DITTO: Dual and Integrated Latent Topologies for Implicit 3D Reconstruction

Jaehyeok Shim and Kyungdon Joo*

Artificial Intelligence Graduate School, UNIST

{jh.shim,kyungdon}@unist.ac.kr

https://vision3d-lab.github.io/ditto

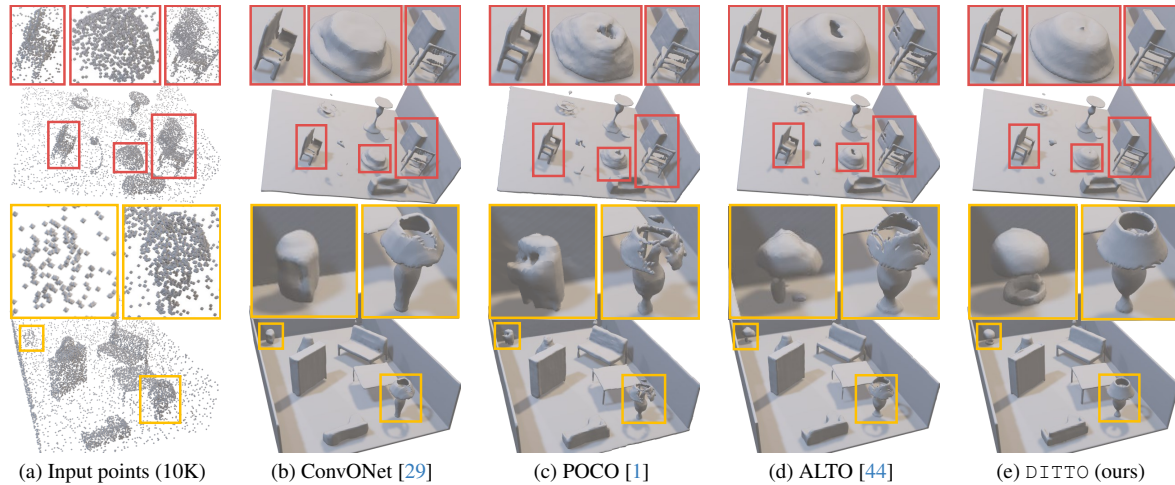|				
(a) Input points (10K)	(b) ConvONet [29]	(c) POCO [1]	(d) ALTO [44]	(e) DITTO (ours)

Figure 1. **Scene-level 3D reconstruction comparison on the Synthetic Rooms dataset [29].** DITTO maximizes the benefits of both grid and point latents, thereby improving 3D surface reconstruction performance. We particularly focus on refining features based on point latents along with grid latents and integrating them (namely, dual and integrated latent topologies). This advancement enhances the ability to restore complex structures precisely, such as thin and intricate geometries, which posed challenges for previous methods.

Abstract

We propose a novel concept of dual and integrated latent topologies (DITTO in short) for implicit 3D reconstruction from noisy and sparse point clouds. Most existing methods predominantly focus on single latent type, such as point or grid latents. In contrast, the proposed DITTO leverages both point and grid latents (i.e., dual latent) to enhance their strengths, the stability of grid latents and the detail-rich capability of point latents. Concretely, DITTO consists of dual latent encoder and integrated implicit decoder. In the dual latent encoder, a dual latent layer, which is the key module block composing the encoder, refines both latents in parallel, maintaining their distinct shapes and enabling recursive interaction. Notably, a newly proposed dynamic sparse point transformer within the dual latent layer effectively refines point latents. Then, the integrated implicit decoder systematically combines these refined latents, achieving high-fidelity 3D reconstruction and surpassing previous state-of-the-art methods on object- and scene-level datasets, especially in thin and detailed structures.

1. Introduction

Implicit 3D reconstruction aims to determine surface boundaries by estimating implicit values, such as occupancy and signed distance fields, based on given query coordinates [24]. In particular, implicit 3D reconstruction has evolved using geometric primitives like vectors [6, 24, 27, 32], grids [19, 29, 30, 37], and point clouds [1, 43] as intermediaries, namely, *latent representations*. Prior studies have focused on selecting appropriate latent representations for 3D reconstruction. Specifically, early methods [6, 24, 27, 32] use vectors as their latent representation because of their simplicity. However, they fall short in handling large-scale scenes due to the absence of a geometric prior (*i.e.*, positional information). To alleviate this issue, subsequent methods based on grid latent have emerged [7, 19, 29, 37]. Grid latents have similar shapes with occupancy cube, the target domain of 3D reconstruction. Thus, they offer high-fidelity reconstructions at the scene-level but often lack detail because of resolution constraints. On the other hand, point latent-based approaches [1, 43] enable detailed reconstruction because they preserve the details of the input points without information loss (*e.g.*, quantization). However, they can produce unsta-
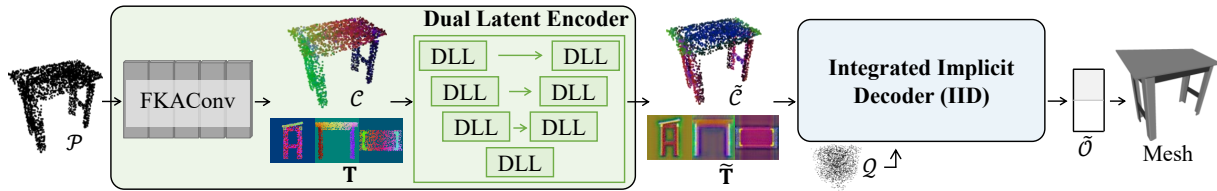
*Corresponding author.

Figure 2. **Overview of DITTO**. DITTO architecture consists of the proposed dual latent encoder and integrated implicit decoder (IID) modules. In the encoder, DITTO receives a point cloud $\mathcal{P}$ and generates point and grid latents $\mathcal{C}$ and $\mathbf{T}$, respectively, using shallow FKAConv layers [2]. These latents are refined in a U-shaped network composed with the proposed DLL to produce refined point and grid latents, respectively $\tilde{\mathcal{C}}$ and $\tilde{\mathbf{T}}$. Our IID estimates the occupancy of given arbitrary query locations. The mesh can be obtained by applying the Marching Cubes algorithm [22] to the occupancies estimated as form of a regular grid.

ble results due to ambiguities, such as holes in thin structures, because they can be sensitive to the noise inherent in the input point coordinates. In short, each latent has its own pros and cons, so it is crucial to leverage the strengths of each representation properly.

As an attempt to combine the strengths of each latent, Wang *et al.* [44] introduces a new alternating latent topology concept, so-called ALTO. Concretely, ALTO simultaneously utilizes two latent representations by alternatively projecting one latent into another. ALTO then decodes the combined features in the form of a grid for 3D reconstruction. Such an intuitive and alternative approach improves 3D reconstruction performance and is meaningful as a first attempt. However, ALTO may overlook the advantages of abundant features extractable from point latents and makes the implicit decoder rely solely on a grid latent-based decoder for convenience.

In this work, we propose a novel concept of dual and integrated latent topologies (DITTO) for implicit 3D reconstruction. The proposed DITTO aims to systematically integrate the strengths of each latent while maintaining their spatial structure of point and grid latents (*i.e.*, dual latent). Specifically, we seek to offset the inherent ambiguity of point latents through the stability of grid latents and, conversely, complement the resolution constraints of grid latents through the detailed representation by point latents.

The proposed DITTO employs an encoder-decoder architecture for dual latent (see Fig. 2). From a given point cloud, our encoder, called *dual latent encoder*, constructs point and grid latents and refines this dual latent while preserving their original shapes. In particular, we propose a new dynamic sparse point transformer (DSPT) for point features, which leverages large receptive fields, enabling effective learning of point-based spatial patterns. Based on DSPT, we design a dual latent layer (DLL) that iteratively and separately updates dual latent with the correlation between two latents. This DLL module allows us to implicitly learn challenging patterns, such as thin objects, that cannot be handled by grid latent alone. Then, our decoder, called *integrated implicit decoder (IID)*, integrates enhanced dual latent to estimate the implicit value. Unlike previous meth-

ods that utilize only a subset of latents, our decoder considers dual latent; especially, we unify grid-based and point-based implicit decoders together by introducing the concept of integrated latent. IID helps to restore details by adjusting the relationship between neighbor points and query, adapting to the surface proximity. Finally, the proposed DITTO improves 3D surface reconstruction performance, outperforming previous approaches and establishing a new state-of-the-art (see Fig. 1).

The main contributions of DITTO are as follows:
- DITTO is a new implicit 3D reconstruction method focusing on advanced feature extraction and fusion of grid and point latents, enhancing 3D understanding capabilities.
- We design a new dual latent layer module that refines dual latent while preserving their individual strengths. Particularly, we present a dynamic sparse point transformer (DSPT) to emphasize point feature refinement.
- We present a novel integrated implicit decoder that uniquely integrates two latents, providing clear surface boundaries, and robust to thin and intricate structures.

2. Related Work

3D reconstruction can be explicitly represented using a variety of geometric primitives, such as point, voxel, and mesh, or it can be inherently represented by leveraging such geometric primitives as latent representations [1, 6, 19, 24, 27, 29, 37, 40, 43, 44]. We refer to the former as explicit 3D reconstruction and the latter as implicit 3D reconstruction. The readers refer to [44] for explicit representations. In this section, we discuss the strengths and weaknesses of each latent for implicit 3D reconstructions.

Vector Latent Topologies. Early approaches, such as [6, 24, 27], employ an encoder-decoder architecture in a similar manner. They encode a 3D shape into vector latents by the encoder and then reconstruct the 3D shape by the decoder. The decoder estimates the implicit values of a given query point at arbitrary locations. However, since vector latents lack geometric priors, they show decreased detailed reconstruction performance (*i.e.*, vector latent-based methods are vulnerable in large scenes with complex geometry).

Grid Latent Topologies. As an alternative to vector latents, grid latents that encode geometric priors have been proposed [29, 30, 37, 40, 44]. Grid latent-based methods quantize the input point cloud into a grid latent during encoding. Then, they extract latent features at the query points using linear interpolation of the adjacent grids during decoding.

According to geometric primitives, we can divide grid latent-based methods into voxel-based and triplane-based methods. Voxel latents can densely store latent features in the form of 3D grids [12, 26, 33, 36, 45]. ConvONet [29] expands the previous ONet [24] by utilizing voxel latents instead of vector latents. ConvONet and ALTO [44] take voxel latents as one of the base representations, which reveal their effectiveness, especially for scene-level reconstructions. However, voxels require cubic computation and inherently have limited resolutions. On the other hand, triplanes use 2D planes, which have less resolution constraints, allowing higher resolution than voxels. This advantage leads to more effective restoration than voxels, particularly for object-level tasks [3, 5, 8, 11, 13, 14, 34, 42], which has slightly lower geometric complexity than scenes. For this reason, ConvONet and ALTO use triplanes as the primary latent topologies in object-level reconstruction.

Even though grid latent-based approaches show effective reconstruction, but still require point feature quantization. This process can result in a loss of fine details of the 3D surface, which is a fundamental limitation of grid latents.

Point Latent Topologies. There are a few methods [1, 43, 46, 47] that encode features into point latents, typically offering benefits for the preservation of spatial information. Namely, there is no need for quantization, preventing the loss of details. POCO [1] encodes latent features into point latents and further enhances each feature by leveraging the point latents of neighbor points through point convolution [2] and attention mechanism [39]. ARO-Net [43] improves point features by introducing methods like anchor points and radial observations, enhancing the performance of the point-based implicit decoder. However, these point latent-based methods may suffer instability from preserving spatial information of the point cloud, including even noise points. Concretely, point latents from noise points can affect neighbor points. In addition, the query feature extraction process may introduce ambiguity since different queries can share the same neighbor points. We address these issues of point latents by leveraging the stability of grid latents.

Blended Latent Topologies. ALTO [44] introduces a new method that leverages multiple types of feature representations, utilizing both grid and point latents. This attempt is the first approach to combine the strengths of each latent. Specifically, ALTO iteratively projects its features from grid to point cloud and vice versa. By doing this, ALTO aims
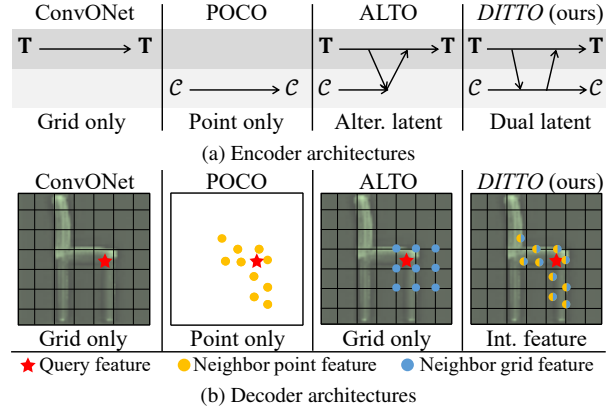

Figure 3. **Conceptual comparison of DITTO.** We compare the concept of implicit 3D reconstruction methods in terms of latent representations: (a) encoders and (b) decoders. In (b), the image of green chairs represents grid features.

to preserve the details inherent in the point cloud and enable feature sharing between planes, leading to improved detailed surface reconstruction. However, ALTO primarily relies on grid latents, with limited feature extraction from point latents, hindering its capacity to fully exploit the potential of point latents. In particular, its decoder exclusively employs grid latents, making it directly susceptible to the resolution constraints inherent in grid latents.

To address these issues, we propose DITTO designed with advanced 3D geometry understanding capabilities. DITTO comprises an enhanced point encoder, based on FKAConv [2], and advanced module for extracting features from both point and grid latents. Furthermore, we present an implicit decoder that leverages the fusion of both grid and point latents for improved performance.

3. Dual and Integrated Latent Topologies

In this section, we propose a new topological concept, *dual and integrated latent topologies* (DITTO), for implicit 3D reconstruction from a given noisy and sparse point cloud. DITTO employs dual latent (*i.e.*, point and grid latents) to leverage both the structural stability of grid latents and the preciseness of point latents. DITTO, composed of encoder-decoder architecture for dual latent, refines and integrates these two latents, overcoming individual limitations and improving overall efficacy. This strategy leads to high-fidelity surface reconstruction, even for thin, intricate structures.

3.1. Overview

Given a noisy and sparse point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^N$ as input, the goal of DITTO is to accurately reconstruct 3D surfaces in a form of occupancy $\mathcal{O} = \{o_j \in \{0,1\}\}_{j=1}^M$ for query coordinates of arbitrary location $\mathcal{Q} = \{\mathbf{q}_j \in \mathbb{R}^3\}_{j=1}^M$, where N and M are the number of input points and queries.

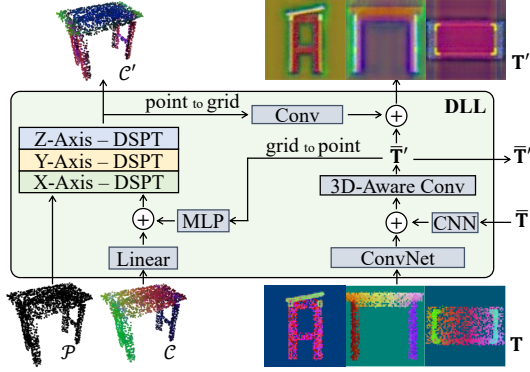The proposed DITTO comprises two main parts: dual la-

Figure 4. **Overview of our proposed DLL**. The input consists of $\mathcal{C}$ and $\mathbf{T}$, while the output comprises $\mathcal{C}'$ and $\mathbf{T}'$, representing the grid and point latents of the current layer, respectively. The $\bar{\mathbf{T}}$ and $\bar{\mathbf{T}}'$ represent the input and output grid latents, respectively, used for establishing a dense skip-connection between the layers.

tent encoder and integrated implicit decoder (see Fig. 2). In the dual latent encoder, we first extract the point latents $\mathcal{C} = \{\mathbf{c}_i \in \mathbb{R}^d\}_{i=1}^{N}$ for $\mathcal{P}$ based on FKAConv layers [2], where d is the dimension of the point latents. We then project $\mathcal{C}$ to grid latents. Following the convention [29, 44], we use triplanes $\mathbf{T} \in \mathbb{R}^{3 \times R \times R \times d}$ or voxels $\mathbf{V} \in \mathbb{R}^{R \times R \times R \times d}$ as grid latents, where R is the resolution of grids. In this section, we explain the details of DITTO based on triplanes as grid latents, but they can seamlessly be replaced with voxels. After extracting the initial dual latent ($\mathcal{C}$ and $\mathbf{T}$), we refine them using a UNet architecture [31], where each layer consists of the proposed dual latent layer (DLL). This UNet estimates refined grid $\tilde{\mathbf{T}}$ and point $\tilde{\mathcal{C}}$ latents. The detailed description of DLL is in Sec. 3.2. Then, integrated implicit decoder (IID) estimates occupancy probability $\tilde{\mathcal{O}} = \{\tilde{o}_j \in \mathbb{R}\}_{j=1}^{M}$ of given query coordinates $\mathcal{Q}$ by integrating latents $\tilde{\mathbf{T}}$ and $\tilde{\mathcal{C}}$. IID effectively manipulates the distinct characteristics of grid and point latents, facilitating the reconstruction of detailed and thin structures. The detailed description of IID is in Sec. 3.3. More detailed network architectures are provided in supplementary materials.

3.2. Dual Latent Layer

We present a dual latent layer (DLL) that adeptly combines the advantages of both grid and point latents. Specifically, DLL individually enhances two latents and infuses correlation between them to optimize their combined performance while preserving their own strengths, instead of combining them into a single latent (see Fig. 3a). In particular, we newly present a dynamic sparse point transformer (DSPT) as a point feature extractor. DSPT boosts point feature refinement by directly employing a transformer to point-based dynamic windows.

Dynamic Sparse Point Transformer. The concept of DSPT is to apply a transformer directly to points, stimulat-
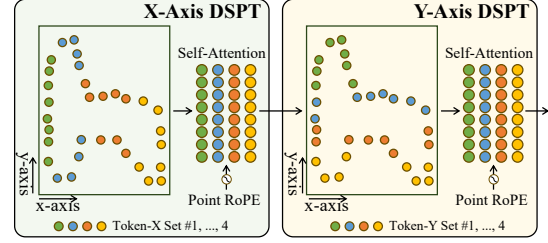

Figure 5. **Conceptual illustration of DSPT.** We visualize DSPT in the 2D domain for better understanding, but DSPT works in the 3D domain by sequentially processing the x-, y-, and z-axes.

ing local-global interactions among point latents. Inspired by the concept of windowed attention [20], we create point-based non-overlapping windows and apply self-attention to each window. However, unlike images or voxels having regular grids, points have free form and are unordered. To alleviate this issue, we adopt a sorting-based windowing scheme, motivated by DSVT [41], FlatFormer [21] and CSwinTransformer [10]. Concretely, we sort points for a certain axis and divide them into multiple windows so that each window has an equal number of points. Subsequently, self-attention is applied within each window. In a DSPT block, we repeat this procedure for each of the x-, y- and z-axes (see 2D illustration of DSPT in Fig. 5).

Here, we describe the details of DSPT. For given point features $\mathcal{C}$ and their coordinates $\mathcal{P}$, we first sort $\mathcal{C}$ based on $\mathcal{P}$ along a specific axis. We then divide the sorted point features into windows $\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L}$ so that the number of points belonging to each window is equal:

$$\{\mathcal{C}_l^{\text{wnd}}\}_{l=1}^{L} = split(sort_{\text{x}}(\mathcal{C}, \mathcal{P}), L),$$
$$\text{where} \quad \mathcal{C}_l^{\text{wnd}} = \{\mathbf{c}_{i_{\text{sorted}}} \in \mathbb{R}^d\}_{i_{\text{sorted}}=1}^{N/L}, \quad (1)$$

L is the number of windows, $sort_{\text{x}}(\cdot, \cdot)$ denotes a sort function, which sorts $\mathcal{C}$ w.r.t. their coordinates $\mathcal{P}$ of x-axis, $split(\cdot, \cdot)$ denotes a splitting function, which divides the given $\mathcal{C}$ into L windows, and i_{sorted} indicates the sorted index. For convenience, we use the x-axis in Eq. (1), but we can also apply the y- and z-axes. Before applying self-attention, a rotary positional embedding (RoPE) [35], modified one for point clouds [18], is applied to the query and key of the inputs of self-attention. Subsequently, a shared MLP is applied to each $\mathbf{c}$, and the points are re-sorted in reverse to restore the original point order.

Overall Architecture of DLL for Dual Latent. The proposed DLL takes dual latent, $\mathcal{C}$ and $\mathbf{T}$, from the previous DLL as input and systematically refines each latent feature while maintaining their shape and strengths (see Fig. 4). Within DLL, we first enhance grid latents $\mathbf{T}$ using a simple CNN-based architecture. Note that when the triplane is used as grid latents, 3D-aware-conv [42] is additionally applied to induce feature exchange among three planes. We call this enhanced grid latents as intermediate grid features $\bar{\mathbf{T}}'$,

which is used to refine $\mathcal{C}$ and form a dense skip-connection between consecutive DLL modules, similar to [44].

In the point latent perspective, we combine $\mathcal{C}$ with the point features projected from $\bar{\mathbf{T}}$, inducing information of stable grid latents to point latents. We then refine these point latents using the proposed DSPT. We denote these refined point latents as $\mathcal{C}'$. In the grid latent perspective, we project $\mathcal{C}'$ into grid domain and merge with intermediate latent $\bar{\mathbf{T}}'$, forming enhanced grid latents $\mathbf{T}'$, which creates a synergy of dual latent.

3.3. Integrated Implicit Decoder

We present an integrated implicit decoder (IID) that estimates implicit values (*i.e.*, occupancy) for a given query location using refined dual latent $\tilde{\mathbf{T}}$ and $\tilde{\mathcal{C}}$. In particular, IID integrates the grid and point latents while reinforcing the strengths and compensating for shortcomings associated with each latent (see Fig. 3b). To this end, we first analyze the pros and cons of each latent in terms of decoding perspective and then introduce IID in detail.

Point Latents *vs.* Grid Latents in Decoding. Point-based implicit decoding approaches like POCO [1] utilize K-nearest neighbors (KNN) to define neighbor points for a given query point and use features of neighbor points to calculate query feature. This point-based decoder is advantageous for detail restoration. However, such point-based decoders can be fragile when handling thin structures due to inherent ambiguity because different query points may share the same neighbors, resulting in instability.

Grid-based decoding methods (*e.g.*, ConvONet [29] and ALTO [44]) estimate occupancy by interpolating features from adjacent grids for a given query to determine the query feature. Additionally, ALTO employs an attention mechanism to alleviate the resolution constraints inherent to grid latents, thereby enhancing performance. Specifically, ALTO compares query feature and adjacent grid features through subtraction-based cross-attention [48]. This comparison allows ALTO to use not only the query feature but also the varying patterns of adjacent grid features, mitigating the resolution constraints of grid latents. However, despite this improvement, detailed reconstruction remains limited due to the inherent limitations of relying solely on grid latents.

Integrated Decoder for Dual Latent. The proposed IID selectively integrates the advantages of each latent decoding method (*i.e.*, a hybrid approach between point and grid latent decoding). We basically adopt KNN-based neighbors of point latents to determine neighbor point features since it can recover the detailed reconstruction free from the resolution limit. Instead, we handle the inherent limitation of point features by using grid-based decoding. That is, we combine point features at neighbor locations with adjacent grid features via interpolation. This integration en-
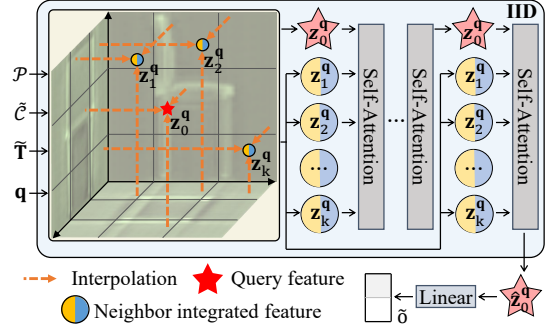

Figure 6. **Illustration of IID.** IID makes query feature $\mathbf{z}_0^{\mathbf{q}}$ and neighbor integrated features $\{\mathbf{z}_1^{\mathbf{q}}, \ldots, \mathbf{z}_k^{\mathbf{q}}\}$. Then, IID iteratively applies self-attention to a sequence of $\{\mathbf{z}_0^{\mathbf{q}}, \ldots, \mathbf{z}_k^{\mathbf{q}}\}$ to update query feature $\mathbf{z}_0^{\mathbf{q}}$. Finally, we estimate the query feature $\hat{\mathbf{z}}_0^{\mathbf{q}}$ and apply linear layer to estimate occupancy $\tilde{o}$.

ables IID to effectively consider both latents. We call these combined features as neighbor integrated features. Subsequently, we estimate implicit value through self-attention between neighbor integrated features and query feature obtained from grid latents. Note that since neighbor points are located around the surface, the region of interest (*i.e.*, size of the neighbor integrated features) is adaptively defined *w.r.t.* query-neighbor distances. This adaptive mechanism facilitates the reconstruction of clear surface boundaries of detailed and intricate structures.

The detailed IID is visualized in Fig. 6. First, we estimate query feature $\mathbf{z}_0^{\mathbf{q}}$ by interpolating the grid features as in [29, 44] and applying linear layer so that the query feature has $2d$ dimensions:

$$\mathbf{z}_0^{\mathbf{q}} = interpolate(\tilde{\mathbf{T}}, \mathbf{q})\mathbf{W}, \quad \mathbf{z}_0^{\mathbf{q}} \in \mathbb{R}^{2d}, \qquad (2)$$

where $interpolate(\cdot, \cdot)$ computes a grid feature of a given location using linear interpolation for adjacent grid latents, and $\mathbf{W} \in \mathbb{R}^{d \times 2d}$ is a weight matrix of linear layer. Subsequently, we find neighbor points $\mathcal{P}^{\mathbf{q}}$ and neighbor point features $\tilde{\mathcal{C}}^{\mathbf{q}}$ of the query point $\mathbf{q}$ by using KNN:

$$(\mathcal{P}^{\mathbf{q}}, \tilde{\mathcal{C}}^{\mathbf{q}}) = KNN(\mathbf{q}, \mathcal{P}, \tilde{\mathcal{C}}, K), \qquad (3)$$

where $KNN(\cdot, \cdot, \cdot, \cdot)$ returns given K number of neighbor point coordinates $\mathcal{P}^{\mathbf{q}} = \{\mathbf{p}_k^{\mathbf{q}} \in \mathbb{R}^3\}_{k=1}^K$ and their features $\tilde{\mathcal{C}}^{\mathbf{q}} = \{\tilde{\mathbf{c}}_k^{\mathbf{q}} \in \mathbb{R}^d\}_{k=1}^K$. After that, we compute neighbor grid features $\mathbf{T}^{\mathbf{q}}$ by interpolating grid latents for every $\mathbf{p}_k^{\mathbf{q}}$:

$$\tilde{\mathbf{T}}^{\mathbf{q}} = \{interpolate(\tilde{\mathbf{T}}, \mathbf{p}_k^{\mathbf{q}})\}_{k=1}^K. \qquad (4)$$

Then, we construct the neighbor integrated features $\mathcal{Z}^{\mathbf{q}}$ by concatenating $\tilde{\mathcal{C}}^{\mathbf{q}}$ and $\tilde{\mathbf{T}}^{\mathbf{q}}$ in channel direction:

$$\mathcal{Z}^{\mathbf{q}} = concat(\tilde{\mathcal{C}}^{\mathbf{q}}, \tilde{\mathbf{T}}^{\mathbf{q}}) = \{\mathbf{z}_k^{\mathbf{q}} \in \mathbb{R}^{2d}\}_{k=1}^K, \qquad (5)$$

where $concat(\cdot, \cdot)$ is a concatenation function. Then, we refine $\mathbf{z}_0^{\mathbf{q}}$ by applying self-attention multiple times on a sequence $\{\mathbf{z}_0^{\mathbf{q}}, \ldots, \mathbf{z}_K^{\mathbf{q}}\}$ including both $\mathbf{z}_0^{\mathbf{q}}$ and $\mathcal{Z}^{\mathbf{q}}$. We denote $\hat{\mathbf{z}}_0^{\mathbf{q}}$ as the refined query feature. While applying self-attention, we use point-based RoPE [35] as mentioned in

Table 1. **Object-level quantitative comparison on ShapeNet**. From left to right, the input point clouds become sparse (variations in density); they have 3K, 1K, and 300 points, respectively. The best scores are **red** in bold, and the secondary ones are <u>blue</u> with underline.

Method	normal (3K points & noise level 0.005)				sparse (1K points & noise level 0.005)				sparse (300 points & noise level 0.005)			
	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑
ONet [24]	0.761	0.87	0.891	0.785	0.772	0.81	0.894	0.801	0.778	0.80	0.895	0.806
ConvONet [29]	0.884	0.44	0.938	0.942	0.859	0.50	0.929	0.918	0.821	0.59	0.907	0.883
POCO [1]	0.926	<u>0.30</u>	0.950	<u>0.984</u>	0.884	0.40	0.928	0.950	0.808	0.61	0.892	0.869
ALTO [44]	<u>0.930</u>	<u>0.30</u>	<u>0.952</u>	0.980	<u>0.905</u>	<u>0.35</u>	<u>0.940</u>	<u>0.964</u>	<u>0.863</u>	<u>0.47</u>	<u>0.922</u>	<u>0.924</u>
DITTO (ours)	**0.949**	**0.27**	**0.957**	**0.988**	**0.926**	**0.32**	**0.949**	**0.975**	**0.882**	**0.43**	**0.931**	**0.940**

(a) GT Mesh   (b) Input points   (c) ConvONet [29]   (d) POCO [1]   (e) ALTO [44]   (f) DITTO (ours)
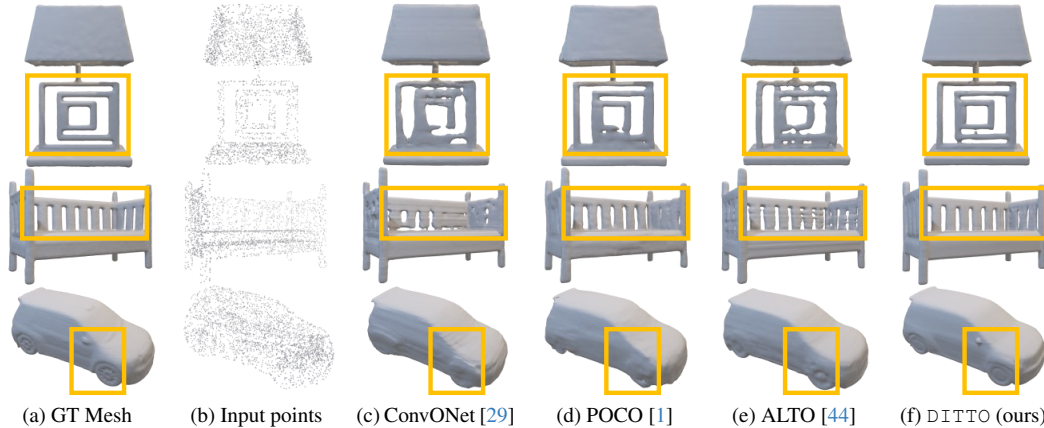
Figure 7. **Object-level 3D reconstruction comparison on ShapeNet [4] with 3K input points**. DITTO distinctively excels in reconstructing thin structures, evidenced by the intricate details of lamps and benches. Uniquely, DITTO is the only method that accurately captures the side mirror and the fine details of car wheels.

Sec. 3.2. Note that during the self-attention, we update only $\mathbf{z}_0^{\mathbf{q}}$ and the other elements of the sequence $\{\mathbf{z}_1^{\mathbf{q}}, \ldots, \mathbf{z}_K^{\mathbf{q}}\}$ remain unchanged. Finally, we estimate occupancy $\tilde{o}$ by applying linear layer to $\hat{\mathbf{z}}_0^{\mathbf{q}}$:

$$\tilde{o} = \hat{\mathbf{z}}_0^{\mathbf{q}} \mathbf{W}_{\text{out}}, \qquad (6)$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{2d}$ is a weight matrix of linear layer.

3.4. Training Objectives

We use binary cross entropy objective between $\tilde{\mathcal{O}}$ and $\mathcal{O}$:

$$\mathcal{L}(\tilde{\mathcal{O}}, \mathcal{O}) = -\frac{1}{M} \sum_{j=1}^{M} [o_j \log(\tilde{o}_j) + (1 - o_j) \log(1 - \tilde{o}_j)]. \quad (7)$$

4. Experiments

We evaluate DITTO against SoTA methods. Details of implementation are provided in Sec. 4.1. Qualitative and quantitative comparisons for object-level and scene-level are in Sec. 4.2 and Sec. 4.3, respectively. In addition, we validate the generality in Sec. 4.4. Additional experiment results are available in the supplementary materials.

4.1. Baselines, Datasets, Metrics

Implementation Details. We implement DITTO in PyTorch [28], utilizing xFormers [17] and mixed-precision [25]. We train DITTO with Adam optimizer [16]

and cosine annealing learning rate scheduler [23]. For a fair comparison, we set the resolution of triplanes as $R{=}64$ and $R{=}128$ for object- and scene-level tasks, respectively. For voxels, we set $R{=}64$ in scene-level tasks. In DSPT, we use $L{=}25$ for $\{3\text{K}, 1\text{K}, 0.3\text{K}\}$ input points and $L{=}20$ for 10K input points. More detailed hyperparameters are described in the supplementary materials.

Comparison Methods. To assess the 3D reconstruction performance of DITTO, we compare it with various baseline methods. These methods include a non-learning-based method [15], as well as implicit methods that utilize diverse latent topologies, such as vector [24], grid [19, 29], point [1] and blended [44] latents. Our evaluation procedure primarily follows the previous SoTA method, ALTO, including several additional experiments.

Datasets. For the evaluation of object-level surface reconstruction, we use ShapeNet [4], which contains 13 categories of object watertight meshes. For assessment of scene-level surface reconstruction, we use the Synthetic Rooms dataset [29], which has 5K synthetically created rooms utilizing objects from ShapeNet. We follow the same train/val/test splits in both datasets with convention [1, 29, 44] for fair comparison. The points are randomly sampled, and Gaussian noise is applied. In addition, we adopt ScanNet-v2 [9], which contains 1,513 scene scans, for generality evaluation.

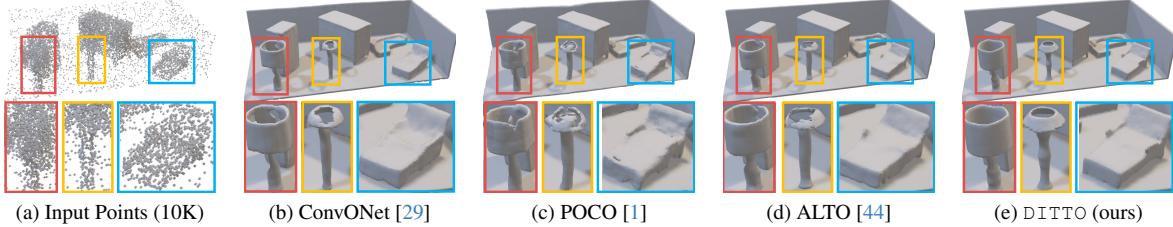| (a) Input Points (10K) | (b) ConvONet [29] | (c) POCO [1] | (d) ALTO [44] | (e) `DITTO` (ours) |

Figure 8. **Qualitative comparison of scene-level 3D surface reconstruction on the Synthetic Rooms dataset [29].**

Table 2. **Scene-level quantitative comparison on the Synthetic Rooms dataset [29].** We train each method for 10K input points with 0.005 noise level. The *triplane comparison* involves assessing the impact of triplane in methods using grid latents.

Method	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑
ONet [24]	0.475	2.03	0.783	0.541
SPSR [15]	-	2.23	0.866	0.810
SPSR trimmed [15]	-	0.69	0.890	0.892
ConvONet [29]	0.849	0.42	0.915	0.964
DP-ConvONet [19]	0.800	0.42	0.912	0.960
POCO [1]	0.884	0.36	0.919	0.980
ALTO [44]	0.914	0.35	0.921	0.981
Ours	**0.928**	**0.34**	**0.930**	**0.984**
Triplane comparison				
ConvONet [29]	0.805	0.44	0.903	0.948
ALTO [44]	0.895	0.37	0.910	0.974
Ours	**0.931**	**0.33**	**0.931**	**0.984**

Evaluation Metrics. We measure the reconstruction performance using standard quantitative metrics, such as IoU, Chamfer-L_1 distance, normal consistency (NC), and F-score [38], following baseline methods. For Chamfer-L_1 distance, we multiply 100 for convenience and use the threshold value as 1% for F-score.

4.2. Object-Level 3D Surface Reconstruction

Quantitative Evaluation. In Table 1, we quantitatively compare the object-level surface reconstruction performance on ShapeNet [4]. `DITTO` exhibits superior performance across all metrics. Notably, `DITTO` demonstrates a substantial lead in the IoU metric; a four times larger gap compared to the previous SoTA [44]. In addition, evaluation on various input point densities implies that `DITTO` shows robust scores regardless of the number of points, even though we use point latents. This result demonstrates the effectiveness and robustness of `DITTO`.

Qualitative Evaluation. Figure 7 shows the qualitative results. `DITTO` shows clear surface boundaries, especially for thin and intricate structures. Note that `DITTO` is the only method that successfully reconstructs the complex structures, such as the intricate pedestal of the lamp (first row), back of the bench (second row), rearview mirror and pattern on wheels of the car (third row). In particular, reconstructing shapes with repeated thin structures is a challenging

Table 3. **Scene-level quantitative comparison on the Synthetic Rooms dataset [29] with sparse and noisy input data.** We train each method for sparse (3K points with 0.005 noise level) and noisy (10K points with 0.025 noise level) input point clouds.

Method	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑
Sparse input points (3K input points)				
ConvONet [29]	0.818	0.46	0.906	0.943
POCO [1]	0.801	0.57	0.904	0.812
ALTO [44]	0.882	0.39	0.911	0.969
`DITTO` (ours)	**0.900**	**0.37**	**0.919**	**0.975**
Noisy input points (0.025 noise level)				
ConvONet [29]	0.777	0.57	0.872	0.885
POCO [1]	0.701	0.64	0.848	0.857
ALTO [44]	0.804	**0.55**	**0.877**	**0.898**
`DITTO` (ours)	**0.811**	**0.55**	0.875	**0.898**

problem. Grid latents struggle due to resolution constraints, while point latents often fail to create clear boundaries due to their inherent ambiguity. In this challenging case, `DITTO` successfully creates a clear surface boundary.

4.3. Scene-Level 3D Surface Reconstruction

Quantitative Evaluation. We assess the scene-level reconstruction performance on the Synthetic Rooms dataset [29]. The quantitative results are in Table 2. `DITTO` surpasses previous methods in most of the metrics. Regarding grid latent representation, `DITTO` marks a turning point. Most grid-based methods [29, 44] with triplane representations show decreased performance in complex scene-level reconstructions. In contrast, `DITTO` maintains consistent performance, even with triplane representations. In addition, quantitative results focusing on sparse and noisy inputs are in Table 3. `DITTO` shows outstanding performance robust to sparse and noisy input point clouds. These results contrast the method solely based on point latents, which is vulnerable to contamination of the input point clouds.

Qualitative Evaluation. We visualize the qualitative comparisons in Fig. 1 and Fig. 8. The results of ConvONet [29], a grid-based method, are relatively stable, with fewer holes in thin structure, but they lack detail (see lamps in red boxes). On the other hand, POCO [1], a point-based method, shows better detail but is less stable, often resulting in holes and artifacts (see lamps and chairs in yellow boxes). ALTO [44], employing both latents, tends to offer better detail compared to ConvONet and better stabil-
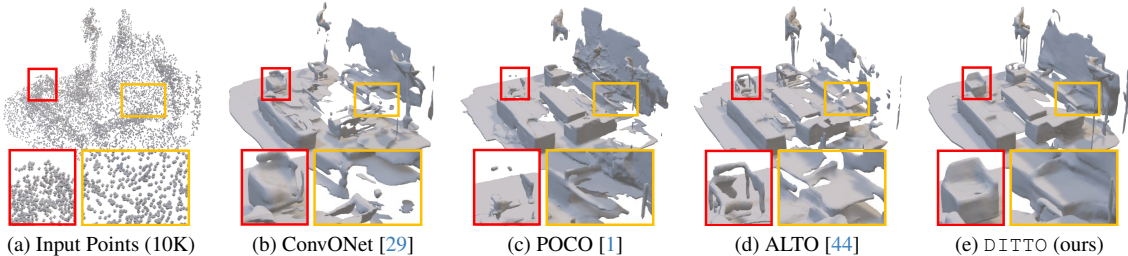
| (a) Input Points (10K) | (b) ConvONet [29] | (c) POCO [1] | (d) ALTO [44] | (e) DITTO (ours) |

Figure 9. **Qualitative comparison of ScanNet-v2 [29]**.

Table 4. **Ablation study on the proposed modules.** We train networks on the Synthetic Rooms dataset by switching modules one by one from the baseline to the suggested method.

Method	IoU ↑	Chamfer-L_1 ↓	NC ↑	F-score ↑
ALTO [44] (triplane; baseline)	0.895	0.35	0.921	0.981
+ DLL (DSPT backbone)	0.921	0.35	0.925	0.981
+ DLL (FKAConv [2] backbone)	0.917	0.35	0.922	0.979
+ DLL (PointTransformer [48] backbone)	0.911	0.36	0.917	0.976
+ IID	0.907	0.35	0.913	0.976
+ PointEncoder (FKAConv [2])	0.912	0.36	0.918	0.978
+ DLL + IID	0.925	0.34	0.927	0.983
+ DLL + PointEncoder	0.926	0.34	0.928	0.981
+ IID + PointEncoder	0.918	0.35	0.920	0.981
DITTO (ours)	**0.931**	**0.33**	**0.931**	**0.984**

ity than POCO. However, ALTO displays less stability than ConvONet and less detail than POCO. Unlike the previous approaches, DITTO validates both superior stability and detail. This result demonstrates that DITTO achieves synergy from the integration of two latents.

4.4. Ablation Study

We evaluate the impact of each proposed module by incrementally incorporating them into baseline ALTO (see Table 4). DLL demonstrates significant performance improvements regardless of the type of point backbones. This result reveals that the point feature extraction module that learns spatial patterns of point latents is more crucial than the simple MLP of ALTO. Additionally, while NC and F-score slightly decrease with IID alone, other metrics improve. However, combining DLL with IID significantly boosts performance across all metrics. We deduce that the variation in performance is due to the latent points without DLL not being optimized to learn spatial patterns.

4.5. Real-World 3D Surface Reconstruction

We conduct an additional experiment to assess generality (see Table 5 and Fig. 9). DITTO demonstrates superior performance than previous methods. DITTO successfully restores the sofas (see red and yellow boxes), where other methods encounter difficulties. We also evaluate the performance using triplanes as grid latents (see *Triplane comparison for grid latents*). Unlike the scene-level results in Sec. 4.3, triplanes generally show poorer performance on ScanNet-v2. We attribute this performance issue to a lack of geometric inductive bias in triplanes. Since each plane in

Table 5. **Quantitative comparison on ScanNet-v2 [9].** We test models pre-trained with the Synthetic Rooms dataset [29] on ScanNet-v2, which is not used during training. Since the dataset has no ground truth occupancy, we only measure Chamfer distance and F-score, following [44].

Method	$N_{train}, N_{test} = 10K, 10K$ Chamfer-L_1	F-score	$N_{train}, N_{test} = 10K, 3K$ Chamfer-L_1	F-score
ConvONet [29]	1.02	0.694	1.01	0.719
POCO [1]	0.87	0.757	0.93	0.737
ALTO [44]	<u>0.79</u>	<u>0.779</u>	<u>0.87</u>	<u>0.746</u>
DITTO (ours)	**0.70**	**0.808**	**0.78**	**0.773**
Triplane comparison				
ConvONet [29]	1.45	0.636	1.55	0.614
ALTO [44]	<u>1.43</u>	<u>0.640</u>	<u>1.44</u>	<u>0.601</u>
DITTO (ours)	**1.26**	**0.677**	**1.26**	**0.660**

a triplane misses information in a direction, triplane latents are required to learn spatial rules that the voxels naturally have. This limitation poses challenges for triplanes when reconstructing out-of-distribution data. Despite these challenges, DITTO shows better performance than other grid-based methods with triplanes.

5. Conclusion

We have proposed DITTO, a novel concept of dual and integrated latent topologies for implicit 3D reconstruction from noisy and sparse point clouds. Specifically, we have studied the use of grid and point latents together as dual latent to integrate their own strengths. To this end, we have proposed the DLL architecture with the DSPT module for enhancing dual latent while maintaining their original shape at the encoder level. Then, we explored how to utilize both refined latents in the proposed integrated implicit decoder. DITTO outperforms previous state-of-the-art implicit 3D reconstruction methods, especially DITTO facilitates the reconstruction of thin structures and intricate shape details.

References

[1] Alexandre Boulch and Renaud Marlet. Poco: Point convolution for surface reconstruction. In *CVPR*, 2022. 1, 2, 3, 5, 6, 7, 8

[2] Alexandre Boulch, Gilles Puy, and Renaud Marlet. Fkaconv: Feature-kernel alignment for point cloud convolution. In *ACCV*, 2020. 2, 3, 4, 8

[3] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022. 3

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 6, 7

[5] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, 2022. 3

[6] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 1, 2

[7] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *CVPR*, 2020. 1

[8] Gene Chou, Yuval Bahat, and Felix Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. In *ICCV*, 2023. 3

[9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 6, 8

[10] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *CVPR*, 2022. 4

[11] Zijian Dong, Xu Chen, Jinlong Yang, Michael J Black, Otmar Hilliges, and Andreas Geiger. Ag3d: Learning to generate 3d avatars from 2d image collections. In *CVPR*, 2023. 3

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022. 3

[13] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *NeurIPS*, 2022. 3

[14] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023. 3

[15] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *TOG*, 2013. 6, 7

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[17] Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, and Daniel Haziza. xformers: A modular and hackable transformer modelling library. https://github.com/facebookresearch/xformers, 2022. 6

[18] Yang Li and Tatsuya Harada. Lepard: Learning partial point cloud matching in rigid and deformable scenes. In *CVPR*, 2022. 4

[19] Stefan Lionar, Daniil Emtsev, Dusan Svilarkovic, and Songyou Peng. Dynamic plane convolutional occupancy networks. In *WACV*, 2021. 1, 2, 6, 7

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 4

[21] Zhijian Liu, Xinyu Yang, Haotian Tang, Shang Yang, and Song Han. Flatformer: Flattened window attention for efficient point cloud transformer. In *CVPR*, 2023. 4

[22] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 1987. 2

[23] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. 6

[24] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 1, 2, 3, 6, 7

[25] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *ICLR*, 2018. 6

[26] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *CVPR*, 2022. 3

[27] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 1, 2

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019. 6

[29] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[30] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. *NeurIPS*, 2021. 1, 3

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 4

[32] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, 2019. 1

[33] Jaehyeok Shim, Changwoo Kang, and Kyungdon Joo. Diffusion-based signed distance fields for 3d shape generation. In *CVPR*, 2023. 3

[34] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *CVPR*, 2023. 3

[35] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. 4, 5

[36] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022. 3

[37] Jiapeng Tang, Jiabao Lei, Dan Xu, Feiying Ma, Kui Jia, and Lei Zhang. Sa-convonet: Sign-agnostic optimization of convolutional occupancy networks. In *ICCV*, 2021. 1, 2, 3

[38] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *CVPR*, 2019. 7

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 3

[40] Rahul Venkatesh, Tejan Karmali, Sarthak Sharma, Aurobrata Ghosh, R Venkatesh Babu, László A Jeni, and Maneesh Singh. Deep implicit surface point prediction networks. In *ICCV*, 2021. 2, 3

[41] Haiyang Wang, Chen Shi, Shaoshuai Shi, Meng Lei, Sen Wang, Di He, Bernt Schiele, and Liwei Wang. Dsvt: Dynamic sparse voxel transformer with rotated sets. In *CVPR*, 2023. 4

[42] Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, et al. Rodin: A generative model for sculpting 3d digital avatars using diffusion. In *CVPR*, 2023. 3, 4

[43] Yizhi Wang, Zeyu Huang, Ariel Shamir, Hui Huang, Hao Zhang, and Ruizhen Hu. Aro-net: Learning implicit fields from anchored radial observations. In *CVPR*, 2023. 1, 2, 3

[44] Zhen Wang, Shijie Zhou, Jeong Joon Park, Despoina Paschalidou, Suya You, Gordon Wetzstein, Leonidas Guibas, and Achuta Kadambi. Alto: Alternating latent topologies for implicit 3d reconstruction. In *CVPR*, 2023. 1, 2, 3, 4, 5, 6, 7, 8

[45] Xingguang Yan, Liqiang Lin, Niloy J Mitra, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *CVPR*, 2022. 3

[46] Biao Zhang, Matthias Nießner, and Peter Wonka. 3dilg: Irregular latent grids for 3d generative modeling. *NeurIPS*, 2022. 3

[47] Biao Zhang, Jiapeng Tang, Matthias Nießner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *TOG*, 2023. 3

[48] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021. 5, 8