VPC-Net: Completion of 3D Vehicles from MLS Point Clouds

Yan Xia[a], Yusheng Xu[a], Cheng Wang[b], and Uwe Stilla[a]

[a]Photogrammetry and Remote Sensing, Technical University of Munich (TUM)
{*yan.xia, yusheng.xu, stilla*}*@tum.de*
[b]Fujian Key Laboratory of Sensing and Computing, School of Informatics, Xiamen University
cwang@xmu.edu.cn

Abstract

Vehicles are the most concerned investigation target as a dynamic and essential component in the road environment of urban scenarios. To monitor their behaviors and extract their geometric characteristics, an accurate and instant measurement of the vehicles plays a vital role in remote sensing and computer vision field. 3D point clouds acquired from the mobile laser scanning (MLS) system deliver 3D information of unprecedented detail of road scenes along with the driving. They have proven to be an adequate data source in the fields of intelligent transportation and autonomous driving, especially for extracting vehicles. However, acquired 3D point clouds of vehicles from MLS systems are inevitably incomplete due to object occlusion or self-occlusion. To tackle this problem, we proposed a neural network to synthesize complete, dense, and uniform point clouds for vehicles from MLS data, named Vehicle Points Completion-Net (VPC-Net). In this network, we introduced a new encoder module to extract global features from the input instance, consisting of a spatial transformer network and point feature enhancement layer. Moreover, a new refiner module is also presented to preserve the vehicle details from inputs and refine the complete outputs with fine-grained information. Given the sparse and partial point clouds of vehicles, the network can generate complete and realistic structures, and keep the fine-grained details from the partial inputs. We evaluated the proposed VPC-Net in different experiments using synthetic and real-scan datasets and applied the results to 3D vehicle monitoring tasks. Quantitative and qualitative experiments demonstrate the promising performance of VPC-Net and show state-of-the-art results.

1. Introduction

Vehicles are the most concerned investigation targets as a dynamic and essential component in the road environment of urban scenarios. For monitoring their behaviors and extracting their geometric characteristics, an accurate and instant measurement of the vehicles plays a vital role. The measurements can be achieved via either cameras or laser scanners providing 2D images [37] and 3D point clouds [44, 45], respectively. Comparing with 2D images, 3D point clouds acquired from mobile laser scanning (MLS) systems deliver detailed 3D information of road scenes during the driving. Acquiring point clouds via a MLS system provides an efficiently solution for tasks like autonomous driving and urban street mapping. Especially for vehicle extraction, MLS systems have been chosen as a key sensor by plenty of autonomous driving companies and research institutes since it can provide highly accurate geometric information (e.g., 3D coordinates of vehicle points) and reliable ratiometric attributes (e.g., reflectivities of various surface materials) of multiple instances simultaneously.

However, acquired 3D point clouds of vehicles from MLS systems are inevitably incomplete due to object occlusion or self-occlusion. For instance, in Fig. 1a, a few typical point clouds of vehicles on urban roads from the KITTI dataset [10] are illustrated. As seen from this figure, we can clearly observe the missing parts in the vehicles' scanned point clouds. This incompleteness significantly hinders the potential uses of vehicle point clouds, because it actually has changed the dimension of shapes, biased the volume of objects, and destroyed the topology of the surfaces. In generic applications like 3D traffic monitoring, the complete geometric shapes of vehicles provide solid foundations for 3D perceptual tasks, including instance extraction, type classification, and track estimation [39]. Recently, [47] proposed a alternative strategy, which first estimates the vehicle poses and then retrieve the similar CAD model of this

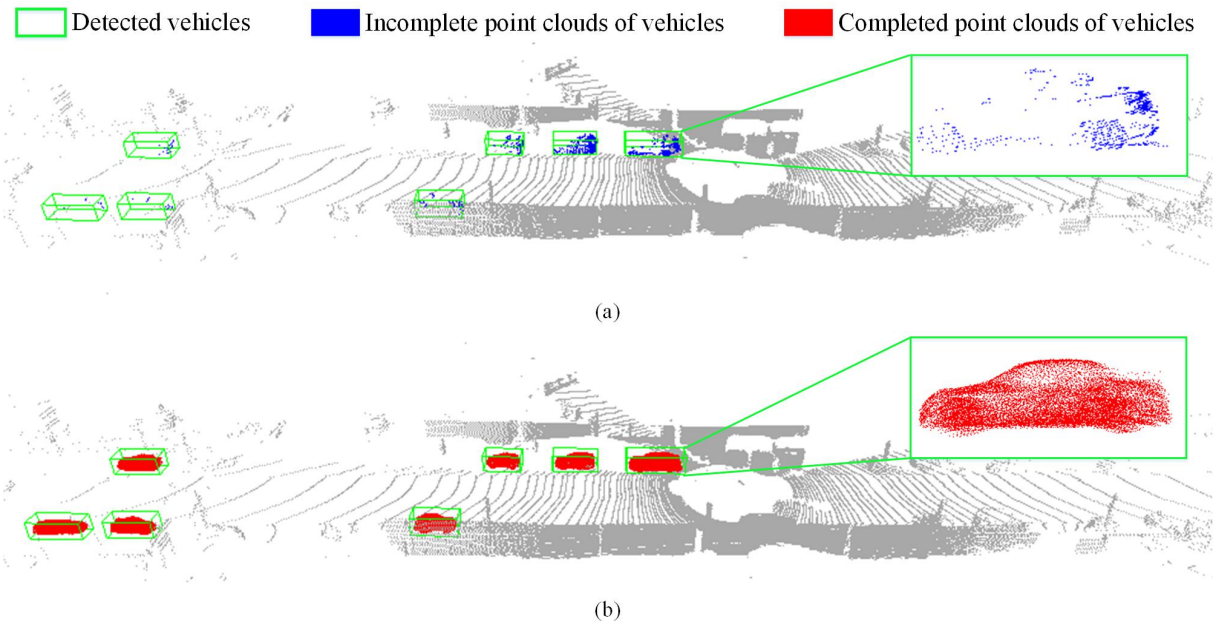

Figure 1. Illustration of the incomplete and complted point clouds of vehicles. (a) The single-frame raw real-scan data from KITTI [10]. (b) Completed scan generated by the proposed VPC-Net.

vehicle from the large-scale CAD model datasets to replace the raw point clouds. However, this method can not deal with occluded vehicles, and it is unable to preserve the real knowledge of raw point clouds. Besides, for specific applications like the measure of vehicle-induced aerodynamic loads in bridge engineering, the complete surface, as well as the shape, of the measured vehicles is the key to the estimation wind pressure caused by the vehicles driving close the sound barrier, which has a considerable impact on designing the structure of urban highway viaducts [21].

For addressing the problem of incompleteness, the conventional strategy of using multiple scans cannot work for dynamic moving vehicles, so the completion of sparse and partial point clouds becomes a possible solution. Nevertheless, this is not an easy task since the incomplete point clouds miss geometric and semantic information. The challenges include the following aspects:

- Compensate missing points and holes and recover the occluded structures.
- Guarantee an adequate point density for vehicle modeling with an even distribution.
- Reconstruct the proper topology of shapes and geometric details of objects.

In this paper, we propose a neural network, named Vehicle Points Completion-Net (VPC-Net), to synthesize complete, dense, and uniform point clouds for vehicles from MLS data. Given the sparse and partial point clouds of vehicles, our network can generate complete and realistic structures, and keep the fine-grained details from the partial inputs, as shown in Fig. 1b. The significant contributions are as follow:

- We design a novel end-to-end network (termed as VPC-Net) for completing point clouds of 3D vehicle shapes, operating on the partial and sparse point clouds directly. By endorsing an architecture with the encoder, decoder and refiner, VPC-Net can produce uniform, dense, and complete point clouds from partially scanned vehicles in MLS datasets.
- We propose a novel encoder module to better extract global features from the instance, including a spatial transformer network (STN) and point feature enhancement (PFE) layer. STN ensures the extracted features are invariant to geometric transformations from input point clouds with different resolutions. Besides, the PFE layer combines the low-level and high-level information to enhance the feature representation ability.
- We propose a new refiner module to preserve the vehicle details from inputs and refine the complete outputs with fine-grained details. To fully retain the details of the input point cloud, the partial inputs and outputs generated by the decoder are combined uniformly. Besides, the point feature residual network is designed to predict per-wise offsets for every point.
- We conduct experiments on one 3D synthetic dataset

(i.e., ShapeNet [5]) and two real MLS datasets (i.e., KITTI [10] and TUM-MLS-2016 [49]), demonstrating that the proposed network VPC-Net achieves superior performance over the baseline methods. Also, we adopt this method to the 3D vehicle monitoring task, which completes dynamic 3D vehicles of the 3D scene online only based on the single frame of raw real-scan data, not relying on the information of time sequences.

The remainder of this paper is organized as follows. Section 2 briefly reviews and discusses the related works for point clouds and shapes completion. Section 3 introduces an overview of the proposed network VPC-Net. Section 4 elaborates on the detailed architecture of the networks, including explanations on each module. Section 5 presents the experiments. Experimental results are shown in Section 6 and Section 7 gives a detailed discussion and analysis of the derived results. Finally, Section 8 concludes the paper and outlines future work.

2. Related work

3D shape completion has long been an attractive research topic in robotics and computer vision for many years [1, 12]. There is a series of methods for recovering complete geometry information from partial point clouds in recent years. Generally, the related methods can be primarily classified into three major categories: (i) geometry-based methods, (ii) template-based methods, and (iii) learning-based methods. In the following subsection, we briefly review these three types of methods.

2.1. Geometry-based shape completion

The geometry-based completion methods depend highly on the geometric cues, such as the continuity of local surfaces or volumetric smoothness, which have been applied to retouch the small holes on the incomplete point clouds successfully [15, 32, 40]. However, it is unable for completing missing points of larger regions. Thus, approaches using hand-designed heuristics are proposed to reconstruct surfaces of the 3D objects with a large percentage of missing areas. For example, [27] presented a method to complete 3D shapes with merely partial inputs by combining a series of planes and cylinders. Furthermore, in [17], relations among geometric shapes like planes and cylinders were proposed to be learned, which is beneficial for improving the completing performance. For objects with arterial surfaces, [16] proposed a novel deformable model named arterial snake and successfully captured the topology and geometry simultaneously from arterial objects with noise and large parts missing.

Additionally, [36, 48, 23, 35, 13] considered the symmetry of human-made objects, which usually have structural regularity. [36] identified the probable symmetries and applied them to extend the partial 3D model to the occluded space. [23] leveraged regular structures that form a lattice with discrete rotational, translational, and scaling symmetries to fill the missing regions. [48] automatically consolidated and densified real-scan data by detecting repeating structures in input 3D models. [35] explored to quantify the relationship between shapes based on the regularities of symmetric parts. The shape of objects is firstly decomposed into a set of regions and then a graph is applied to represent the relations between the regions in terms of symmetric transformations. [13] utilized context information to synthesize geometry that is similar to the remainder of the input objects. However, all these methods are limited to only completing input point clouds with moderate degrees of missing regions.

2.2. Template-based Shape Completion

In addition to the geometry-based methods, some methods follow an alternative strategy, in which they will complete the 3D surfaces by deforming or reconstructing the point clouds according to the retrieved most similar templates from a prepared 3D shape database. This type of methods are called template-based methods, which are also known as retrieval-based methods. As a precondition for the retrieval, a 3D shape database was created in [22] to extract some geometric clues for completing missing regions. However, this method embedding a database retrieval process is time-consuming and labor-intensive since manual interaction is inevitable to constraint the categories of 3D objects. Similarly, [25] proposed a novel completing method automatically for any categories of objects based on the use of additional depth image as auxiliary data. An adequate auxiliary database with sufficient elements plays an essential role in the performance of this method.

To avoid the high dependency of large-scale 3D shape databases, some works [27, 20, 6, 17, 28, 31]were proposed to apply geometric primitives in place of a shape database. For example, [27] reconstructed the missing parts with the guidance of a set of detected primitive shapes (e.g. planes, cylinders etc.) [20] presented a novel interactive tool called SmartBoxes to reconstruct the structures which are partially missing from inputs. It allowed the user interactively to fit polyhedral primitives, avoiding the expensive search. [6] plausibly completed the missing scene parts by decomposing 3D space based on planar primitives. [17] explored to simultaneously recover the local missing parts using structural relations from man-made objects, which must include basic primitives. [28] presented an assembly approach using predefined geometric primitives to recover 3D structures with a small-scale shape dataset. [31] employed a global optimization method to reconstruct entire surfaces using inference from given geometric information from partial inputs.

However, such methods exist several limitations. First

of all, these are not suitable for online applications since the optimization schemes are time-consuming. Secondly, preparing a 3D shape database is labor-intensive since every shape is labeled and segmented manually. At last, they are not always robust to noise or disturbances (e.g., dynamic changes).

2.3. Learning-based Shape Completion

Recently, learning-based methods for 3D shape completion have obtained significant developments with the emergency of large-scale 3D synthetic CAD model datasets. The state-of-art methods have shown excellent performance on various representing formats of 3D models, including voxel grids, point clouds, and meshes. Earlier studies of this category of methods [8, 29, 30] tend to choose the voxel grids as the representation of 3D shapes since 3D convolution and distance field formats are well suited for processing this kind of discrete and rasterized data. [8, 29] are typical examples of voxel-based methods, which adopted a 3D convolutional network to achieve the excellent performance of completing shapes. [30] proposed a weakly-supervised learning-based method to complete a 3D shape, which is more available in practice. However, the voxelization representation reveals a series of issues. For instance, grid occupancy is predicted independently, which results in the complete results often miss thin structures or contain flying voxels. Moreover, volumetric representation obscures natural invariance when it comes to geometric transformations and manipulations. In addition, it is computationally expensive to predict volumes of high spatial resolution.

Thus, some recent works [11, 38, 18], which focus on the completion using a mesh-based representation, have emerged. [11] proposed a novel shape generation network called AtlasNet, which represents a 3D shape as a collection of parametric surface elements. [38] introduced a graph-based network named Pix2Mesh to reconstruct 3D manifold shapes. [18] explored a variational autoencoder using graph convolutional operations to deformable meshes, which focus on the certain objects that undergo non-rigid deformations such as faces or humans. However, they reconstruct the shape information by deforming a reference mesh to a target mesh. Therefore, they are not flexible with any typologies.

In comparison to 3D meshes, or voxels representations, point clouds are a simple structure during the network training procedure. Besides, newly created points can easily be added or interpolated to a point cloud since all the points are independent and we do not need to update the connectivity information. Some recent work also processes discrete points without structuring via voxels or meshes. For example, PCN [46] was the first approach that directly operated on raw point clouds and outputted complete and dense point clouds robustly with partial inputs. Furthermore, TopNet [34] proposed a tree-structured decoder for point cloud generation. However, they are unable to simultaneously produce evenly distributed and complete point clouds with fine-grained details. In this work, our method falls into this category and builts upon the recent network PCN. Different from PCN, our model can generate more uniform point clouds with fine-grained details.

3. Overview of methodology

The point cloud completion task can be regarded as a set problem: given the partial and low resolution points $X = \{P_i : i = 1, ..., N\}$, the proposed network VPC-Net aims to generate the complete 3D point clouds $Y = F(x) = F(P_i : i = 1, ..., N)$ with F being the prediction function. Notably, X is not necessarily a subset of Y since they can be obtained from vehicle surface independently. The critical architecture of VPC-Net is shown in Fig. 2, which consists of the encoder module, decoder module, and refiner module. Firstly, the encoder is to extract the global features from the raw and sparse point clouds. Secondly, the decoder consists of two parts: (i) it takes the generated global features as input to produce the coarse but complete point cloud, (ii) it combines the coarse point cloud and global feature to generate dense point clouds. Finally, we use the skip-connection to concatenate the partial inputs with the previous dense point cloud for preserving the original details. Then the refiner further refines the fused 3D point clouds to produce the final completion result. The generated point clouds by the network VPC-Net should include three outstanding characteristics: (i) completing the missing surface with fine-grained structures; (ii) preserving the original details of the inputs; (iii) producing the uniform point clouds.

4. Network architecture

The detailed architecture of VPC-Net is shown in Fig. 3. It includes three sub-networks: feature extraction (encoder), coarse-to-dense reconstruction (decoder), and the refiner. The detailed explanation of each core step will be introduced in the following sections.

4.1. Encoder

The encoder is to build a set of features F for the decoder to estimate the missing surface of 3D vehicles. Therefore, the feature extraction ability of encoder plays a vital role in the whole network. If the encoder can effectively combine the local features and global features from the partial inputs, it is significantly beneficial for the 3D coordinates regression of the dense point cloud generation. Our encoder consists of two modules: one spatial transform network module and one global feature extraction module. Especially, it can be modeled by the combination of two functions, which is

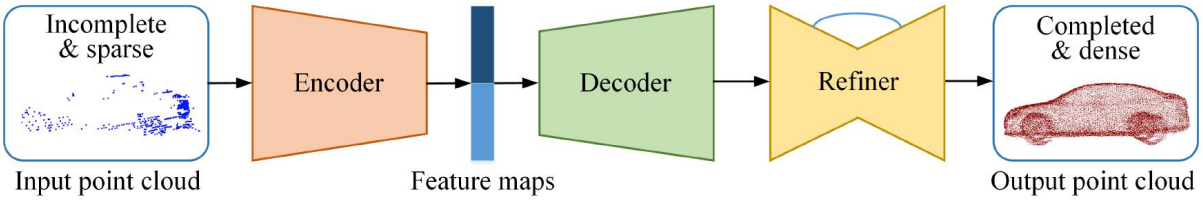

Figure 2. Workflow of the proposed network VPC-Net.

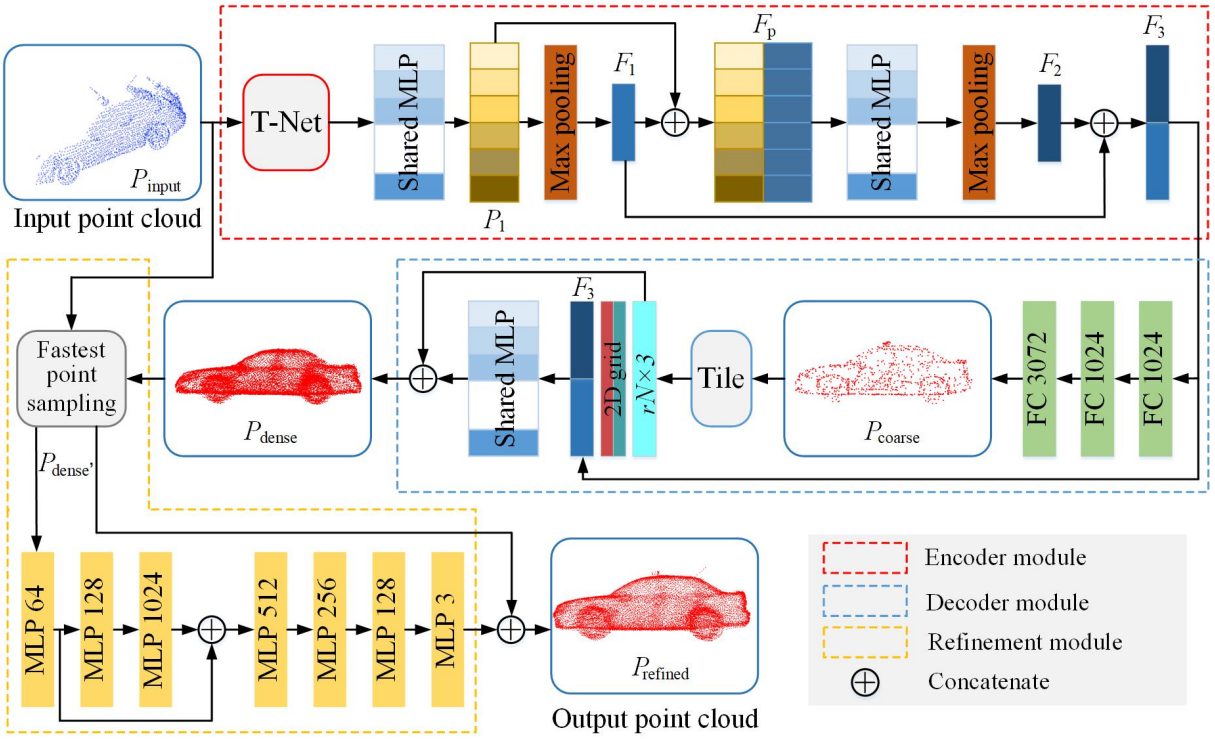

Figure 3. The network architecture of VPC-Net.

defined as follows:

$$F = Q(P_{input}|w_Q), Q = Q_1 \circ Q_2 \quad (1)$$

where Q_1 and Q_2 are the spatial transform network and the global feature extraction module, respectively. And w_Q denotes the weight parameters of Q, P_{input} is the partial input point cloud.

4.1.1 Spatial transform network

Since the input point clouds of the vehicle are disordered and their poses are diverse, this will cause difficulty in accessing the unified features for neural networks. Therefore, in order to facilitate the extraction of features, we hope the input point clouds have a neat pose. In other words, learned features from input point sets should be invariant to geometric transformations.

Aligning all point sets to a canonical space is a natural solution. [14] used a spatial transformer for learning invariance to translation and rotation in 2D images. Inspired this, we propose to adopt a 3D spatial transform network named T-Net [24] to predict a 3×3 transformation matrix for the original point clouds. Furthermore, we directly multiply this transformation matrix and the coordinates of input points. Thus, the inputs are aligned to a canonical space so that the following network can attentively learn a unified and standardized feature.

The T-Net is like a mini-PointNet [24], which includes a shared shared Multiple Layer Perception (MLP) network, a max-pooling layer, and two fully connected layers. It takes the raw point clouds as inputs and outputs to a 3×3 matrix. In detail, the MLP network first encode each point to multiple dimensions $[64, 128, 1024]$. Then a max-pooling layer across these points and followed by two fully con-

nected layers with output sizes $[512, 256]$. The regressed matrix is initialized as an identity matrix. Except for the last layer, all layers followed by a ReLU activation and a batch normalization layer.

4.1.2 Global feature extraction module

Generally, our global feature extraction module is based on recently advanced feature extraction network PointNet [24], which directly operates on the point clouds. Inspired by this, the encoder, as illustrated in Fig. 3, adopts two stacked PointNet layers to extract the geometric information for the input point cloud. Each PointNet layer comprises of one shared MLP and one max-pooling layer as a basic module. In the first PointNet layer, we learn a point-wise feature P_1 from the points of $N_{input} \times 3$ transformed by the STN, where N_{input} is the amount of points and *3* is the *x,y,z* coordinates of each point. Afterwards, a max-pooling layer is employed on P_1 to output a 256-dimensions local feature vector F_1. In the second PointNet layer, we firstly concatenate the local latent space with every independent point feature by feeding F_1 back to the point-wise feature P_1. Then the global latent vector F_2 is extracted from the aggregated point features F_p through the second PointNet layer, where size $F_2 := 1024$.

However, it always loses the fine details of the inputs since the latent space extracted by the last max-pooling layer only represents the rough global shape. Inspired by the skip-connection from U-Net [26], we design a point feature enhancement (PFE) layer, which concatenate the global feature F_2 with the local feature F_1 to synthesize the final feature space F_3. Size $F_3 := 1280$, and it includes both low-level and high-level feature information. Experimental results in Section 7.2 show this design improves the feature extraction ability of encoder for partial inputs.

4.2. Decoder

The decoder is responsible for converting the final global latent vector F_3 into dense, evenly, and complete 3D point clouds. In this stage, a coarse-to-fine completion strategy is applied for generating the 3D coordinates of point clouds. Inspired by 3D object reconstruction network RealPoint3D [41], we explore three fully-connected layers to generate a sparse point cloud with a complete geometric surface. Lastly, it outputs the final vector with $3N$ units and we reshape it to a $N \times 3$ coarse point cloud P_{coarse}.

However, the fully-connected layer is not suitable to generate dense points. It will cause some points to be over-concentrated when we regress a large number of points. The reason is fully-connected layers are not restrictive on the local density. Therefore, in the second stage, we first tile the points in P_{coarse} to produce a dense point set $P^{'}_{coarse} := rN \times 3$, where r is the up-sampling rate. Then we apply a folding-based operation [43] to deform a unique 2D grid vector and concatenate it with each point of the coarse point cloud to obtain new patches. Actually, this operation can increase the difference between the duplicated points. In other words, we regard each point of the coarse point cloud as spatial keypoints and take them as center points to generate a series of surrounding points. To make full use of the features of input point clouds, we concatenate the points $P^{'}_{coarse}$, global feature space F_3 and the deformed 2D grids to obtain a new aggregated feature. Finally, we pass the a aggregated feature through a shared MLP and then transformed into the dense outputs P_{dense} by adding each point in $P^{'}_{coarse}$ to the output .

4.3. Refiner

Although the decoder can produce impressive results, it always lost the fine-grained details of the inputs, and the points are unevenly distributed. To tackle these problems, we propose to combine the partial inputs P_{input} with the outputs P_{dense} generated by the decoder. This operation can fully retain the details of the input point cloud. However, the linear combination will cause the merged points to be not uniform since the two point clouds have a different density and there maybe overlapping between them. Thus, we apply the farthest point sampling (FPS) to sample a uniform distributed subset point cloud $P^{'}_{dense}$ with a size of $rN \times 3$.

The refiner can be regarded as a point feature residual network. We hope the refiner can predict per-wise offsets o_x, o_y, o_z for every point in $P^{'}_{dense}$. Therefore, we pass the points $P^{'}_{dense}$ through a series of MLPs to predict point feature residuals since neural networks are better at residuals [38]. Specifically, inspired by the structure of encoder-decoder network, we adopt a bottom-up and top-down strategy to refine points coordinates. The refiner consist of seven MLPs. It first encode each point into multiple dimensions $[64, 128, 1024]$. Then we decode it to generate the offsets of each point with dimensions of $[512, 256, 128, 3]$. Expect for the last layer, followed by a batch normalization layer and a Tanh activation, other MLPs are followed by a batch normalization layer. In addition, we hope the local feature can be preserved in the following layers. Thus, we combine the feature with dimensions of 64 and the bottleneck layer with the size of 1024, as shown in Fig. 3. Overall, in this refiner, the final generated point sets $P^{'}_{dense}$ is defined as:

$$P_{refined} = R(P^{'}_{dense}) + P^{'}_{dense} \tag{2}$$

where $R\{.\}$ predicts per-wise displacements by the refiner.

4.4. Loss function

The loss function of our network is defined as the topological distance between the completed object and the

ground truth. Inspired by [9], we adopt the Chamfer Distance (CD) and Earth Mover's Distance (EMD) to optimize the network. These distance functions are highly efficient and invariant to permutations of the relative ordering of points. Chamfer distance between the completed point cloud P_c and the ground truth P_{gt} is defined as:

$$\begin{aligned} d_{chamfer}(P_c, P_{gt}) = &\sum_{x \in P_c} \min_{y \in P_{gt}} \| x - y \|_2^2 \\ &+ \sum_{x \in P_{gt}} \min_{y \in P_c} \| x - y \|_2^2 \end{aligned} \tag{3}$$

where $P_c, P_{gt} \subseteq R^3$. Intuitively, it aims to find the closet neighbor between the two point sets in two directions. Each point of P_c is mapped to the closet point in P_{gt}, and vice versa. Thus, the size of P_c and P_{gt} is not required to be same. Besides, it is a computationally light function with $O(nlogn)$ complexity for the nearest neighbor search. However, it is a problematic metric since it can not ensure the uniformity of predicted points [19]. In addition, it is sensitive to the detailed geometry of outliers [33]. To alleviate these problems, Earth Mover's Distance between P_c and P_{gt} is proposed by:

$$d_{EMD}(P_c, P_{gt}) = \min_{\phi: P_c \to P_{gt}} \sum_{p \in P_c} \| p - \phi(p) \|_2 \tag{4}$$

where $P_c, P_{gt} \subseteq R^3$, $\phi : P_c \to P_{gt}$ is a bijection. Unlike CD, it requires the size of P_c and P_{gt} must be the same since it is a point-to-point mapping function. However, it has a major drawback that the $O(n^2)$ computing complexity is too expensive. It is not suitable for predicting dense points in the network.

Therefore, we propose a training strategy that can take advantage of both the distance functions. To make sure the generated coarse point cloud to be evenly and have general geometry, we apply the EMD loss for P_c predicted by the encoder. The predicted dense point clouds P_{dense} and $P^{'}_{dense}$ are optimized via the CD loss. More formally, the overall loss is defined as:

$$\begin{aligned} L(P_{coarse}, P_{dense}, P^{'}_{dense}, P_{gt}) = &\, d_{EMD}(P_{coarse}, \tilde{P_{gt}}) \\ &+ \gamma d_{chamfer}(P_{dense}, P_{gt}) \\ &+ \beta d_{chamfer}(P^{'}_{dense}, P_{gt}) \end{aligned} \tag{5}$$

where $\tilde{P_{gt}}$ is the subsampled ground truth with the same size as P_{coarse}. γ and β are hyperparameters to balance the relationship of them.

5. Experiments

In this section, we performed experiments to demonstrate the effectiveness of the proposed VPC-Net when completing point clouds of real LiDAR scans. We will first introduce the experimental datasets and the generation of training data in Section 5.1. In Section 5.2, we will describe the evaluation metrics used for assessing the performance of VPC-Net, as well as baseline methods. Furthermore, the implementation details and training processing will be introduced in Section 5.3.

5.1. Experimental datasets

In experiments, we tested our proposed VPC-Net method on three different datasets, including ShapeNet dataset [5], KITTI dataset [10], and TUM-MLS-2016 dataset [49].

5.1.1 ShapeNet dastaset

ShapeNet [5] is a richly-annotated and large-scale 3D synthetic dataset, which covered 220,000 CAD models and 3,135 categories of objects. In this work, we use synthetic CAD models on the category of cars from ShapeNet to create a vehicle dataset containing pairs of partial and complete point clouds, in order to train our model. Specifically, it includes a total of 5677 different instances of vehicles, which are split into the training data, validation data, and test data. Among them, 100 instances are used for validation and 150 instances are utilized for testing. The rest instances are reserved for training. For creating complete point clouds as ground truth, for each CAD model of a vehicle instance, 16384 points are sampled uniformly on the surface of each CAD model of a vehicle as the synthetic point cloud. Fig. 4 shows examples of complete point clouds of vehicle instances from CAD models in ShapeNet. Instead of using subsets of complete point clouds as partial inputs, we render the CAD models of vehicle instances to a set of depth images from a variety of view angles and then back-projected these depth images to different view planes to generate partial point clouds. This operation can make the incompleteness distribution of partial point clouds closer to real-scan data.

In Fig. 5, we illustrate the pipeline of generating partial inputs from the ShapeNet dataset. The depth images are generated by placing a virtual RGB-D camera at different view angles. The camera is designed to be oriented towards the center of the 3D model. Then, we randomly select a series of viewpoints only to generate incomplete shapes simulating scan obtained through limited view access. Lastly, the resulting depth maps are back-projected to form partial point clouds. In this work, we choose eight randomly distributed viewpoints to generate eight partial point clouds for each training 3D CAD model of a vehicle. Notably, the resolution of these partial scans can be different. The reason for generating training point clouds from a synthetic 3D dataset is that it consists of a wide vari-

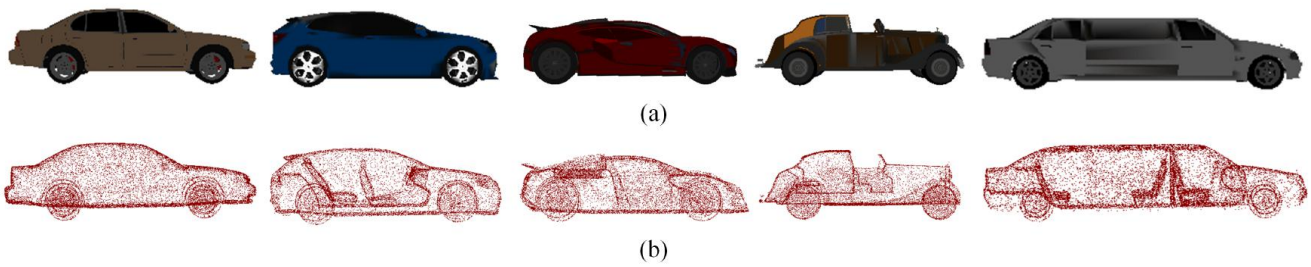

Figure 4. Examples of CAD models and sampled point clouds of vehicle instances from the ShapeNet dataset. (a) CAD models of vehicle instances stored in ShapeNet. (b) Generated complete point clouds sampled uniformly from these CAD models.

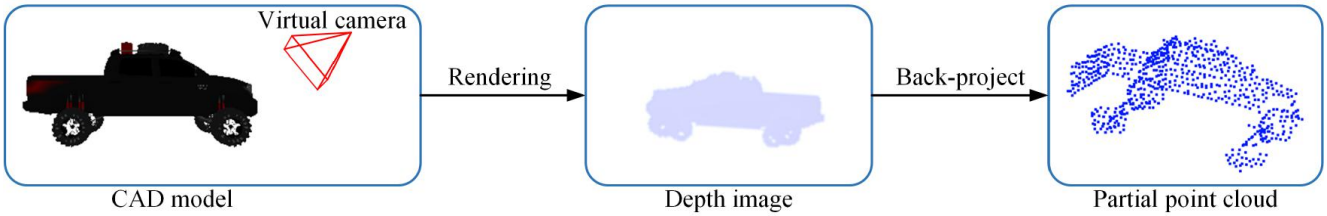

Figure 5. The pipeline of partial inputs generation.

ety of complete and detailed 3D vehicle models, while they are not available in real-scanned LiDAR datasets. Moreover, scanning thousands of vehicles using LiDAR systems for acquiring complete point clouds as the ground truth is quite time-consuming and labor-intensive, which is not a practical solution. Recently, some high-quality 3D reconstruction datasets have emerged like ScanNet [7] and S3DIS [2], which can also provide training data with high quality. However, they are mainly focused on the indoor scene, not including any objects in outdoor scenarios.

5.1.2 KITTI dataset

KITTI [10] dataset provides raw point clouds collected by Velodyne HDL-64E rotating 3D laser scanner and annotations for the vehicle instances in the form of 3D bounding boxes. It records six hours of traffic scenarios, which are diverse and capturing real-world traffic situations with many static and dynamic vehicles. The raw dataset includes five categories of objects, namely 'Road', 'City', 'Residential', 'Campus', and 'Person'. In the data category 'City', it is composed of about 28 sequences (i.e., 8477 frames). In each sequence of the raw data, apart from objects annotated with 3D bounding boxes, tracklets and calibration are also provided. Three example frames in 'City' are shown in Fig. 6. As seen in Fig. 6, we can find that the major challenge of this dataset is twofold. One is that the point clouds of vehicles are very sparse with significant missing of contents, while another is that target vehicles appear in an arbitrary location with variable sizes. In this work, we took one sequence in the category 'City' from the KITTI dataset as experimental data. Specifically, we extracted 2483 partial point clouds of vehicles from every frame based on their bounding boxes.

5.1.3 TUM-MLS-2016 dataset

TUM-MLS-2016 [49] is a mobile laser scanning dataset covering around 80000 m^2 with annotations. This dataset was acquired by Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB) via two Velodyne HDL-64E laser scanners and then annotated by the Chair of Photogrammetry and Remote Sensing of TUM. Unlike the KITTI dataset, the TUM-MLS-2016 dataset provides an aggregated point cloud of the whole obtained sequence. It covers an urban area of approximately 1 km long roadways and includes more than 40 million annotated points with eight classes of objects labels. In Fig. 7, we give an illustration of scanned vehicles on the Arcisstrasse of this dataset. We extracted point clouds of vehicles as the testing data based on the provided annotations of parked vehicles. From Fig. 7, it is seen that the point clouds of vehicles in the TUM-MLS-2016 dataset are denser than in those in the KITTI dataset. However, they are also incomplete, although the extent of missing is relatively smaller.

5.2. Evaluation metrics

The performance of our proposed method is evaluated by two commonly applied metrics: CD (see Eq. 3) and EMD (see Eq. 4) between the completed point cloud and ground

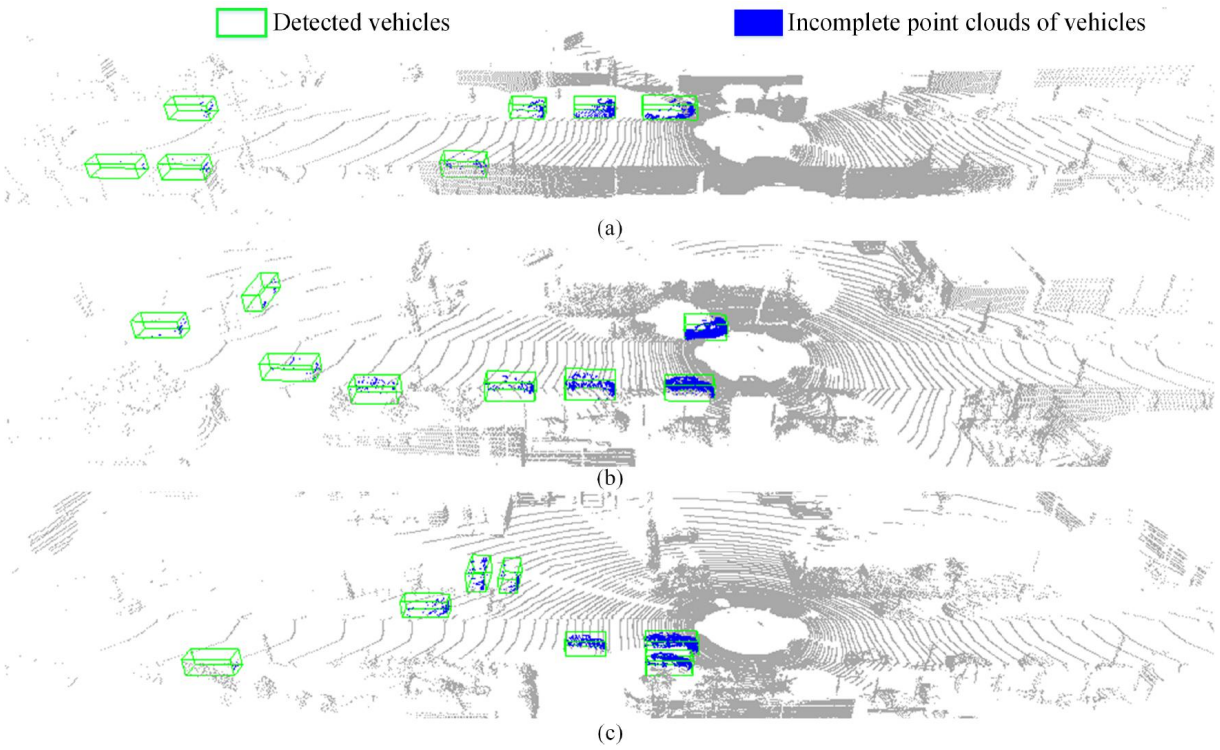

Figure 6. Example frames on 'City' category from the KITTI dataset. The vehicle points, background points and bounding boxes are shown in blue, gray, and green colors, respectively.

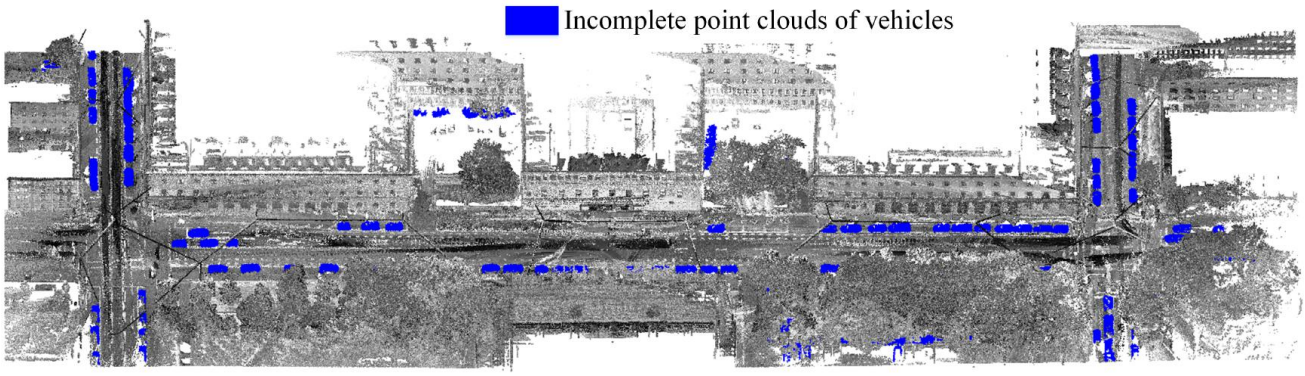

Figure 7. Point clouds of Arcisstrasse from the TUM-MLS-2016 dataset. The vehicle points and background points are shown in blue and gray colors, respectively.

truth. The definitions of CD and EMD have been given in Section 4.4. For computing the metrics with a lower computational cost, we normalized the dimensions of both the ground truth and completed point clouds, by regarding the length of the bounding box of length as one unit.

5.3. Implementation details and training process

The proposed network VPC-Net is implemented in the Tensorflow framework and trained on a single Nvidia Titan Xp GPU with 12G memory. In the training stage, we set the batch size to eight. Adam optimizer is used in the models for 100K steps. The size of the coarse output generated by the encoder is 1024. The initial learning rate is set to 0.0001. They are decayed by 0.7 after every 50K steps and clipped by 10^{-6}. γ and β are set to equal. They gradually increase from 0.01 to 1 in the first 50K steps. Notably, the resolutions of the inputs are various, from a few hundred points to thousands of points.

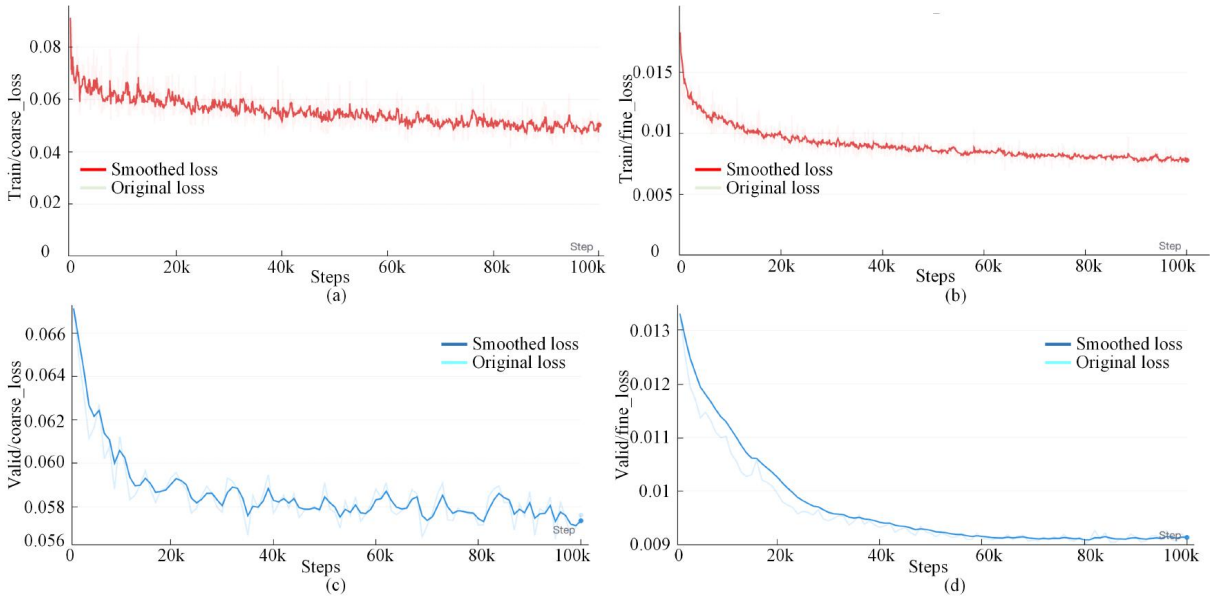

Figure 8. Visualization of the training process. EMD errors for coarse point cloud generated by the decoder in (a) training stage and (c) validation stage. CD errors for dense point cloud produced by the refiner in (c) training stage and (d) validation stage.

Additionally, to demonstrate the training process more vividly, we plot the learning curve of the proposed method VPC-Net, see Fig. 8 for illustration. The training losses and validation losses both consist of two different types of losses. One is the Chamfer Distance for generated coarse point clouds, while another is the Earth Mover's Distance for produced dense point clouds. From Figs. 8a and 8b, one can find out that the training losses gradually decreases as the number of training steps increases and converges until 100K steps. The validation losses are shown in Figs 8c and 8d, which also prove the proposed method VPC-Net converges at 100K training steps.

6. Results

6.1. Point completion on the ShapeNet dataset

For evaluating the performance of our proposed method in completing point clouds of synthetic models, we compare our approach against the following state-of-art methods on the ShapeNet testing data, including 3D-EPN [8], PCN [46], and TopNet [34]. 3D-EPN [8] is a typical volumetric completion method, which is trained on the large-scale synthetic dataset as well. PCN [46] is a pioneering method that completes partial inputs using point clouds directly, which conducted an end-to-end training through an auto-encoder. TopNet [34] is the newest end-to-end point cloud completion method. For a fair comparison, all methods are trained and tested on the same data for all experiments in this section. The size of the output point cloud and the ground truth is fixed to 16384 points. Quantitative and qualitative results are shown in Table 1 and Fig. 9, respectively.

Table 1 shows that our proposed VPC-Net outperforms other methods significantly. In this table, the value of CD and EMD metrics are scaled by 1000 and 100, respectively. Especially, we obtain relative improvement on the average CD value by 25.7% and the average EMD value by 14.6% over the second-best approach PCN. Note that the values of EMD are much higher than those of CD. The reason is that EMD is a one-to-one distance matching metric, whereas CD can be one-to-many correspondences between the points. Besides quantitative results, the visualization of point cloud completion results for different methods is shown in Fig. 9. Comparing the results generated by our method and by others, we can observe that VPC-Net can produce more uniform point clouds with more fine-grained details, while others fail to recover such structures. Particularly, from the below-up views, it is clearly seen that our method can preserve fine details in the completed results, such as the signboard on the roof of the taxi (see Fig. 9a, car spoilers (see Fig. 9b), and the antenna of cars (the last row (see Fig. 9d).

To better display the more specific performance, in Fig. 10, we visualize the extent of improvement of our results over the second-best approach PCN on CD and EMD for all instances in the test dataset. In this figure, the horizontal axis indicates different vehicles. The height of the blue bar represents the increased value of VPC-Net over PCN. The green curve is the error of PCN and the difference between the green curve and the blue bar is the error of VPC-Net. It can be seen that our proposed method

Table 1. Quantitative comparison (smaller value represents better performance) of our method against the state-of-arts methods on ShapeNet.

Methods	Mean Chamfer Distance per point (10^{-3})	Mean Earth Mover's Distance per point (10^{-2})
3D-EPN [8]	22.308	10.7080
PCN [46]	11.668	6.0480
TopNet [34]	13.765	9.6840
VPC-Net	**8.662**	**5.1677**

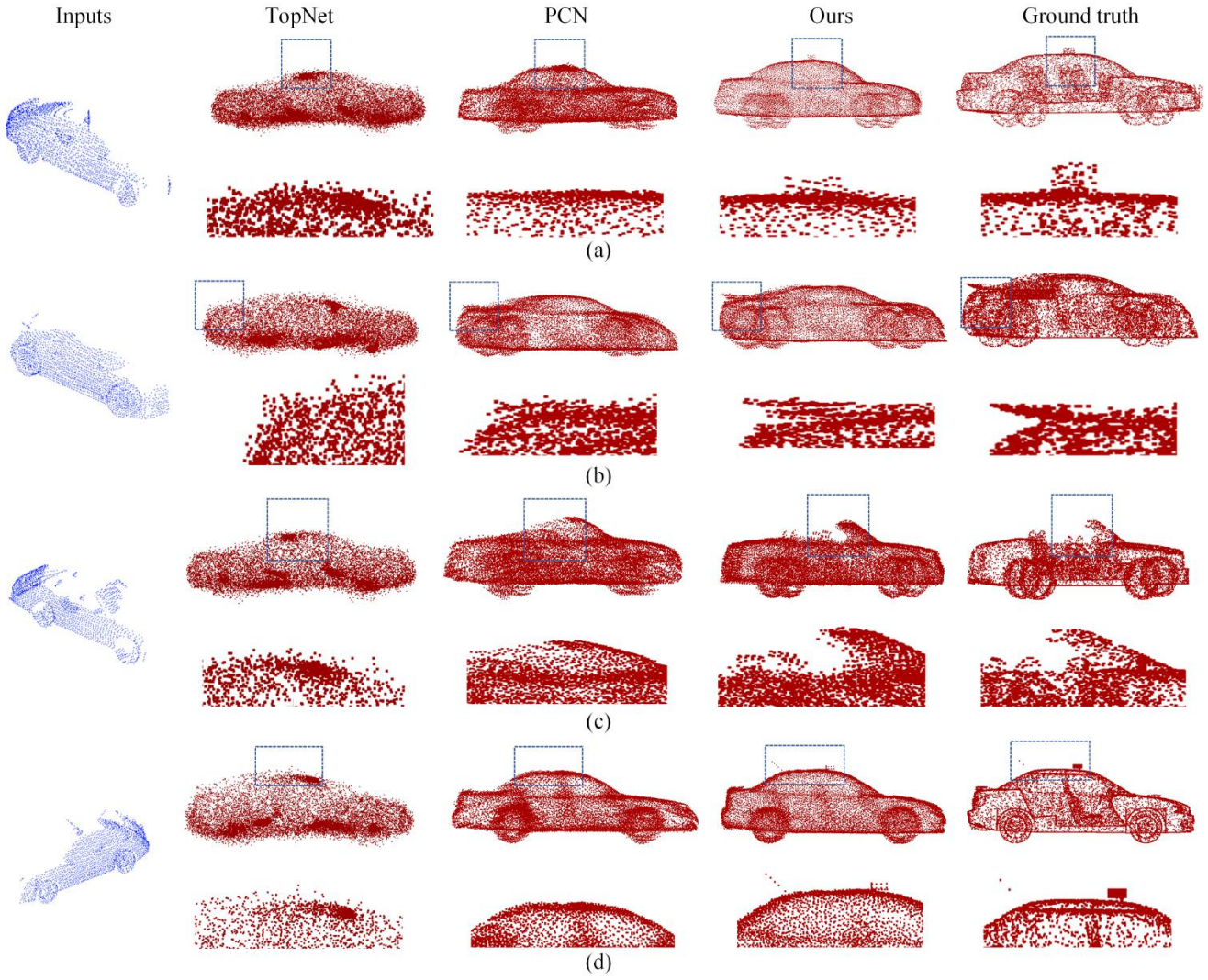

Figure 9. Qualitative comparison of our method against other state-of-arts methods on ShapeNet. (a)-(d) are four different testing examples. From left to right: input partial point clouds, TopNet [34], PCN [46], VPC-Net, and ground truth.

has significant improvement for the majority of shown instances. In addition, we can also find that our method achieves the greatest improvement in instances that PCN can provide highly noisy results, illustrating our method is able to handle these challenging examples where previous methods fail. Also, to demonstrate the points completed by VPC-Net are much more uniform than those generated by other baseline methods, three patches of spots in the same area produced by different methods are shown in Fig. 11. From the blown-up views of Fig. 11, we can see that both TopNet and PCN have heavily cluttered regions while our completion is more evenly distributed. We can also observe

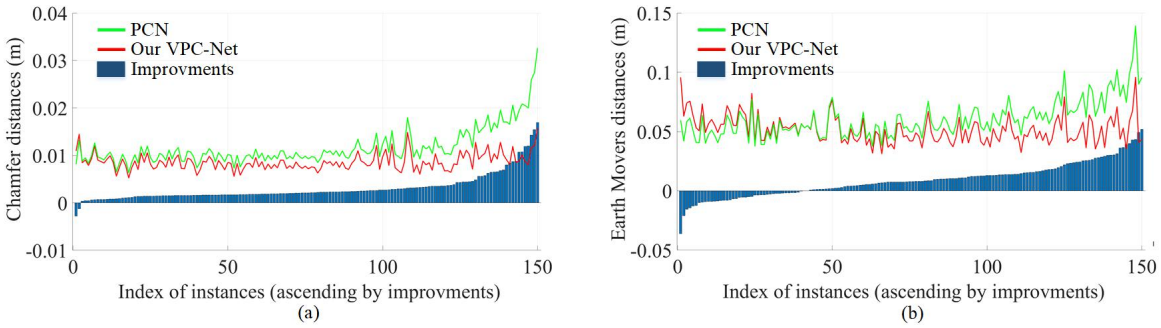

Figure 10. Comparisons between quantitative results of (a) PCN and (b) VPC-Net.

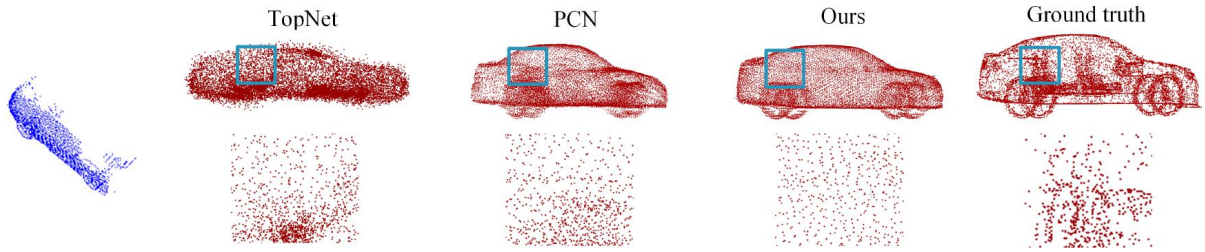

Figure 11. Example point cloud in the same area with different completion methods. From left to right: input partial point clouds, TopNet [34], PCN [46], VPC-Net, and ground truth.

that TopNet tends to generate several subsets of clustered points. Regarding PCN, it can not ensure the uniformity of local distribution of points. As a comparison, the global distribution of points in our outputs is more even than those of TopNet and PCN.

The uniformity and fine-grained details in our completions can be attributed to two factors: (1) We adopt two loss functions at two stages of the network. At the first stage of generating coarse point clouds, EMD loss forces the predictions to be uniform. Thus, the produced dense point clouds tend to be even since the following stage is trained to upsample the coarse predictions, despite using the CD as the loss. (2) The refiner first adopts the FPS method to sample the aggregated point cloud by concatenating the partial inputs and the dense outputs from the decoder and then using residual networks to refine it. This operation preserves the details of inputs and guarantees certain degrees of uniformity.

6.1.1 Point completion on the KITTI dataset

For evaluating the performance of our method on real scan LiDAR data, we test our method for point cloud completion on the KITTI dataset. Specially, we extract 2483 partial point clouds of cars from every frame based on their bounding boxes. Each extracted point cloud is transformed into the bounding box's coordinate system and then completed by our method trained on the ShapeNet dataset. Lastly, we turn them back to the world coordinates. It is considering that some extra noisy points from the ground or nearby objects within the cars bounding box, we remove the FPS operation in the refiner since it will bring these noises to the final completed results. Note that there are no ground truth point clouds in this dataset.

The qualitative results are shown in Fig. 12. We visualize the single frame raw data and choose five detected vehicles as the testing data, as shown in Fig. 12a. Fig. 12b shows five sparse and partial input point clouds, while Figs. 12c and 12d display the completed point clouds by PCN and our method, respectively. From Fig. 12, we can see that VPC-Net has better generalization capability and complete shapes that display the point sets are evenly distributed on the vehicle surface. Note that both networks are trained on the same ShapeNet training set and tested on KITTI. For example, for Example 2 in Fig. 12, the result generated by our method includes the details of missing parts and all points are more evenly distributed on the geometric surface, while point sets completed by PCN are messy and lose the detailed structured of the rear of the car. Besides, we can see that many points from PCN escaped the car surface, which can be observed in Example 3, Example 4, and Example 5.

Based on obtained outputs and comparisons, we can conclude from Fig. 12 that our method is robust to different resolutions of input point clouds, which is an essential char-

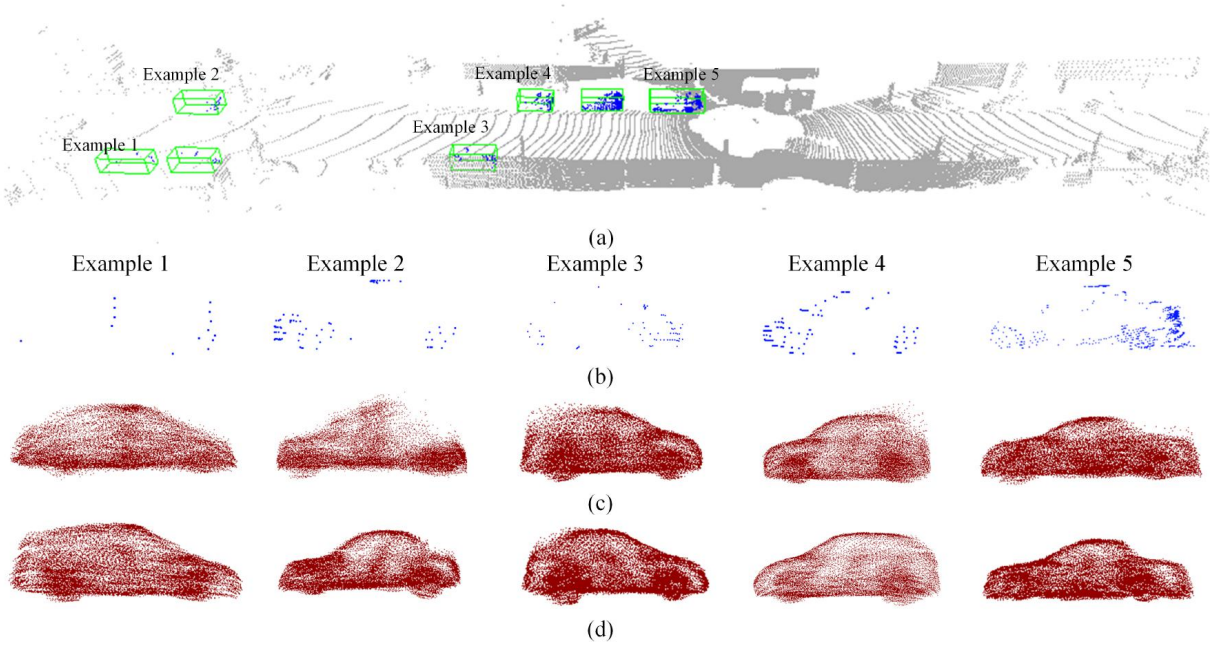

Figure 12. Completed 3D point clouds using real-scan data from the KITTI dataset. (a) Five detected vehicle examples in a single frame. (b) shows partial point clouds. (c) Completed point clouds by PCN [46] and (d) Completed point clouds by VPC-Net.

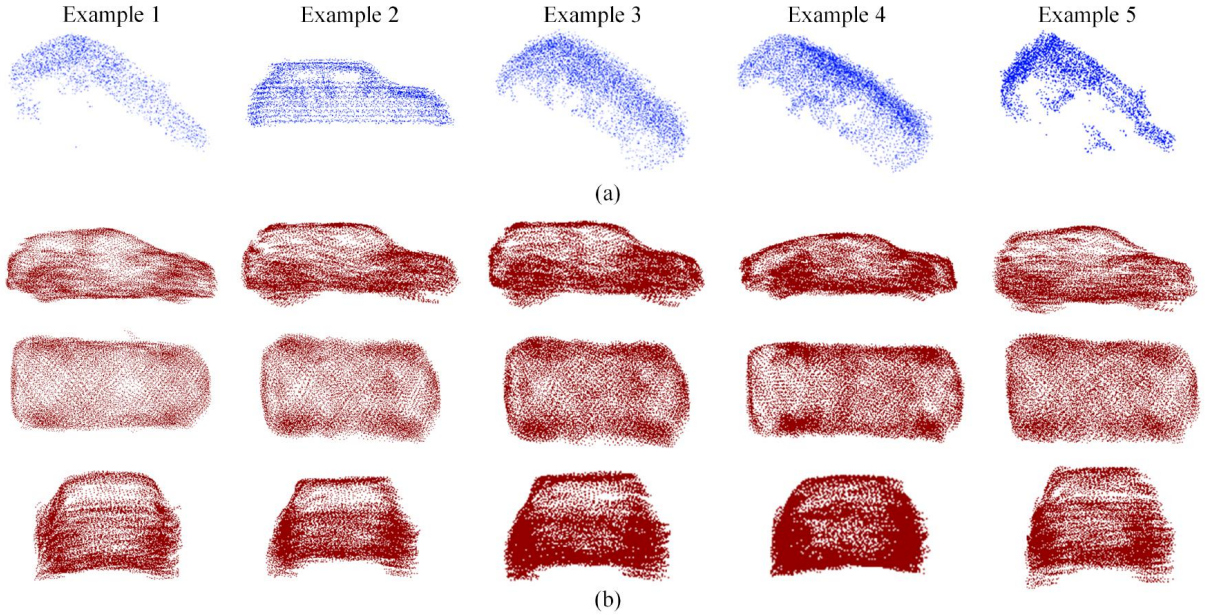

Figure 13. Completed 3D point clouds using real-scan data from the TUM-MLS-2016 dataset. (a) Five vehicle examples of partial point clouds. (b) Completed point clouds displayed from different viewpoints: side view, top view, and rear view.

acteristic for handing real scan data. For example, the input point clouds of Examples 1 and 3 have fewer than 100 points, while it includes more than 400 points in the case of Example 5. In spite of this, our method is able to produce uniformly, dense, and complete point clouds with finely detailed structures.

6.1.2 Point completion on the TUM-MLS-2016 dataset

To further illustrate our method's effectiveness and generalization ability on real scan data, we select the TUM-MLS-2016 dataset as a test set. We do not have complete point clouds as ground truth for the TUM-MLS-2016 dataset as well. Therefore, we select and show qualitative results of some vehicle instances in Fig. 13. Unlike point clouds from the KITTI dataset, point clouds from the TUM-MLS-2016 dataset are very dense. These partial point clouds contain 4200 points on average here. In spite of this, our method can still generate detailed information not only in the partial inputs but also for the missing structures. For example, in the fourth row of Fig. 13, the point cloud completed by our method preserves the shape of input and reconstruct the wheels and other missing parts. It verifies that our approach can transfer easily between the different distributions without any fine-tuning operations, whether partial point clouds are from the KITTI dataset, TUM-MLS-2016 dataset, or ShapeNet dataset.

6.2. Application

Apart from evaluating the effectiveness of the proposed method, actually, more complete and denser point clouds can be helpful or assistant for many common tasks [46]. In this section, we explored to apply the completed results into the 3D vehicle monitoring task. The proposed method VPC-Net can provide complete shape information of vehicles, which can be regarded as an assistant for this task. From another point of view, it also demonstrates that the proposed method is suitable for real-time applications. Note that we do not handle with existing issues in monitoring task using completed vehicles, the goal is to provide the shape of the vehicles for monitoring task only based on the existing raw LiDAR data.

Therefore, one Velodyne HDL-64E rotating 3D laser scanner is placed on the center of crossroads to collect the spatially dense and accurate 3D information. The round hole in Fig. 14a is the location of the LiDAR system. Then, the typical monitoring technique Simultaneous Localization and Mapping (SLAM) [4] is leveraged to estimate the vehicles in a 3D map while simultaneously localizing the object within it. The velocity, orientation, and trajectory of vehicles can be obtained using the SLAM method. However, it can not reconstruct the complete shape of moving vehicles, as shown in Fig. 14a. From Fig. 14a, it is seen that the brown point clouds represent a moving car passed by this LiDAR-based system, which formed a band shape. For such dynamic vehicles, we detected them from each frame's raw data and completed them by VPC-Net trained on the ShapeNet dataset. Figs. 14b-d show the completed vehicle appeared on this crossroads at continuous time T_1 , T_2 , T_3, and T_4 , respectively. As can be seen, the proposed method VPC-Net can be applied to the real-time 3D vehicle monitoring task. Besides, the completed point clouds have full-content information on vehicle models, which help the traffic management systems make the right decision [21].

Table 2. Performance comparison of the proposed method with different components. The mean Chamfer distance and Earth's Mover distance per point are reported, multiplied by 10^3 and 10^2 respectively.

STN	PFE	Refiner	CD	EMD
			11.668	6.0480
✓			8.922	5.1947
✓	✓		8.916	5.1777
✓	✓	✓	**8.662**	**5.1677**

7. Discussion

7.1. Visualization of completion details

To better gain further insights about the details of completion performance, we visualize the residual distance between corresponding points from the outputs of our method VPC-Net to the ground truth in Fig. 15. The ten different vehicles are from ShapeNet test data. This figure provides detailed information about which vehicle parts are completed correctly. Different colors encode the normalized distance between the corresponding shapes. From Fig. 15, it is clearly seen that the output point clouds completed by our proposed method recover the most of vehicle parts correctly. Besides, observing the red area in Examples 1,3,5 and 6, it turns out our method cannot capture the fine-grained details in terms of the roof of vehicles. However, from the perspective of human perception, it can be tolerated since humans tend to judge the object quality by global features and tolerate if some parts are slightly incorrect in shape or location [33].

7.2. Ablation study

The ablation studies evaluate the effectiveness of the different proposed components in our network, including spatial transform network (STN), point feature enhancement operation (PFE), and refiner. We developed four models: (1) the model without STN, PFE, and refiner, (2) the model with STN only, (3) the model with both STN and PFE, (4) the model with STN, PFE, and refiner. We use CD and EMD as the evaluation metric and the quantitative results of these models are shown in Table 2. All experiments are conducted on the ShapeNet dataset and the resolution of points is 16384. It is clearly seen that our full pipeline has the best performance. As shown in Table 2, with the proposed STN module, our model gets an improvement of 23.5 % and 14.1 % on CD and EMD respectively. This is because the rigid geometric transformation has a significant effect on extracting features from partial inputs, while STN can learn

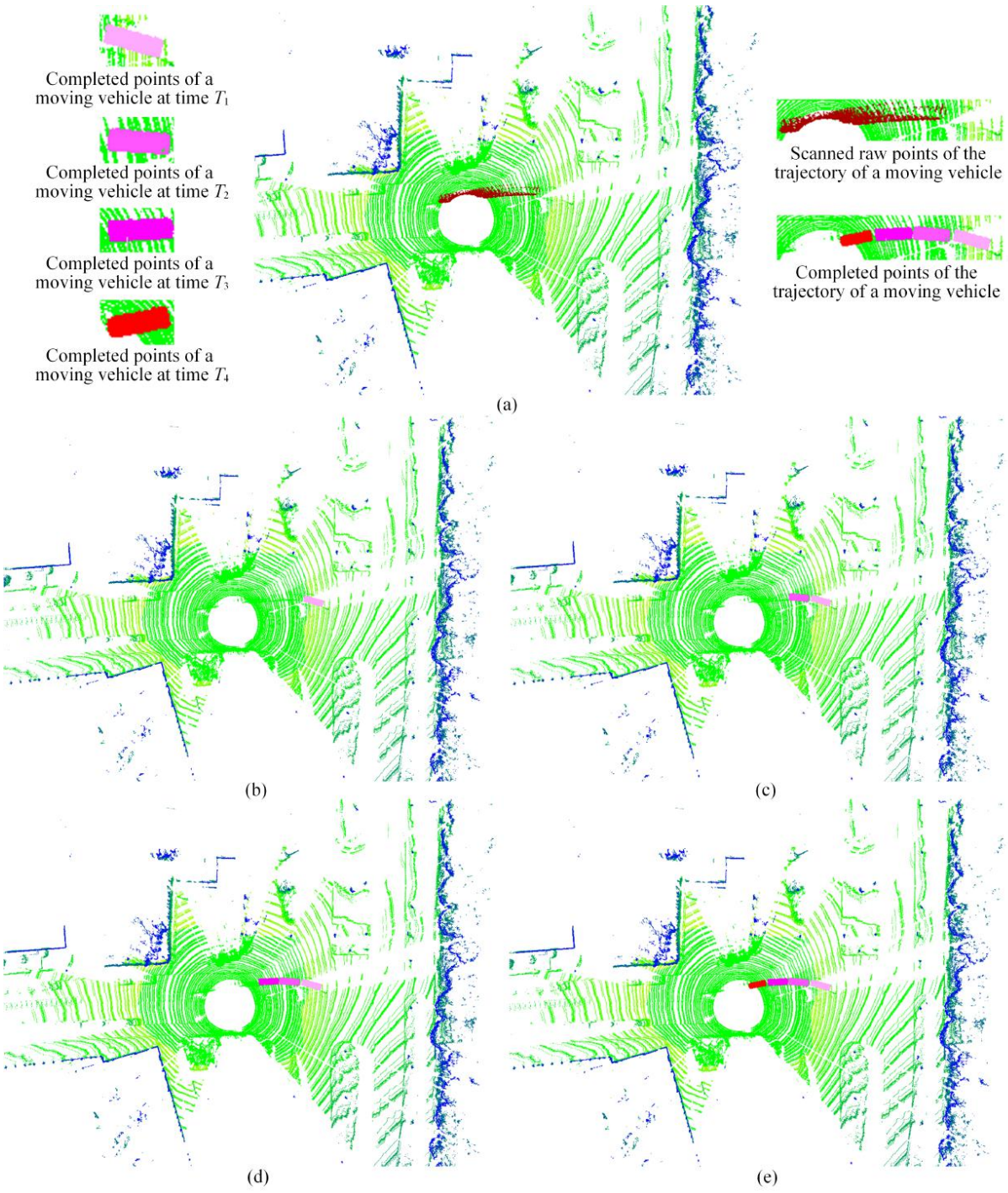

Figure 14. Application on 3D traffic monitoring. (a) 3D traffic scene at the crossroad visualized using SLAM technique. (b)-(d) Different colorful point clouds represent the completed point clouds by VPC-Net of this vehicle appears at different times.

invariance to translation and rotation. With the proposed PFE module, our model brings substantial improvements (0.1 %, 0.3 %) on CD and EMD. This is consistent with our expectation that enhancing the global feature is essential for determining to generate a more accurate coarse point cloud. The proposed refiner module can further improve the performance by 3 %, 0.2 % on CD and EMD. The improvement is especially significant on CD. This is because the re-

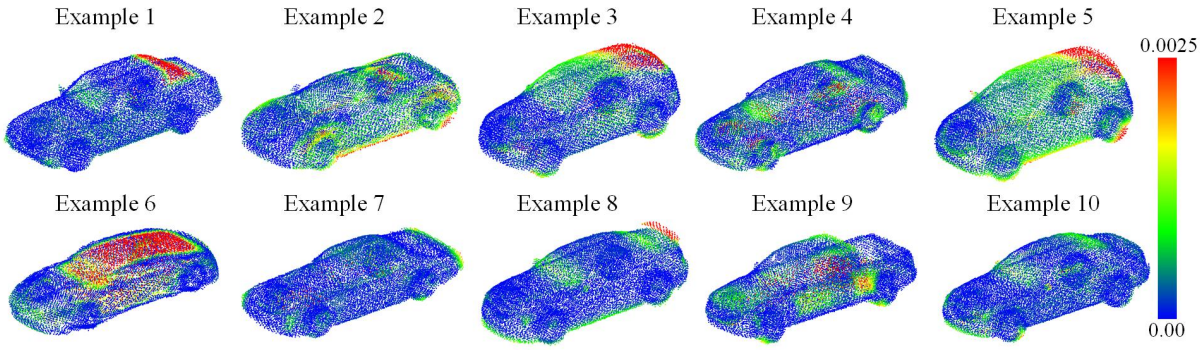

Figure 15. Visualizing point distances between the completed point clouds with ground truth point clouds.

Table 3. Quantitative results on inputs with different content of missing. The Chamfer Distance is reported by PCN and our method, multiplied by 10^3.

Visible Ratio	25%	40%	60%	80%
PCN [46]	21.555	13.979	12.002	11. 884
VPC-Net	**14.786**	**12.377**	**7.926**	**7.612**

finer actually improves the fine-grained details of completed point clouds, while CD is better to measure the global structure of objects than EMD. The ablation studies demonstrate that each proposed module plays significant roles in our network for performance improvements. Removing any modules will decline the performance, which proves that each proposed module contributes.

7.3. Robustness test

In robust tests, we carried out some experiments to evaluate our method's robustness for input point clouds with various missing degrees. First of all, the completeness statistics of test data from the ShapeNet dataset, KITTI dataset, and TUM-MLS-2016 dataset are collected, as shown in Fig. 16. We refer the overlap ratio R_o between input partial point clouds and completed point clouds as the completeness metric, which defined by:

$$R_o = S_p / S_c \tag{6}$$

where S_p and S_c are surface areas of input partial point clouds and completed point clouds, respectively.

As can be seen in Fig. 16, most of input instances from KITTI dataset are very sparse, whose completeness are less than 50 %. In contrast, the examples from the TUM-MLS-2016 dataset have enough completeness since the TUM-MLS-2016 dataset provides the aggregated point clouds, not the original scan data. Besides, the completeness of test data from the ShapeNet dataset is a normal distribution. From experimental results in Section 6, our method VPC-Net can handle with these inputs with different completeness.

To better illustrate the robustness performance, we perform the robustness test experiment on the ShapeNet test data since there are ground truth point clouds. Especially, we change the incompletion degree *d* of input point clouds, where *d* ranges from 20% to 75%. The qualitative and quantitative results are shown in Fig. 17 and Table 3 respectively. The visible ratios 0.25, 0.4, 0.6, and 0.8 mean that four incomplete inputs lose 75%, 60%, 40%, and 20% of the ground truth data, respectively.

As illustrated in Figure 17 and Table 3, we can draw three conclusions: (1) Our method is more robust than PCN when dealing with the high incompletion degree. For example, when the visible ratio is 0.25, our method is able to generate the general shape of the car, but PCN is failed. (2) With the more and more regions are missing, CD and EMD errors slowly increase. That implies our method is still robust when meeting the inputs with different incompletion degrees. (3) The outputs completed by both methods will be plausible when dealing with incomplete inputs with a large percentage of missing. For example, the car generated by our method becomes is a cabriolet, while the ground truth is a non-convertible car. However, this ambiguity is a common issue [9], because even for humans, it is difficult for us to know what this car is like based on just one wheel.

7.4. Registration Test

An even density and completeness are key factors for a successful registration between two point clouds [42]. Correspondingly, the registration result can also reflect the quality (e.g., evenness of point density or completeness of points) of the input point clouds [46]. Here, similar to the test conducted in the work of the baseline method PCN [46], we also conduct registration experiments between pairs of vehicle point clouds. By comparing the registration accuracy using incomplete and completed point clouds, it demonstrates the feasibility of the proposed vehicle point

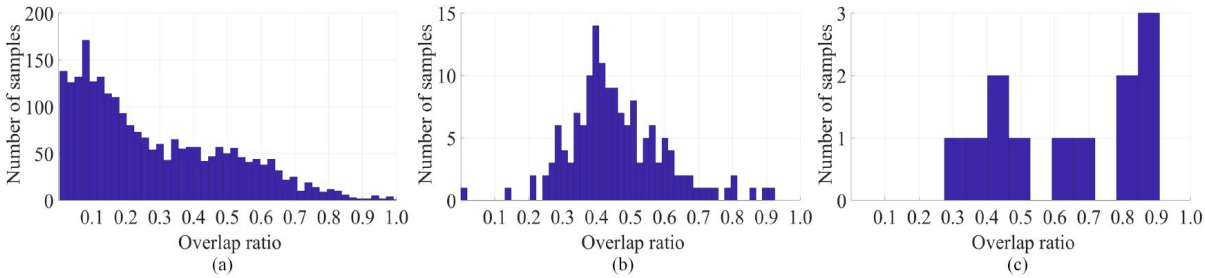

Figure 16. Completeness on tested datasets. Overlap ratio between input point clouds and completed point clouds in (a) KITTI dataset, (b) ShapeNet dataset, and (c) TUM-MLS-2016 dataset.

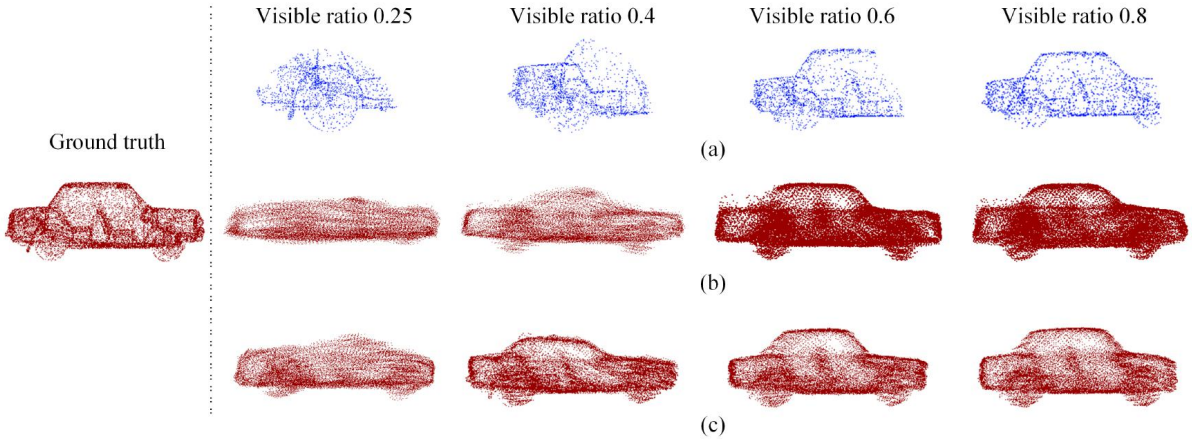

Figure 17. Qualitative results on inputs with different missing content. (a) Partial point clouds with different levels of visibility. Completed point clouds by (b) PCN and (c) our VPC-Net.

Table 4. Averaged rotation and translation errors of point cloud registration using different inputs.

Inputs	Average error	
	rotation (°)	translation (m)
Partial inputs	13.9422	7.0653
Complete inputs	**7.9599**	**4.2059**

cloud completion method. The vehicle point clouds of adjacent frames in the same Velodyne sequence from the KITTI dataset are chosen as test data. Specifically, we adopt two types of inputs to the registration method: one is partial point clouds from real-scan data, while another is completed point clouds by the proposed VPC-Net.

Besides, a simple point-to-point ICP [3] is applied as a registration algorithm, which minimizing distances iteratively between points from two point clouds. Notably, the ICP algorithm is not the only choice for registration tasks. Any registration algorithm that can be applied to illustrate the completed results have a good and consistent shape for the same vehicle instances in different frames. The average rotational and translational error on the registration results with partial and complete input point clouds are compared. The rotational error E_R and translational error E_T are defined as following respectively:

$$E_R = 2\cos^{-1}(2 < R_1, R_2 >^2 -1) \tag{7}$$

$$E_T = ||T_1 - T_2||_2 \tag{8}$$

where R_1 and T_1 are the rotation, translation of groud truth in the KITTI dataset, respectively. R_2 and T_2 are the rotation, translation measured by the ICP method, respectively.

As shown in Table 4, quantitative results demonstrate that using the complete point clouds generated by VPC-Net provides a more accurate estimation of translation and rotation than that of incomplete point clouds when used to conduct the registration test. Specifically, both rotation and translation accuracy are improved by 42.9% and 40.5%, respectively. In Fig. 18, ten qualitative examples are displayed. It can be seen that the completed point clouds have large overlapping regions recovered by VPC-Net, which demonstrates the VPC-Net can generate consistent shape with high quality for the same vehicle in different frames. We list the corresponding rotation and translation errors for

Table 5. Quantitative comparison of point cloud registration task with different inputs.

	Partial inputs		Complete outputs	
Example	Rotation error	Translation error	Rotation error	Translation error
1	4.5159	1.4715	**1.8219**	**0.5904**
2	11.4627	2.1093	**0.5678**	**0.1060**
3	4.5159	1.4715	**1.8219**	**0.5904**
4	143.9396	58.0907	**1.5606**	**0.7201**
5	178.6335	54.5161	**3.1471**	**1.5235**
6	14.8757	7.8894	**2.4544**	**1.2499**
7	3.1952	1.8321	**2.4083**	**1.3489**
8	1.7482	0.6973	**0.9957**	**0.2084**
9	**0.0270**	**1.3128**	5.5954	3.0927
10	**0.6646**	**1.3941**	4.1969	3.8149

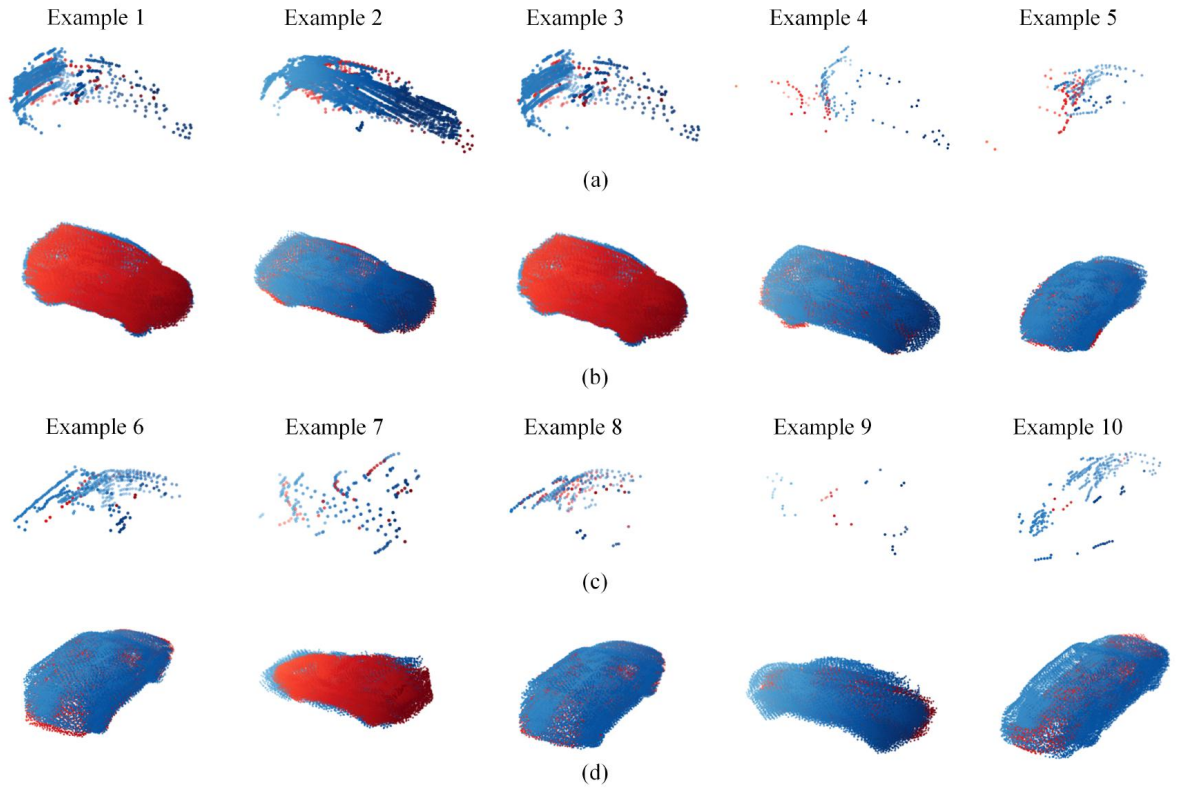

Figure 18. Qualitative comparison of point cloud registration task with different inputs. (a) and (c) Registered results with partial point clouds. (b) and (d) Registered completed results of the same examples.

these examples in Table 5. As can be seen from Example 1 to Example 8, the registration using complete point clouds shows an improvement in both rotation and translation accuracies. The improvement is most significant when the error with partial inputs is relatively large. Examples 9 and 10 are the failure cases that the registered partial inputs have better performance than registered complete inputs. However, we can get the reason from the qualitative results in Fig. 18, namely the registered partial inputs have too few points, only about ten points so that the ICP method is not able to compute the errors accurately.

8. Conclusion

In this paper, we propose a novel end-to-end network VPC-Net to vehicle points completion using the sparse and partial point clouds. Our method can generate complete and realistic structures, and keep the fine-grained details in an efficient manner. Besides, it is effective across different resolutions of inputs from various sensors. Experimental results on the ShapeNet dataset, KITTI dataset, and

TUM-MLS-2016 dataset demonstrate the effectiveness of our proposed VPC-Net outperfomes the state-of-art methods. Moreover, it has strong generalization performance on real-scan datasets, which makes it be suitable and beneficial for practical applications. The main benefits of the VPC-Net can be summarized as follows:

- The quality of point clouds has less influence on the VPC-Net, which indicates that the proposed method is more robust to the various resolution and missing degree of point clouds compared to other 3D point cloud completion network.
- The VPC-Net provides satisfying results for various datasets, which contributes to three aspects. The first one is the PFE layer, which combines the low-level local features and high-level semantic features. Besides, the spatial transformer network guarantees extracted features are invariant to rigid rotation and translation. The last one is the refiner module, which tends to preserve the fine details of input point clouds.

However, it is existing some limitations of the proposed method. For example, the designed refiner will grow the number of training parameters compared with previous point cloud completion network PCN. Considering the ambiguity of the completion at test time, in the future, we will explore to generate multiple plausible shapes and then assess the plausibility of several various completions. Besides, we will also investigate to complete other objects in an urban scene, such as buildings, traffic signs, road lanes, and so on.

Acknowledgments

This research is supported by the China Scholarship Council. This research was funded by Natural Science Foundation of China grant number U1605254. This work was carried out within the frame of Leonhard Obermeyer Center (LOC) at Technische Universitt Mnchen (TUM) [www.loc.tum.de]. We would like to thank Weiquan Liu and Qing Li for their suggestions and support.

References

[1] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005.

[2] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017.

[3] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

[4] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[5] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[6] A.-L. Chauve, P. Labatut, and J.-P. Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1261–1268. IEEE, 2010.

[7] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017.

[8] A. Dai, C. Ruizhongtai Qi, and M. Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5868–5877, 2017.

[9] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 605–613, 2017.

[10] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[11] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 216–224, 2018.

[12] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 85–93, 2017.

[13] G. Harary, A. Tal, and E. Grinspun. Context-based coherent surface completion. *ACM Transactions on Graphics (TOG)*, 33(1):1–12, 2014.

[14] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015.

[15] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, volume 7, 2006.

[16] G. Li, L. Liu, H. Zheng, and N. J. Mitra. Analysis, reconstruction and manipulation using arterial snakes. *ACM Transactions on Graphics (TOG)*, 29(6):152, 2010.

[17] Y. Li, X. Wu, Y. Chrysathou, A. Sharf, D. Cohen-Or, and N. J. Mitra. Globfit: Consistently fitting primitives by discovering global relations. In *ACM SIGGRAPH 2011 papers*, pages 1–12. 2011.

[18] O. Litany, A. Bronstein, M. Bronstein, and A. Makadia. Deformable shape completion with graph convolutional autoencoders. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[19] P. Mandikal and V. B. Radhakrishnan. Dense 3d point cloud reconstruction using a deep pyramid network. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1052–1060. IEEE, 2019.

[20] L. Nan, A. Sharf, H. Zhang, D. Cohen-Or, and B. Chen. Smartboxes for interactive urban reconstruction. In *ACM SIGGRAPH 2010 papers*, pages 1–10. 2010.

[21] Y. Pan, D. Wang, X. Shen, Y. Xu, and Z. Pan. A novel computer vision-based monitoring methodology for vehicle-induced aerodynamic load on noise barrier. *Structural Control and Health Monitoring*, 25(12):e2271, 2018.

[22] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas. Example-based 3d scan completion. In *Symposium on Geometry Processing*, number CONF, pages 23–32, 2005.

[23] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas. Discovering structural regularity in 3d geometry. In *ACM SIGGRAPH 2008 papers*, pages 1–11. 2008.

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[25] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem. Completing 3d object shape from one depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2484–2493, 2015.

[26] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, pages 234–241. Springer, 2015.

[27] R. Schnabel, P. Degener, and R. Klein. Completion and reconstruction with primitive shapes. In *Computer Graphics Forum*, volume 28, pages 503–512. Wiley Online Library, 2009.

[28] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu. Structure recovery by part assembly. *ACM Transactions on Graphics (TOG)*, 31(6):1–11, 2012.

[29] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1746–1754, 2017.

[30] D. Stutz and A. Geiger. Learning 3d shape completion from laser scan data with weak supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1955–1964, 2018.

[31] M. Sung, V. G. Kim, R. Angst, and L. Guibas. Data-driven structural priors for shape completion. *ACM Transactions on Graphics (TOG)*, 34(6):1–11, 2015.

[32] A. Tagliasacchi, M. Olson, H. Zhang, G. Hamarneh, and D. Cohen-Or. Vase: Volume-aware surface evolution for surface reconstruction from incomplete point clouds. In *Computer Graphics Forum*, volume 30, pages 1563–1571. Wiley Online Library, 2011.

[33] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019.

[34] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese. Topnet: Structural point cloud decoder. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 383–392, 2019.

[35] A. Tevs, Q. Huang, M. Wand, H.-P. Seidel, and L. Guibas. Relating shapes via geometric symmetries and regularities. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

[36] S. Thrun and B. Wegbreit. Shape from symmetry. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1824–1831. IEEE, 2005.

[37] S. Tuermer, F. Kurz, P. Reinartz, and U. Stilla. Airborne vehicle detection in dense urban areas using hog features and disparity maps. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(6):2327–2337, 2013.

[38] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018.

[39] C. Wen, Y. Dai, Y. Xia, Y. Lian, J. Tan, C. Wang, and J. Li. Toward efficient 3-d colored mapping in gps-/gnss-denied environments. *IEEE Geoscience and Remote Sensing Letters*, 17(1):147–151, 2019.

[40] S. Wu, H. Huang, M. Gong, M. Zwicker, and D. Cohen-Or. Deep points consolidation. *ACM Transactions on Graphics (ToG)*, 34(6):1–13, 2015.

[41] Y. Xia, C. Wang, Y. Xu, Y. Zang, W. Liu, J. Li, and U. Stilla. Realpoint3d: Generating 3d point clouds from a single image of complex scenarios. *Remote Sensing*, 11(22):2644, 2019.

[42] Y. Xu, R. Boerner, W. Yao, L. Hoegner, and U. Stilla. Automated coarse registration of point clouds in 3d urban scenes using voxel based plane constraint. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W4:185–191, 2017.

[43] Y. Yang, C. Feng, Y. Shen, and D. Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018.

[44] W. Yao, S. Hinz, and U. Stilla. Extraction and motion estimation of vehicles in single-pass airborne lidar data towards urban traffic analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(3):260–271, 2011.

[45] W. Yao, M. Zhang, S. Hinz, and U. Stilla. Airborne traffic monitoring in large areas using lidar data–theory and experiments. *International Journal of Remote Sensing*, 33(12):3930–3945, 2012.

[46] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737. IEEE, 2018.

[47] S. Zhang, C. Wang, Z. He, Q. Li, X. Lin, X. Li, J. Zhang, C. Yang, and J. Li. Vehicle global 6-dof pose estimation under traffic surveillance camera. *ISPRS Journal of Photogrammetry and Remote Sensing*, 159:114–128, 2020.

[48] Q. Zheng, A. Sharf, G. Wan, Y. Li, N. J. Mitra, D. Cohen-Or, and B. Chen. Non-local scan consolidation for 3d urban scenes. *ACM Transactions on Graphics (TOG)*, 29(4):94–1, 2010.

[49] J. Zhu, J. Gehrung, R. Huang, B. Borgmann, Z. Sun, L. Hoegner, M. Hebel, Y. Xu, and U. Stilla. Tum-mls-2016: An annotated mobile lidar dataset of the tum city campus for semantic point cloud interpretation in urban areas. *Remote Sensing*, 12(11):1875, 2020.