

MV-DeepSDF: Implicit Modeling with Multi-Sweep Point Clouds for 3D Vehicle Reconstruction in Autonomous Driving

Yibo Liu^{1,2}, Kelly Zhu^{1,3}, Guile Wu¹, Yuan Ren¹, Bingbing Liu¹, Yang Liu¹, Jinjun Shan²

¹Huawei Noah’s Ark Lab, ²York University, ³University of Toronto

{yorklyb, jjshan}@yorku.ca, kellyk.zhu@mail.utoronto.ca

{guile.wu, yuan.ren3, liu.bingbing, yang.liu9}@huawei.com

Abstract

Reconstructing 3D vehicles from noisy and sparse partial point clouds is of great significance to autonomous driving. Most existing 3D reconstruction methods cannot be directly applied to this problem because they are elaborately designed to deal with dense inputs with trivial noise. In this work, we propose a novel framework, dubbed MV-DeepSDF, which estimates the optimal Signed Distance Function (SDF) shape representation from multi-sweep point clouds to reconstruct vehicles in the wild. Although there have been some SDF-based implicit modeling methods, they only focus on single-view-based reconstruction, resulting in low fidelity. In contrast, we first analyze multi-sweep consistency and complementarity in the latent feature space and propose to transform the implicit space shape estimation problem into an element-to-set feature extraction problem. Then, we devise a new architecture to extract individual element-level representations and aggregate them to generate a set-level predicted latent code. This set-level latent code is an expression of the optimal 3D shape in the implicit space, and can be subsequently decoded to a continuous SDF of the vehicle. In this way, our approach learns consistent and complementary information among multi-sweeps for 3D vehicle reconstruction. We conduct thorough experiments on two real-world autonomous driving datasets (Waymo and KITTI) to demonstrate the superiority of our approach over state-of-the-art alternative methods both qualitatively and quantitatively.

1. Introduction

3D vehicle reconstruction from sparse and partial point clouds is a fundamental need in the autonomous driving industry [12, 17]. It aims to infer the 3D structure of vehicles at arbitrary resolutions in the wild, which is of great significance to many downstream tasks in autonomous driving. Despite the many 3D reconstruction methods

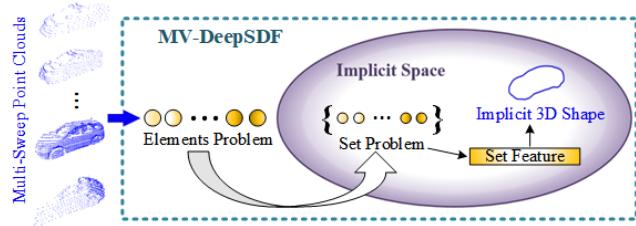


Figure 1. An illustration of the motivation of the proposed approach. Our approach takes multi-sweep point clouds as input and simplifies 3D vehicle reconstruction from multi-sweeps into an element-to-set feature extraction problem. In this way, we infer an optimal estimation of the 3D shape described in the abstract implicit space for 3D vehicle reconstruction in autonomous driving.

[28, 35, 26, 39, 41, 1, 19, 4], most of them focus on image-based inputs [28, 35, 26, 39, 41], dense point cloud inputs [19, 31, 11], or their combination [1, 4, 9] and thus, cannot be directly applied to this problem.

Recently, [12] has shown promising performance in applying implicit modeling to tackle this problem. Contrary to explicit modeling methods [32, 33, 47, 49, 43, 42, 10, 2, 38] that directly represent 3D object shape structure with points, voxels, or meshes, implicit modeling maps the 3D shape to a low-dimensional latent space and learns the projection from the latent space to a continuous function that describes the 3D shape. This presents two significant advantages: 1) the 3D shape can be stored as a low-dimensional memory-saving latent code [31, 11, 12, 6]; 2) the trained network outputs a continuous function in the 3D space which supports mesh extraction at any resolution [31]. To this end, we also choose to employ implicit modeling in our approach.

However, previous point cloud-based implicit modeling methods [31, 11, 12] mainly focus on recovering 3D shapes from a single-view partial point cloud and fail to leverage the multi-sweep information of vehicles. As a result, when noise or annotation errors exist in an individual sweep, single-view methods usually produce low fidelity results. In real-world datasets (e.g., [36, 15]), multi-sweep

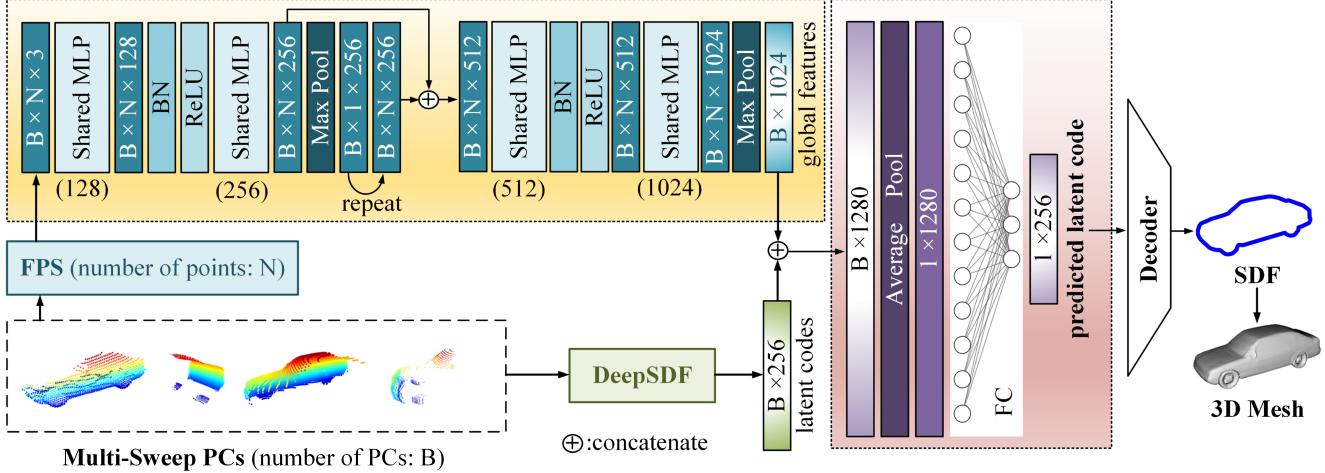


Figure 2. The framework of the proposed MV-DeepSDF. Farthest Point Sampling (FPS) [33] is applied to the raw point clouds for pre-processing and DeepSDF [31] is employed to generate a latent code for each observation. We then extract global features from the standardized point clouds (refer to yellow block) and concatenate the global features with the latent codes as *element-level representations*. Next, the element-level representations are transformed into a *set-level predicted latent code* (refer to red block). Finally, we employ a pre-trained DeepSDF decoder [31] to project the predicted latent code to the SDF of the vehicle in 3D space and recover the 3D mesh. Note that the decoder used in the final step (refer to white trapezoid) is identical to that of DeepSDF (refer to green block).

point clouds are usually available and contain richer shape information because of the various viewing angles offered by multiple observations [17]. Although there are some existing 3D reconstruction methods from multi-view point clouds [17, 19, 3, 34], implicit modeling with multi-sweep point clouds remains an unsolved problem.

In this work, we propose a novel framework, dubbed MV-DeepSDF, to exploit multi-sweep point clouds and implicitly generate high-fidelity 3D reconstructions of vehicles for autonomous driving. An illustration of the motivation for our proposed approach is depicted in Figure 1. Specifically, to shed light on the importance of exploring multi-sweep point clouds for 3D vehicle reconstruction, we first analyze multi-sweep consistency and complementarity in the latent feature space. We then propose to consider the problem of shape estimation in the implicit space as an element-to-set feature extraction problem, where a set represents a collection of multi-sweeps and an element represents a sparse and partial point cloud. Next, we devise a new architecture to simultaneously extract a global feature and latent code for each element in the multi-sweep and concatenate them as element-level representations. These element-level representations are then transformed into a set-level predicted latent code with the aid of average pooling and mapping. This set-level latent code is an optimal estimation of the 3D shape described in the abstract implicit space, which we subsequently decode to a continuous SDF of the vehicle with a pre-trained DeepSDF decoder [31] and recover the 3D mesh from the SDF. The **contributions** of this work are threefold:

- We analyze multi-sweep consistency and complemen-

tarity in the latent feature space and transform the problem of shape estimation in the implicit space into an element-to-set feature extraction problem. This simplifies 3D vehicle reconstruction from multi-sweep point clouds into an optimal estimation of the 3D shape described in the abstract implicit space.

- We propose a novel MV-DeepSDF framework (see Figure 2) for implicit modeling with multi-sweep point clouds. A new architecture is constructed to extract the element-level representations and generate the set-level predicted latent code.
- We qualitatively and quantitatively demonstrate the superior performance of our approach over the state-of-the-art alternative methods through extensive experiments on real-world datasets [36, 15].

2. Related Work

2.1. Explicit Modeling-Based 3D Reconstruction

Conventional 3D explicit modeling can be classified into three categories, namely point-based [32, 33, 47, 49, 43], voxel-based [42, 10], and mesh-based [2, 38]. Point-based methods, such as Point Completion Network (PCN) [49], GRNet [43], and PoinTr [47], output a point cloud with limited resolution since the number of points is fixed and not suitable for generating watertight surfaces due to lack of topology description. Voxel-based methods [42, 10] describe volumes by subdividing the space into a 3D grid, but they are memory and compute intensive, leading to slow

training and low resolution representations [31]. Mesh-based methods only generate meshes with simple topologies [40] given by a fixed reference template from the same object class and cannot guarantee watertight surfaces [22, 16]. In contrast, we propose implicit modeling with multi-sweep point clouds in a novel MV-DeepSDF framework, which estimates the optimal SDF shape representation for reconstructing 3D structures of vehicles in autonomous driving.

2.2. Implicit Modeling-Based 3D Reconstruction

Implicit modeling [27, 31, 11, 12] uses function-based decision boundaries to implicitly define surfaces for 3D representations. DeepSDF [31] is a classical approach to implicit modeling. It estimates the Signed Distance Function (SDF) of an object from a partial point cloud, where the SDF specifies whether a querying position is inside or outside the surface of the object and the distance of the querying position from the surface. There have been some improved versions [11, 12, 6, 45] of DeepSDF, but these methods only focus on 3D reconstruction from a single-view point cloud and thus, generally produce low fidelity results when noise or annotation errors exist in an individual sweep. In contrast, we propose a novel framework for implicit modeling with multi-sweep point clouds. We analyze multi-sweep consistency and complementarity in the latent feature space and propose to resolve this problem by an optimal estimation of the 3D shape described in the abstract implicit space. Moreover, other implicit modeling-based methods, such as Neural Radiance Fields (NeRF) [28], Point-NeRF [44], Direct Voxel Grid Optimization [35], NeuS [41], NEAT [26], and [1, 4, 9] exist, but they all require image input and cannot deal with only LiDAR data.

2.3. Multi-View 3D Reconstruction

3D reconstruction from multi-view point clouds can be categorized into conventional approaches [3, 34, 29] and deep learning-based approaches [17, 19, 5]. Conventional geometric approaches, such as Iterative Closest Point (ICP), TSDF [29], and [3, 34], only leverage multi-view observations geometrically, so the resulting completion is merely an aligned stack of partial point clouds and cannot produce watertight shapes unless the sensor fully loops around the object. In deep learning-based approaches, [17] proposes a weakly-supervised framework to directly learn 3D shape completion from multi-view LiDAR sweeps, but its reconstruction result is not watertight. [19] presents a shape completion framework using a multi-view depth map-based shape representation approach, while [5] designs a network to use multiple partial point clouds encoded into the latent space to estimate the optimal shape of the object. However, the output of these networks [19, 5] are point-based repre-

sentations and in contrast with continuous field functions such as SDFs [31], lack continuity in 3D space. Moreover, some methods, such as [13, 14, 46], exist for reconstructing vehicle shapes from multiple observations, but [13] relies on stereo images and motion priors to regularize the shape estimation of vehicles, [14] requires multi-view camera images to build semantic maps which contain 3D vehicle shapes, and [46] requires inputs to follow a time sequence to track and reconstruct 3D objects. In contrast, our approach simplifies 3D vehicle reconstruction from multi-sweep point clouds into an optimal estimation of the 3D shape described in the abstract implicit space, which supports mesh extraction at any resolution and can deal with noise and annotation errors that exist in an individual sweep.

3. Methodology

3.1. Preliminaries

In this section, we introduce preliminaries regarding DeepSDF [31] since our approach utilizes its framework for extracting the latent code and decoding the continuous SDF for 3D mesh generation. Formally, when adopting the DeepSDF decoder into the shape completion task, points of partial point clouds are taken as given surface points and the SDF value is defined as the distance from the querying point to the surface of the object. Interior and exterior querying points are sampled along the normals of the surface points and the SDF value is computed for each querying point. Suppose the set of newly sampled interior and exterior points is denoted as \mathcal{P}_M , where M is the number of points. Define the m th point in \mathcal{P}_M as \mathbf{x}_m and its corresponding SDF value as s_m . In the domain of DeepSDF [31], a 3D shape can be represented by a latent code $\mathbf{z} \in \mathbb{R}^{256}$ in the latent space. The complete shape is obtained by optimizing \mathbf{z} via Maximum a Posterior estimation:

$$\hat{\mathbf{z}} = \frac{1}{\sigma^2} \|\mathbf{z}\|_2^2 + \arg \min_{\mathbf{z}} \sum_{m=1}^M \mathcal{L}_1(f_{\theta}(\mathbf{z}, \mathbf{x}_m), s_m), \quad (1)$$

where \mathcal{L}_1 represents the clamped \mathcal{L}_1 distance and σ is a regularization hyperparameter. Once the optimal latent code $\hat{\mathbf{z}}$ is determined, the complete 3D shape can be recovered by finding the zero-SDF-value isosurface through the DeepSDF decoder $f(\cdot)$, which is defined as:

$$\hat{s} = f_{\theta}(\hat{\mathbf{z}}, \mathbf{x}), \quad (2)$$

where θ represents the parameters of the decoder, $\mathbf{x} \in \mathbb{R}^3$ denotes the 3D coordinates of the querying point, and \hat{s} denotes the estimated SDF value given by $f(\cdot)$. Here, the sign of \hat{s} , either positive or negative, indicates whether the point lies on the exterior or interior of the surface, respectively. Thus, the surface of the object can be implicitly represented by the isosurface composed of points with zero SDF values.

Despite its simplicity and efficiency, DeepSDF only takes the given surface points into consideration when generating 3D shapes (*i.e.*, Eq. (1)). This can lead to suboptimal reconstruction results in areas not captured by the partial point clouds since DeepSDF will reconstruct arbitrarily based on prior knowledge learned during training. For instance, referencing the ground truth of Case 1 in Figure 3, the reconstruction results of DeepSDF for Cases 2, 3, and 4 only demonstrate high fidelity in areas with well-captured surface points. The missing areas are reconstructed arbitrarily. To resolve this problem, we analyze multi-sweep consistency and complementarity in the latent feature space and propose implicit modeling with multi-sweep point clouds in MV-DeepSDF.

3.2. Approach Overview

As shown in Figure 2, there are three main steps in MV-DeepSDF, namely preprocessing, optimal latent code prediction, and 3D mesh extraction.

1) Preprocessing. First, we carry out Farthest Point Sampling (FPS) [33] to sample a fixed number of points on each raw partial point cloud since our global feature extractor requires standardized point clouds as input. For FPS, the number of centroids is set as 256.

2) Optimal Latent Code Prediction. The post-FPS point clouds are used as input into the global feature extractor (yellow block in Figure 2) to extract global features for each individual point cloud. Meanwhile, the original point clouds are used to extract latent codes for each partial point cloud through the pre-trained DeepSDF (green block in Figure 2). Then, as shown in the red block of Figure 2, the global features and latent codes are concatenated as element-level representations, followed by an average pooling operation to aggregate the information into a single instance tensor. Finally, this instance tensor is transformed into the predicted latent code by a fully-connected layer as the set-level representation.

3) 3D Mesh Extraction. The predicted latent code is converted to an SDF using a pre-trained DeepSDF decoder [31]. The isosurface composed of querying points with zero SDF values represents the surface of the instance and the implicit surface can be rasterized to a 3D mesh using Marching Cubes [24]. Since the SDF is a continuous function, it supports 3D mesh extraction at any resolution.

3.3. Consistency and Complementarity Analysis

Background. In literature [37, 50], multi-view consistency refers to the consistency of describing the same instance from different viewpoint observations and multi-view complementarity refers to the complementary information of the same instance provided by different observations. Existing methods mainly explore multi-view consistency and complementarity for 3D reconstruction with par-

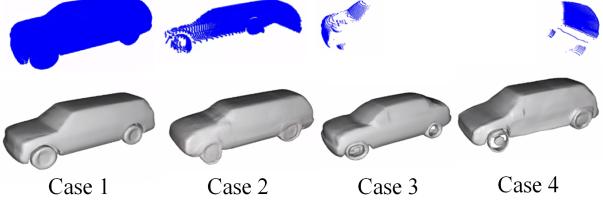


Figure 3. The 3D reconstruction results of DeepSDF [31] (row 2) using different point clouds (row 1). All point clouds are sampled from the same CAD model of ShapeNetV2 [7]. Case 1 shows the complete dense point cloud used to train the DeepSDF decoder, while Cases 2, 3, and 4 show the partial point clouds.

tial point clouds in explicit modeling [34, 3].

However, existing methods are not applicable for exploring multi-sweep-/view consistency and complementarity in implicit modeling because the abstract feature space operates differently than the explicit space. In implicit modeling, point clouds are expressed as low-dimensional feature vectors (a.k.a. latent vectors) in the latent feature space and the latent vectors are generated through a non-linear learning process that projects the 3D space to the feature space through a deep neural network [31, 32, 33, 49].

Theoretical Analysis. Despite these differences, we can still explore multi-sweep consistency and complementarity in the latent feature space for implicit modeling. Inspired by [25], consider a total of B observations. The latent code for the i th observation, denoted by \mathbf{z}_i , can be represented as:

$$\mathbf{z}_i = \mathbf{z}_{i,c} + \mathbf{z}_{i,s} + \mathbf{e}_i, \quad (3)$$

where $i = \{1, \dots, B\}$, $\mathbf{z}_{i,c}$ is the consistent component, $\mathbf{z}_{i,s}$ is the specific component, and \mathbf{e}_i is the error component. In this work, $\mathbf{z}_{i,c}$ and $\mathbf{z}_{i,s}$ can be defined as follows:

$$\mathbf{z}_{i,c} = \mathbf{z}_i \cap \mathbf{z}_{gt} \quad \text{and} \quad (\mathbf{z}_{i,s} + \mathbf{e}_i) = \mathbf{z}_{gt} - \mathbf{z}_i, \quad (4)$$

where \mathbf{z}_{gt} is the latent code corresponding to the ground truth shape and ‘ \cap ’ and ‘ $-$ ’ represent the similarity and difference in information captured by the two feature vectors, respectively. Now, consider any two consistent components $\mathbf{z}_{i,c}$ and $\mathbf{z}_{j,c}$. In the first case where $\mathbf{z}_{i,c} \cap \mathbf{z}_{j,c} = \emptyset$, the two components contain completely different consistent information, which is complementary. In the second case where $\mathbf{z}_{i,c} \cap \mathbf{z}_{j,c} \neq \emptyset$, but $\mathbf{z}_{i,c} \not\subseteq \mathbf{z}_{j,c}$ and $\mathbf{z}_{j,c} \not\subseteq \mathbf{z}_{i,c}$, the two components contain some different information, which can be aggregated as complementary information to approach the ground truth. In the final case where $\mathbf{z}_{i,c} \cap \mathbf{z}_{j,c} \neq \emptyset$, but $\mathbf{z}_{i,c} \subseteq \mathbf{z}_{j,c}$ or $\mathbf{z}_{j,c} \subseteq \mathbf{z}_{i,c}$, either $\mathbf{z}_{i,c}$ or $\mathbf{z}_{j,c}$ contains redundant information from the other component. While redundant, this will not adversely impact complementary aggregation. Thus, to make the optimal $\hat{\mathbf{z}}$ approach the ground

truth \mathbf{z}_{gt} , it is desired to aggregate all consistent and complementary information among all individual features:

$$\hat{\mathbf{z}} = \mathbf{z}_{1,c} \cup \mathbf{z}_{2,c} \dots \cup \mathbf{z}_{B,c} + \mathbf{p}, \quad (5)$$

where ‘ \cup ’ refers to the complementary aggregation of information among features and \mathbf{p} denotes the predicted information for regions not captured in the multi-sweeps (*e.g.*, the side of the vehicle not captured by the multi-sweeps as shown in the Cases 3 and 4 of Figure 3). In summary, with implicit modeling, we can learn the optimal latent code $\hat{\mathbf{z}}$ by aggregating consistent (Eqs. (3)-(4)) and complementary (Eq. (5)) information among multi-sweeps.

3.4. Architecture Design of MV-DeepSDF

To take advantage of multi-sweep consistency and complementarity in the latent feature space, we design a new architecture for implicit modeling. Our proposed architecture uses multi-sweep point clouds as model input and generates an optimal estimation of the 3D shape in the implicit space (*a.k.a.* the optimal latent code) as the output. The functionality of the network can be formulated as:

$$g_\alpha(\mathbf{Z}, \mathbf{P}) = \hat{\mathbf{z}} \rightarrow \mathbf{z}_{gt}, \quad (6)$$

where $g(\cdot)$ denotes the desired network with associated parameters α , $\mathbf{Z}=\{\mathbf{z}_1, \dots, \mathbf{z}_B\}$ denotes the set of latent codes, B represents the number of observations or latent codes, $\mathbf{P}=\{\mathcal{P}_1, \dots, \mathcal{P}_B\}$ denotes the set of multi-view point clouds, and $\hat{\mathbf{z}} \in \mathbb{R}^{256}$ and $\mathbf{z}_{gt} \in \mathbb{R}^{256}$ represent the estimated optimal and ground truth latent codes, respectively. Hence, the goal is converted into learning a network to make $\hat{\mathbf{z}}$ approach \mathbf{z}_{gt} for subsequent reconstructions. To this end, we devise a new architecture to realize the functionality required by simultaneously learning global features and latent codes for generating predicted latent codes.

Specifically, inspired by PointNet [32], a pioneering work for 3D classification and segmentation, we transform the problem of shape estimation in the implicit space into an element-to-set feature extraction problem. Given a point cloud containing a set of 3D points, where all 3D points consistently describe a single object since they all belong to the same instance, we can learn a global feature for each point cloud that aggregates complementary features from each individual 3D point. However, only learning a global feature cannot be used for 3D reconstruction, so we propose to abstract the pipeline of PointNet (see Figure 4(a)) into a generalized process (see Figure 4(b)). As shown in Figure 4(a), PointNet first computes 3D point features using a shared MLP on each 3D point and then transforms these point features into a single global feature through max pooling. Now, suppose there exists a set composed of multiple elements. The goal of the generalized pipeline is to extract a feature for the entire set by aggregating all elements. To this

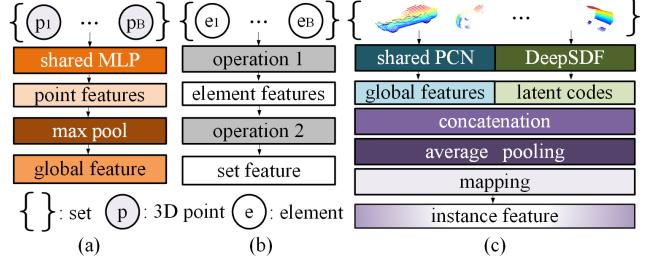


Figure 4. Comparison of three pipelines. (a) The pipeline of PointNet [32]. (b) The pipeline of an abstract generalized process to extract the set feature from the elements. (c) The proposed pipeline to extract the instance feature from multi-sweep point clouds.

end, we propose two generalized operations. As shown in Figure 4(b), we define operation 1 as the computation of the element features (*e.g.*, the shared MLP in PointNet) and operation 2 as the transformation of the element features into a set feature (*e.g.*, the max pooling operation in PointNet).

Using this generalized process, we can now devise our architecture to simultaneously learn global features and latent codes for generating predicted latent codes. First, we construct a global feature extractor (see yellow block in Figure 2) for learning a global feature for each point cloud that aggregates complementary features from each individual 3D point. The architecture of this feature extractor is inspired by the PCN encoder [49], a variant of PointNet [32]. This block is a stack of four PointNet [32] encoders with 128, 256, 512, and 1024 units, respectively. Since this architecture is inspired by the PCN encoder [49] and its global feature extraction operation is applied to every individual point cloud, we name it *shared PCN*. Meanwhile, to exploit the implicit information of each element, we employ a pre-trained DeepSDF [31] model (as denoted by Eq. (1)) to generate a latent code for each partial point cloud. Next, we concatenate the global features and latent codes to generate the element-level representations. Then, we employ average pooling and mapping to aggregate the element-level representations into a set-level predicted latent code. As shown in Figure 4(c), operation 1 of our pipeline consists of a global feature extractor and a pre-trained DeepSDF model, while operation 2 consists of concatenation, average pooling, and mapping. This pipeline realizes a symmetric operation on unordered multi-sweep point clouds.

3.5. Model Training of MV-DeepSDF

We adopt a curriculum learning strategy for model training, which consists of two stages.

Stage One. In stage one, we pre-train the DeepSDF decoder using watertight CAD models belonging to the car taxonomy of the ShapeNetV2 dataset [7]. The latent code \mathbf{z}

and parameters of the decoder θ are jointly optimized as:

$$\arg \min _{\theta,\{\mathbf{z}_j\}_{j=1}^J} \sum_{j=1}^J\left(\frac{1}{\sigma^2}\|\mathbf{z}_j\|_2^2+\sum_{k=1}^K \mathcal{L}_1\left(f_{\theta}\left(\mathbf{z}_j, \mathbf{x}_k\right), s_k\right)\right), \quad(7)$$

where J represents the number of 3D shapes used for training and K represents the number of points for each shape. During this stage, the decoder gains prior knowledge of vehicle shapes and once trained, the decoder remains fixed for the latent code generation step of our pipeline. Please refer to [31] for the details of the training at this step.

Training Dataset Preparation for Stage Two. As shown in Eq. (6), ground truth latent codes, partial point clouds, and their corresponding latent codes are required for training the reconstruction network. However, real-world datasets, such as Waymo [36] and KITTI [15], do not contain ground truth shapes, so it is necessary to use a synthetic dataset [7] for generating the training data. To resolve this problem, we utilize the latent codes of the training shapes (*e.g.*, the complete dense point cloud shown in Case 1 of Figure 3) as ground truth latent codes in the second stage of model training. As for the partial point clouds, the domain gap between training instances and in-the-wild instances directly determines the ability of our network to generalize to real-world datasets. To reduce the domain gap, we adopt PCGen [23] as our partial point cloud generation method. PCGen places a virtual LiDAR with real-world parameters (resolution and sampling pattern) around the vehicle and simulates the point cloud captured by the virtual LiDAR. This differs from the method used by DeepSDF [31], where a simulated depth camera is used as the virtual sensor instead of a LiDAR. A visual comparison of the real-world point cloud and simulated partial point clouds obtained using various methods is given in Figure 5. As shown in Figure 5, the partial point clouds of PCGen [23] are visually closer to the in-the-wild LiDAR sweeps of Waymo [36] and KITTI [15] than those of the virtual depth camera approach [31], which only raycasts evenly in the inclination and azimuth directions. Apart from this visual superiority, the advantages of generating partial point clouds with PCGen can also be observed through the improved generalization ability of our network. See experiments for further details.

When generating training data from ShapeNetV2 [7], we randomly sample one side of the vehicle to accurately reflect the real-world scan of a LiDAR on an ego-vehicle. In this way, we produce six partial point clouds for each vehicle. The virtual LiDAR pose for each partial point cloud is generated using a set of restrictions. These restrictions are set by considering the possible relative poses of an ego vehicle’s LiDAR in the coordinate system of the other vehicle. In particular, $\theta \in [0^\circ, 180^\circ]$ or $\theta \in [-180^\circ, 0^\circ]$, $r \in [3, 15]$, and $h \in [0.8, 1.2]$ are used, where θ represents the azimuth,

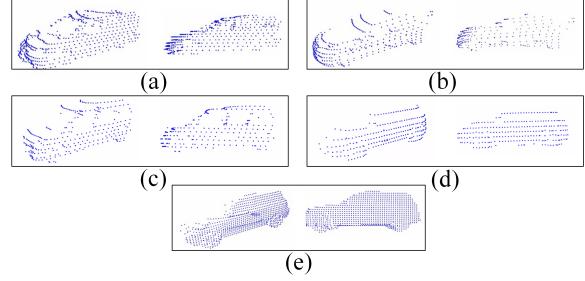


Figure 5. Visual comparison of in-the-wild partial point clouds and raycasted point clouds. (a) A partial point cloud from the Waymo tracking dataset [36]. (b) The raycasted point cloud generated by PCGen [23] using the LiDAR parameters of Waymo. (c) A partial point cloud from the KITTI tracking dataset [15]. (d) The raycasted point cloud generated by PCGen [23] using the LiDAR parameters of KITTI. (e) The raycasted point cloud obtained from the virtual depth camera approach used by DeepSDF [31].

r represents the distance between the LiDAR and the vehicle, and h represents the height of the LiDAR to the ground. θ , r , and h describes the pose of the virtual LiDAR with respect to the frame of the other vehicle. Both r and h are expressed in the normalized space, where the size of the vehicle is normalized into the range $[-1, 1]$. Finally, we generate a latent code for each partial point cloud using Eq. (1).

Stage Two. In stage two, the network is trained to reconstruct 3D vehicles from multi-view partial point clouds and their corresponding latent codes. Consider the c th vehicle instance. Let $\mathbf{P}_{B,c}$ represent the set of partial point clouds, $\mathbf{Z}_{B,c}$ represent the set of latent codes corresponding to the multi-view point clouds, and $\mathbf{z}_{gt,c}$ represent the ground truth latent code acquired from stage one. Furthermore, let $g(\cdot)$ represent the function of the implicit shape prediction network (refer to yellow and red blocks in Figure 2) and α represent the parameters of the model. The training objective of stage two can be defined as:

$$\arg \min _{\alpha} \sum_{c=1}^C \mathcal{L}_2\left(g_{\alpha}\left(\mathbf{Z}_{B,c}, \mathbf{P}_{B,c}\right), \mathbf{z}_{gt,c}\right), \quad(8)$$

where C represents the number of instances employed for training and \mathcal{L}_2 is the Mean Squared Error loss. The training details are presented in Sec. 4.1.

4. Experiments

In this section, we present qualitative and quantitative results on two real-world autonomous driving datasets, namely Waymo [36] and KITTI [15]. The multi-sweep point clouds for Waymo are collected from 136 unique vehicle instances of Waymo Open Dataset’s tracking data [36], while those for KITTI are extracted from 233 vehicle in-

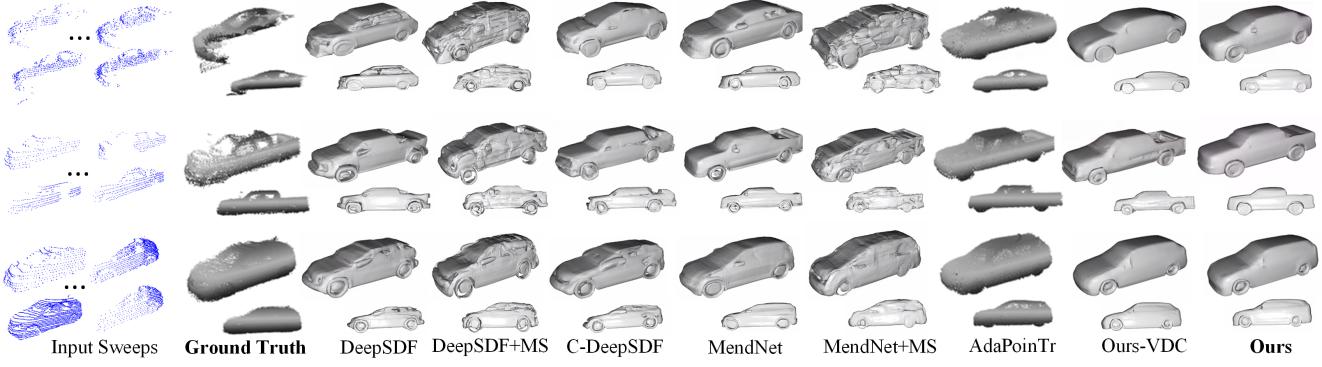


Figure 6. Visual comparison with the state-of-the-art methods (DeepSDF [31], C-DeepSDF [11], MendNet [12], and AdaPoinTr [48]) on the Waymo [36] dataset. DeepSDF+MS and MendNet+MS indicate the models with multi-sweep input.

stances of KITTI’s tracking dataset [15]. Due to page limitations, experiment details and some results (*e.g.*, the results on the synthetic dataset ShapeNetV2 [7]) are discussed in the supplementary material.

4.1. Implementation Details

Architecture Details. Our global feature extractor, named *shared PCN*, is a variant of the PCN encoder [49]. The embedded shared MLP layers are identical to those of PointNet [32], which are implemented using 1D convolution layers. Moreover, since DeepSDF [31] normalizes the latent codes into the range $[-1, 1]$, we add a tanh layer after the last shared MLP to normalize the global features into the same range as the latent codes. The decoder is identical to that of DeepSDF [31]. We report results based on an implementation with Python and PyTorch, but our approach also supports implementation with MindSpore [20].

Training Details. In the first stage of training, we follow the same method as presented in DeepSDF [31] to train the decoder using watertight CAD models from the car taxonomy of ShapeNetV2 [7]. Once trained, the decoder is fixed and projects a 256-dimensional latent code to an SDF in 3D space. In the second stage, we train the model for 20 epochs using the Adam optimizer [21] with a learning rate of 1e-5. The batch size is set to 1, allowing the model to simultaneously see all 6 frames of the given instance at each iteration. Since both the output of the model and the supervision are 1D vectors, the loss computation is very fast. It only takes 10 minutes to train our network on a single NVIDIA GeForce RTX 2080 GPU during stage two. However, the DeepSDF decoder [31] used to prepare the training dataset for stage two through latent code generation is a more time-consuming process.

4.2. Evaluation Protocol

Metrics. Since an in-the-wild instance does not have a ground truth 3D shape, we follow [12] to evaluate the recon-

struction results. In particular, the multiple LiDAR sweeps are stacked to construct the ground truth points set \mathbf{X} . Moreover, we sample 30,000 points on the surface of each reconstructed mesh to generate the reconstruction points set \mathbf{Y} . We employ Asymmetric Chamfer Distance (ACD) [12], which is the sum of the squared distance of each ground truth point to the nearest point in the reconstructed point set, to evaluate reconstruction results on real-world datasets:

$$\text{ACD}(\mathbf{X}, \mathbf{Y}) = \sum_{x \in \mathbf{X}} \min_{y \in \mathbf{Y}} \|x - y\|_2^2. \quad (9)$$

For ACD, a smaller value is preferred. In addition, we compute the recall of the ground truth points from the reconstructed shape, which is defined as:

$$\text{Recall}(\mathbf{X}, \mathbf{Y}) = \frac{1}{|\mathbf{X}|} \sum_{x \in \mathbf{X}} \left[\min_{y \in \mathbf{Y}} \|x - y\|_2^2 \leq t \right], \quad (10)$$

where the threshold t is set as 0.1, following [12].

Competitors. We compare our approach with four state-of-the-art methods, including three implicit modeling-based methods (**DeepSDF** [31], **C-DeepSDF** [11], and **MendNet** [12]) and **AdaPoinTr** [48], which is built upon PoinTr++ [47], the winner of Multi-View Partial Point Cloud Challenge 2021 on Completion and Registration [30]. Training data for DeepSDF and C-DeepSDF are generated following the original papers, while MendNet uses PCGen [23] to generate training data. The pipeline of AdaPoinTr [48] requires online generation of partial point clouds using the virtual depth camera approach and since it is not trivial to replace this process with PCGen, AdaPoinTr is trained using the default setup without PCGen. Note that with multi-sweep point clouds, competitors generate multiple reconstruction results (one for each partial point cloud), so we compare with the best single-shot reconstruction result, which is the mesh with the minimum ACD among the multiple single-shot reconstructed meshes. Furthermore, we report the results of DeepSDF and MendNet with multi-sweep

Method \ Metric	ACD _{mean} ↓	ACD _{median} ↓	Recall ↑
DeepSDF [31]	6.26	5.81	93.51
DeepSDF+MS	5.12	5.09	95.57
C-DeepSDF [11]	6.21	5.64	93.98
MendNet [12]	4.92	4.79	95.39
MendNet+MS	4.85	4.77	95.76
AdaPoinTr* [48]	4.79	4.74	95.95
Ours-VDC	4.76	4.55	96.05
Ours	3.36	2.26	96.84

Table 1. Comparison with the state-of-the-art methods on Waymo. ACD ↓ is multiplied by 10^3 . Recall ↑ is presented as a percentage. * AdaPoinTr is trained with the default setup without PCGen on the large-scale ShapeNet-55 [47] following the original paper [48], and then fine-tuned on the car taxonomy of ShapeNetV2 in our experiments. Other methods are only trained with the car taxonomy of ShapeNetV2.

inputs (denoted as **DeepSDF+MS** and **MendNet+MS**), in which we stack the partial point clouds of multi-sweeps into a single point cloud and perform single-shot-based reconstruction on this stacked point cloud. For our approach, in addition to the default setup which generates the training dataset using PCGen (denoted as **Ours**), we also adapt our approach to use training data generated by the virtual depth camera approach [31] (denoted as **Ours-VDC**).

4.3. Results on Waymo

The qualitative and quantitative comparisons of our approach with the state-of-the-art methods are presented in Figure 6 and Table 1, respectively. Overall, our approach achieves the best reconstruction results both qualitatively and quantitatively. Specifically, from Figure 6 and Table 1, we can see that DeepSDF [31] and C-DeepSDF [11] are sensitive to noise, resulting in low fidelity. MendNet [12] generates more stable results than DeepSDF and C-DeepSDF, but its reconstruction meshes contain some irregular holes on the surface (see sixth column of Figure 6). Although AdaPoinTr [48] outputs point clouds with good fidelity, it expresses the shape with a limited resolution and thus fails to describe continuous local details (see seventh column of Figure 6).

Compared with vanilla DeepSDF and MendNet, DeepSDF+MS and MendNet+MS yield messier surfaces/structures (see fourth and eighth columns of Figure 6), despite their better metric scores. This shows that even though transforming all partial point clouds into a single frame does benefit information aggregation, it also accumulates noise and annotation errors in the process. Hence, due to noise in real-world data, it is not practical to geometrically stack multi-sweep point clouds and directly perform single-shot-based reconstruction. In comparison, our approach generates smooth watertight shapes (see last two columns of Figure 6) with high fidelity (see Table 1). Furthermore, Ours outperforms Ours-VDC, which indicates

Method \ Metric	ACD _{mean} ↓	ACD _{median} ↓	Recall ↑
DeepSDF [31]	6.81	6.17	80.65
DeepSDF+MS	6.11	5.83	82.73
C-DeepSDF [11]	6.73	5.99	80.77
MendNet [12]	5.94	5.64	83.84
MendNet+MS	5.83	5.61	84.26
AdaPoinTr [48]	5.89	5.67	84.20
Ours-VDC	5.75	5.24	84.39
Ours	4.27	3.01	85.88

Table 2. Comparison with the state-of-the-art methods on KITTI. ACD ↓ is multiplied by 10^3 . Recall ↑ is presented as a percentage.

that using PCGen to prepare training partial point clouds (refer to Figure 5) reduces the domain gap between training and in-the-wild instances.

4.4. Results on KITTI

The qualitative and quantitative comparisons of our approach against the state-of-the-art methods are presented in Figure 7 and Table 2, respectively. From these results, we can also observe the superior performance of our approach over the state-of-the-art competitors. Specifically, qualitatively, as shown in Figure 7, our approaches (Ours and Ours-VDC) are more robust to noise when compared to DeepSDF [31], DeepSDF+MS, and C-DeepSDF [11] and generate smoother, watertight surfaces compared to MendNet [12], AdaPoinTr [48], and MendNet+MS. Quantitatively, as shown in Table 2, Ours yields the best ACD and Recall results, while Ours-VDC performs the second best.

4.5. Ablation Study

To verify the effectiveness of the main components of our approach, we carry out a series of experiments on Waymo [36] as shown in Table 3. The first row of Table 3 refers to the baseline model of our approach, which is a shared PCN encoder (Enc.) (yellow block in Figure 2) followed by an average pooling layer (Avg.). This architecture is similar to that of MendNet [12], which adds two stacked PointNet encoders [32] to the DeepSDF decoder. In the second row, we directly use bitwise multiplication to merge the global features ($B \times 256$, instead of $B \times 1024$) and the latent codes ($B \times 256$). However, this results in obvious performance degradation, which shows that directly merging the global features and latent codes obtained from multi-sweep information does not bring improvement to our baseline model. In comparison, in the third row, we add our proposed components, namely the input of latent codes (Dep.), concatenation (Con.), and mapping (Map.), to the baseline model, but with max pooling. This brings significant performance improvement compared to the first and second rows. Then, in the last row, we further replace max pooling with average pooling, which includes all proposed components of this work. This yields the best results.

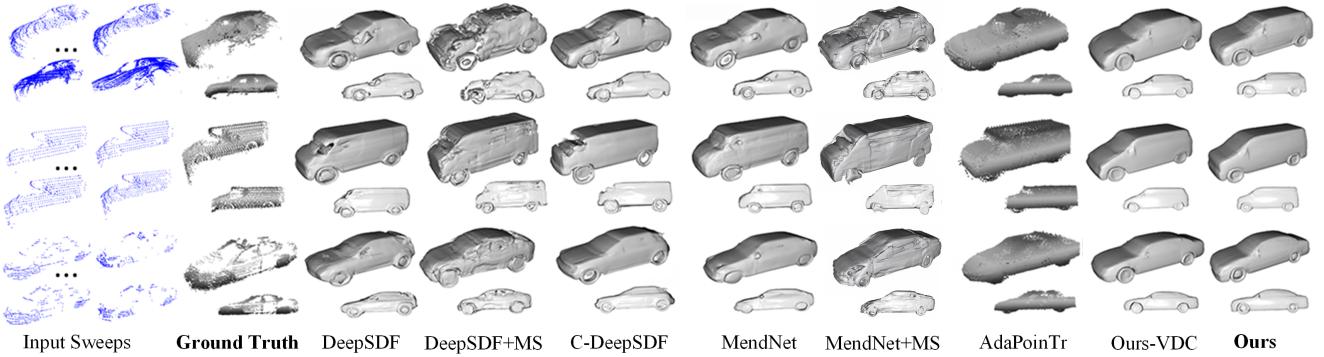


Figure 7. Visual comparison with the state-of-the-art methods (DeepSDF [31], C-DeepSDF [11], MendNet [12], and AdaPoinTr [48]) on the KITTI [15] dataset. DeepSDF+MS and MendNet+MS indicate the models with multi-sweep input.

Operation 1			Operation 2			Metric	
Enc.	Dep.	Mer.	Pool	Map.		ACD \downarrow	Recall \uparrow
✓			Avg.			4.89	95.50
✓	✓	Mul.	Avg.			7.33	84.93
✓	✓	Con.	Max.	✓		4.35	96.61
✓	✓	Con.	Avg.	✓		3.36	96.84

Table 3. The ablation study of the proposed framework. Refer to Figure 4(b) to see the definitions of operations 1 and 2. Enc.: the shared PCN encoder; Dep.: latent code generation through DeepSDF as input; Mer.: the operation to merge global features and latent codes; Map.: the mapping conducted by the fully-connected layer; Con.: the concatenation; Mul.: the bitwise multiplication; Avg.: average pooling; Max.: max pooling. ACD \downarrow is multiplied by 10^3 . Recall \uparrow is presented as a percentage.

4.6. Extension to Other Taxonomies

It is straightforward to extend the proposed shape completion framework to other taxonomies. Suppose that we want to extend MV-DeepSDF to perform shape completion on sofas. First, a new DeepSDF decoder [31] needs to be pre-trained using watertight CAD models from the sofa taxonomy of ShapeNetV2 [7] or another synthetic dataset. Then, PCGen [23] or another sampling technique (*e.g.*, the approaches proposed in [31, 47]) can be used to generate partial point clouds for each sofa instance. These partial point clouds are then passed into the pre-trained DeepSDF decoder [31] to produce their corresponding latent codes. Finally, the partial point clouds and latent codes can be used together to train MV-DeepSDF. Following the pipeline above, we evaluate on indoor sofas of ShapeNetV2 and outdoor cyclists of Waymo and show that our method is versatile and extensible to other taxonomies (see Figure 8).

While existing datasets, such as ScanNet [8] and Semantic3D [18], provide real-world point cloud data for many taxonomies, the availability of tracking data for multi-sweep scans is still fairly limited to the autonomous driving industry. Hence, it is necessary to first obtain the corresponding multi-sweep tracking labels for the desired class



Figure 8. Visual comparison of DeepSDF and MV-DeepSDF on the indoor sofa of ShapeNetV2 and the outdoor cyclist of Waymo.

when choosing to adopt MV-DeepSDF on a new taxonomy.

5. Conclusion

In this work, we propose a new MV-DeepSDF framework to facilitate implicit modeling with multi-sweep point clouds for autonomous driving. The main idea is to abstract 3D vehicle reconstruction from multi-sweeps into an element-to-set feature extraction problem. Namely, we consider the multi-sweeps of a vehicle as elements composing a set and infer the set feature, which is an optimal estimation of the 3D shape described in the abstract implicit space. In particular, we present a theoretical analysis of multi-sweep consistency and complementarity in the latent feature space. Guided by this analysis, we design a new architecture to optimally estimate the Signed Distance Function shape of a vehicle from its in-the-wild multi-sweep point clouds. Qualitative and quantitative evaluations on both real-world and synthetic datasets show the superiority of our approach over the state-of-the-art methods.

Limitation. Despite promising results, our approach still relies on a synthetic 3D dataset to gain prior knowledge of 3D shapes for reconstruction. Exploring reconstruction by learning directly from real data is worthy of further study.

Acknowledgments

The authors gratefully acknowledge the support of MindSpore, CANN, and Ascend AI Processor. The authors would also like to thank Andrew Yang and Hunter Schofield for their assistance.

References

- [1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301, 2022.
- [2] Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman. Multi-chart generative surface modeling. *ACM Transactions on Graphics*, 37(6):1–15, 2018.
- [3] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. Spie, 1992.
- [4] Hongrui Cai, Wanquan Feng, Xuetao Feng, Yan Wang, and Juyong Zhang. Neural surface reconstruction of dynamic scenes with monocular rgb-d camera. In *Proc. of Conference on Neural Information Processing Systems*, 2022.
- [5] Yingjie Cai, Kwan-Yee Lin, Chao Zhang, Qiang Wang, Xiaogang Wang, and Hongsheng Li. Learning a structured latent space for unsupervised point cloud completion. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5543–5553, 2022.
- [6] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Proc. of European Conference on Computer Vision*, pages 608–625. Springer, 2020.
- [7] 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.
- [8] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.
- [9] Angela Dai, Christian Diller, and Matthias Nießner. Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 849–858, 2020.
- [10] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 5868–5877, 2017.
- [11] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevata, and Leonidas J Guibas. Curriculum deepsdf. In *Proc. of European Conference on Computer Vision*, pages 51–67. Springer, 2020.
- [12] Shivam Duggal, Zihao Wang, Wei-Chiu Ma, Sivabalan Manivasagam, Justin Liang, Shenlong Wang, and Raquel Urtasun. Mending neural implicit modeling for 3d vehicle reconstruction in the wild. In *Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1900–1909, 2022.
- [13] Francis Engelmann, Jörg Stückler, and Bastian Leibe. Samp: shape and motion priors for 4d vehicle reconstruction. In *Proc. of IEEE Winter Conference on Applications of Computer Vision*, pages 400–408. IEEE, 2017.
- [14] Qiaojun Feng, Yue Meng, Mo Shan, and Nikolay Atanasov. Localization and mapping using instance-specific mesh models. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4985–4991. IEEE, 2019.
- [15] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.
- [16] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018.
- [17] Jiayuan Gu, Wei-Chiu Ma, Sivabalan Manivasagam, Wenyuan Zeng, Zihao Wang, Yuwen Xiong, Hao Su, and Raquel Urtasun. Weakly-supervised 3d shape completion in the wild. In *Proc. of European Conference on Computer Vision*, pages 283–299. Springer, 2020.
- [18] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. Semantic3d.net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847*, 2017.
- [19] Tao Hu, Zhizhong Han, Abhinav Shrivastava, and Matthias Zwicker. Render4completion: Synthesizing multi-view depth maps for 3d shape completion. In *Proc. of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.
- [20] Huawei. Mindspore. <https://www.mindspore.cn/>, 2020.
- [21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conference on Learning Representations (Poster)*, 2015.
- [22] Chen Kong, Chen-Hsuan Lin, and Simon Lucey. Using locally corresponding cad models for dense 3d reconstructions from a single image. In *Proc. the IEEE conference on computer vision and pattern recognition*, pages 4857–4865, 2017.
- [23] Chenqi Li, Yuan Ren, and Bingbing Liu. Pcgen: Point cloud generator for lidar simulation. In *Proc. of International Conference on Robotics and Automation*, 2023.
- [24] 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.
- [25] Shirui Luo, Changqing Zhang, Wei Zhang, and Xiaochun Cao. Consistent and specific multi-view subspace clustering. In *Proc. of the AAAI conference on artificial intelligence*, volume 32, 2018.
- [26] Xiaoxu Meng, Weikai Chen, and Bo Yang. Neat: Learning neural implicit surfaces with arbitrary topologies from multi-view images. *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2023.
- [27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks:

- Learning 3d reconstruction in function space. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2019.
- [28] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.
- [29] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011.
- [30] Liang Pan, Tong Wu, Zhongang Cai, Ziwei Liu, Xumin Yu, Yongming Rao, Jiwen Lu, Jie Zhou, Mingye Xu, Xiaoyuan Luo, et al. Multi-view partial (mvp) point cloud challenge 2021 on completion and registration: Methods and results. *arXiv preprint arXiv:2112.12053*, 2021.
- [31] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.
- [32] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.
- [33] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.
- [34] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Proc. of IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009.
- [35] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.
- [36] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proc. of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.
- [37] Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 2897–2905, 2018.
- [38] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 2598–2606, 2018.
- [39] Dan Wang, Xinrui Cui, Xun Chen, Zhengxia Zou, Tianyang Shi, Septimiu Salcudean, Z Jane Wang, and Rabab Ward. Multi-view 3d reconstruction with transformers. In *Proc. of the IEEE/CVF International Conference on Computer Vision*, pages 5722–5731, 2021.
- [40] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proc. of the European conference on computer vision*, pages 52–67, 2018.
- [41] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021.
- [42] Xiaogang Wang, Marcelo H Ang, and Gim Hee Lee. Voxel-based network for shape completion by leveraging edge generation. In *Proc. of the IEEE/CVF international conference on computer vision*, pages 13189–13198, 2021.
- [43] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *Proc. of the European conference on computer vision*, pages 365–381. Springer, 2020.
- [44] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022.
- [45] Jianglong Ye, Yuntao Chen, Naiyan Wang, and Xiaolong Wang. Gifs: Neural implicit function for general shape representation. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12829–12839, 2022.
- [46] Jianglong Ye, Yuntao Chen, Naiyan Wang, and Xiaolong Wang. Online adaptation for implicit object tracking and shape reconstruction in the wild. *IEEE Robotics and Automation Letters*, 7(4):8909–8916, 2022.
- [47] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proc. of the IEEE/CVF international conference on computer vision*, pages 12498–12507, 2021.
- [48] Xumin Yu, Yongming Rao, Ziyi Wang, Jiwen Lu, and Jie Zhou. Adapointr: Diverse point cloud completion with adaptive geometry-aware transformers. *arXiv preprint arXiv:2301.04545*, 2023.
- [49] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *Proc. of International Conference on 3D Vision*, pages 728–737, 2018.
- [50] Changqing Zhang, Zongbo Han, Huazhu Fu, Joey Tianyi Zhou, Qinghua Hu, et al. Cpm-nets: Cross partial multi-view networks. *Advances in Neural Information Processing Systems*, 32, 2019.