using local regions $t_i$ merely from the reconstruction target $G$. Although “Self” is obtained with much fewer training regions, it achieves almost the same result as “6$^3$” which is obtained by learning the local context prior from all local regions across different training shapes. This advantage comes from our ability of flexibly searching over the whole prior space, which alleviates the necessity of learning a high quality local context prior. However, the optimization can converge faster if more local regions $t_i$ are used in learning as shown by the loss curve comparison in Fig. 10.

### Limitation
Although we achieve high reconstruction accuracy, we require further optimization during testing. This takes more time than methods [14] leveraging pretrained models for inference.

### 5. Conclusion
We propose to reconstruct surfaces from point clouds by learning implicit functions as a predictive context prior. Our method successfully specializes the learned local context prior into predictive context prior for a specific point cloud, which effectively searches the reconstruction prior across the whole prior space without focusing on some specific locations. This advantage significantly increases our ability of leveraging the learned prior, which makes the learned prior generalize to as many unseen target regions as possible. Our idea is justified by our experimental results which outperform the state-of-the-art in terms of various metrics under widely used benchmarks.
References


[22] Zhizhong Han, Chao Chen, Yu-Shen Liu, and Matthias Zwicker. ShapeCaptioner: Generative caption network for 3D shapes by learning a mapping from parts detected in multiple views to sentences. In ACM International Conference on Multimedia, 2020. 2


[27] Zhizhong Han, Mingyang Shang, Xiyang Wang, Yu-Shen Liu, and Matthias Zwicker. Y2Seq2Seq: Cross-modal representation learning for 3D shape and text by joint reconstruction and prediction of view and word sequences. In AAAI, pages 126–133, 2019. 2


[33] Tao Hu, Zhizhong Han, and Matthias Zwicker. 3D shape completion with multi-view consistent inference. In AAAI, 2020. 2


[38] Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. SDFDiff: Differentiable rendering of signed distance fields for 3D shape optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2


[45] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Point2Sequence: Learning the shape representation of 3D point clouds with an attention-based sequence to sequence network. In AAAI, pages 8778–8785, 2019. 2


[75] Xin Wen, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. Cycle4completion: Unpaired point cloud completion using cycle transformation with missing region coding. In IEEE Conference on Computer Vision and Pattern Recognition, 2021. 2

[76] Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. Point cloud completion by skip-attention network with hierarchical folding. In IEEE Conference on Computer Vision and Pattern Recognition, 2020. 2


[78] Xin Wen, Junsheng Zhou, Yu-Shen Liu, Hua Su, Zhen Dong, and Zhizhong Han. 3D shape reconstruction from 2D images with disentangled attribute flow. In IEEE Conference on Computer Vision and Pattern Recognition, 2019. 2, 5

[79] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Ioan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In IEEE Conference on Computer Vision and Pattern Recognition, 2022. 2


