

Revisiting Point Cloud Completion: Are We Ready For The Real-World?

Supplementary Material

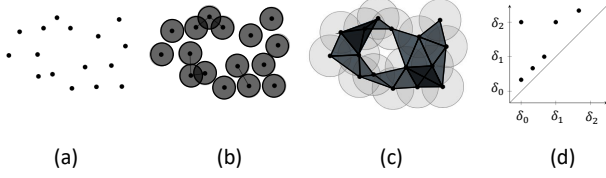


Figure 1. (a) to (c) Progression of filtration on a point cloud over different spatial resolutions as the distance threshold increases [10]. (d) Birth and death of k -dim topological features documented in the form of a persistence diagram, i.e., (b_i, d_i) pairs, so that each point corresponds to a homology which is born at b_i and dies at d_i .

1. Persistent Homology

In this section, we delve into the concept of Persistent Homology (\mathcal{PH}), an important tool in topological data analysis that systematically uncovers and quantifies the topological features of datasets. Persistent Homology combines ideas from algebraic topology, geometry, and computational mathematics to identify meaningful structures, such as connected components, loops, and voids, that persist across multiple scales. It is particularly valuable when applied to data in the form of point clouds or pixelated images, where traditional methods might struggle to capture structural and relational information.

At the core of this process lies the representation of a topological space as a cell complex—a combinatorial structure that encodes the relationships between points in the space. These cell complexes are constructed using simplices, which are the building blocks of higher-dimensional shapes. A 0-dimensional simplex is a point, a 1-dimensional simplex is an edge, a 2-dimensional simplex is a triangle, and so on, with higher-dimensional simplices being generalizations of these structures. Simplices are combined to form simplicial complexes, which generalize graphs to higher dimensions. Figure 1 provides a visual depiction of simplices of various dimensions and how they combine to form simplicial complexes. These structures serve as the foundation for applying homological methods.

Homology, in its simplest sense, is a branch of mathematics that provides a systematic way to analyze and classify the global properties of a topological space by examining its local features. Specifically, homology assigns algebraic invariants to a topological space, allowing us to identify and quantify k -dimensional holes, where k corresponds to the dimension of the feature being analyzed. For example: a 0-dimensional hole corresponds to a connected component of the space, a 1-dimensional hole corresponds to a

loop or cycle, a 2-dimensional hole corresponds to a void or cavity enclosed by a surface.

These holes, generalized across all dimensions, encapsulate the structural essence of a space. Figure 1 illustrates how homology captures these features, translating the geometry of a space into meaningful topological features [5].

Persistent Homology (\mathcal{PH}) extends classical homology by tracking how these k -dimensional features change as the dataset is viewed at different scales. This is accomplished by constructing a filtration, which is a sequence of nested simplicial complexes:

$$\phi \subseteq \mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \mathcal{C}_3 \dots \mathcal{C}_i \dots \mathcal{C}_n = \mathcal{C},$$

where ϕ is the empty complex, and \mathcal{C} is the final simplicial complex encompassing the entire dataset. Each step in this sequence corresponds to a specific scale or resolution. Persistent Homology identifies when topological features, such as connected components, cycles, or voids, are born and when they die as the filtration progresses. Features that persist across a wide range of scales are considered significant and indicative of meaningful structures within the data. Conversely, features that appear and disappear quickly are often interpreted as noise [5].

Filtration is the step-by-step process of building simplicial complexes by progressively adding simplices to the structure. The manner in which this process is defined depends on the type of dataset being analyzed. For example, in the case of point clouds, a filtration is often constructed using distances between points. At each stage of the filtration, new simplices are added based on a chosen threshold parameter, often denoted by α . The filtration progresses monotonically, meaning that each simplicial complex in the sequence contains all simplices from the previous step, along with any new simplices added at that stage.

For point clouds, a common type of filtration is the Vietoris–Rips filtration, which is defined based on pairwise distances between points. For a given value of α , an edge is added between two points if the distance between them is less than or equal to 2α . Higher-dimensional simplices, such as triangles and tetrahedra, are introduced when a set of points becomes fully connected. The progression of this filtration is depicted in Table 1, which illustrates how simplicial complexes evolve as α increases.

As the filtration progresses, topological features are born and die. These events are recorded as pairs (b, d) , where b is the scale at which the feature first appears (birth), and d is the scale at which the feature disappears (death). For example: - The addition of an edge may create a new 1-dimensional cycle, marking the birth of a feature. - Con-

versely, the addition of another edge may fill in that cycle, causing its death.

Consider the example of four points forming a rectangle. Initially, a 1-dimensional hole (cycle) is created when the rectangle is formed. When the diagonal edge is added, the rectangle is divided into two triangles (2-simplices), resulting in the destruction of the cycle. These birth-death pairs can be visualized using barcodes, where the length of each bar represents the persistence of a feature. Long bars correspond to significant features, while short bars typically represent noise [5].

When dealing with images, the process of constructing a filtration differs from that of point clouds. Instead of using pairwise distances, the pixel intensities of the image are used. The final simplicial complex \mathcal{C} corresponds to a triangulation of the image grid, with vertices representing pixels. Sub-level set filtrations are used, where the filtration function is defined as:

$$f((v_0, v_1 \dots v_n)) = \max_{i=0,1,2,3 \dots n} f(v_i),$$

which assigns each simplex a value equal to the maximum intensity of its vertices. Filtration begins with the minimum intensity value and gradually includes pixels with intensities less than or equal to α . As α increases, new simplices are added, and the filtration progresses. This allows the topological features of the image to be analyzed at multiple intensity levels.

Persistent Homology is a powerful framework that extracts meaningful structural information from complex datasets. By studying the persistence of topological features across scales, it provides insights into the underlying geometry and topology of the data. Its versatility makes it applicable to a wide range of domains, including shape analysis, image processing, and network analysis.

2. Evaluation Metrics

We evaluate the difference between model reconstructed complete PC and ground truth complete PC using the following metrics.

- **Chamfer Distance (CD):** It tries to capture the average mismatch between points in two given point clouds $X, Y \in \mathbb{R}^3$ and is given by:

$$d_{CD-L2}(X, Y) = \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 + \sum_{y \in Y} \min_{x \in X} \|x - y\|_2^2$$

$$d_{CD-L1}(X, Y) = \sum_{x \in X} \min_{y \in Y} |x - y| + \sum_{y \in Y} \min_{x \in X} |x - y|$$

- **Hausdorff Distance (HD):** If the distance between a point p on a surface S and a surface S' is given by:

$$d(p, S') = \min_{p' \in S'} \|p - p'\|_2$$

then the HD between S and S' is defined as:

$$d_H(S, S') = \max_{p \in S} d(p, S')$$

For the evaluation of our surface reconstruction task (Section 4.3 of the main paper), we use an average of $d_H(S, S')$ and $d_H(S', S)$.

3. Scene-Level Datasets

We provide a detailed description of the procedure for creation of **RealPC**. To create **RealPC**, as mentioned in Section 4.1 of the main paper, we use four open-source railway datasets: [4] by Hungarian State Railways acquired with a Riegl VMX-450 high-density mobile mapping system; [11] by Wuhan University, in which urban railway dataset was captured using Optech's Lynx Mobile Mapper System, rural railway dataset with MLS system equipped with HiScan-Z LiDAR sensors, and plateau railway dataset with Rail Mobile Measurement System (rMMS) equipped with a 32-line LiDAR sensor; [1] by SNCF Réseau, the French state-owned railway company; and a catenary arch dataset [12] given by Strukton Rail, captured using Trimble TX8 Terrestrial Laser Scanner (TLS). A few scenes from these datasets have been visualized in Figure 3 in the main paper.

We use four open-source scene-level railway datasets from different countries. These scene-level PCs are annotated into relevant sections such as vegetation, overhead cables, railway tracks, industrial support and transmission structures, ground, etc. For this work, we extract industrial structures from these scene-level PCs. As a result, we obtain a PC with multiple industrial structures as shown in Figure 4 in the main paper - Input to (A). Referring to the same figure, our methodology is divided into five main parts. In (A) we employ HDBSCAN [3], a hierarchical clustering algorithm, to cluster individual industrial structures. HDBSCAN works accurately for noisy and complex data and automatically detects the number of data clusters. In (B), a manual inspection is performed to extract industrial structures which can serve as good ground truth for supervised training. Finally, in (C), (D), and (E), we process these ground truth PCs in three different ways to induce uniform sparsity, non-uniform sparsity, and incompleteness as explained in detail ahead.

For making a partial PC (C), we pick a random viewpoint around a ground truth PC and remove N number of points that are farthest away from this viewpoint. Similarly for sparsifying the same PC non-uniformly (D), we again pick a random viewpoint but this time we assign a probability to each point, which is either proportional or inversely proportional to a point's cubed distance to this viewpoint. Then,

we sample N points based on these probability values. This is repeated for different values of N and for different viewpoints. For uniform sparsification (E), we randomly sample N points and then repeat for different values of N . Simultaneous to these steps, we perform manual noise removal. These three steps are conducted for all ground truth PCs to obtain a paired object-level training dataset.

Steps (B) and (C) have manual interventions. We want to highlight that these interventions make **RealPC** robust and accurate. Most point clouds extracted from industrial scenes are incomplete due to view occlusions present during acquisition. To ensure that all ground truth point clouds in **RealPC** are complete in all respects, we manually hand-pick the complete shapes (B). To group samples into different classes, manual inspection and labeling is known to be more accurate as compared to automated processes. We intensively study all available shapes and acquire domain knowledge of the dataset before manually categorizing the point clouds.

RealPC is a real-world object point cloud dataset generated using mostly scene-level parent datasets. Topological properties of **RealPC** are inherited from the parent datasets. Our proposed method exploits these properties using \mathcal{PH} priors.

A differentiating factor of **RealPC** is that - unlike existing datasets (which may or may not be simulated) that are captured in extremely controlled environments, **RealPC** is acquired in uncontrolled real-world multi-sensor industrial settings that have numerous factors of variation, noise and disturbance.

We also highlight in the main paper that several existing datasets (last paragraph - Section 2.1) though not simulated, are captured in extremely controlled environments - unlike real-world settings (e.g. industrial) that have numerous factors of variation, noise and disturbance. These datasets (including KITTI) do not consist of ground truth complete shapes and hence do not consist of partial-complete paired information.

Multiple category variations are present in synthetic datasets as well as in **RealPC**. Table 1 in the main paper demonstrates that the gap between synthetic and real-world data also exists at a per-class level. The gap in noise and non-uniformity of synthetic datasets v/s **RealPC** can be due to the difference between mathematical sampling and real-world acquisition techniques. Mathematical sampling done in case of synthetic data is mostly uniform, and non-noisy due to the smooth surfaces of CAD models. On the other hand, capturing real-world objects/environments using 3D sensors induces noise, non-uniformity at several levels, reasons ranging from sensor-related to environmental factors, which are not present in synthetic settings.

4. Non-Neural Methods

In subsection 4.3 of the main paper, we benchmark our **RealPC** and ShapeNet against three non-neural network-based tasks: (a) Simplification, (b) Surface reconstruction, and (c) Upsampling. Point cloud simplification reduces point density of a point cloud while preserving its salient features and the overall structure. This enables faster processing, lower storage costs, and efficient analysis for large-scale 3D data. We use Weighted Locally Optimal Projection (WLOP) [7] in the mentioned subsection, an enhancement of the parameterization-free denoising and simplification method known as Locally Optimal Projection (LOP) [9]. While LOP struggles with point clouds that exhibit non-uniform distributions, WLOP addresses this shortcoming by integrating locally adaptive density weights.

Surface reconstruction for point clouds aims to generate a continuous surface from discrete points, enabling the creation of complete 3D models essential for visualization, analysis, and manufacturing. An alpha shape [6] is basically made from a subcomplex of the Delaunay triangulation of a point cloud, with its refinement level ranging from a rough approximation to a highly detailed representation of the point cloud’s surface. For our comparison, we use a range of α values to create a mesh from different point clouds. Mesh visualizations shown in Figure 2 support the average high HD values in the case of **RealPC** as compared to ShapeNet (Table 2 of the main paper). The meshes formed from **RealPC** point clouds fail to capture the intricate details of the point cloud at any α value.

Point cloud upsampling increases the density of sparse point clouds, enabling finer surface details and improving the accuracy of reconstruction, analysis, and downstream processing tasks. We adopt an upsampling strategy [8] which uses an edge-aware method to compute normals away from the surface singularities, and then uses the data obtained to resample towards the said singularities with the help of a bilateral projector.

5. Topological Loss

5.1. Formulation

The topological loss function assists the completion process by ensuring that the model generates point clouds along the topological prior generated by $0\text{-dim } \mathcal{PH}$.

The coarse point cloud seeds at the decoder assist the topological module and are used as input for the topological loss. The output of the topological module consists of 0-dim topological features (*birth*, *death*) pairs. The loss function in Equation 1 below uses these pairs for adjusting the *birth* and *death* values of the 0-dim Homology features according to the required skeleton and the available components in the input incomplete point cloud.

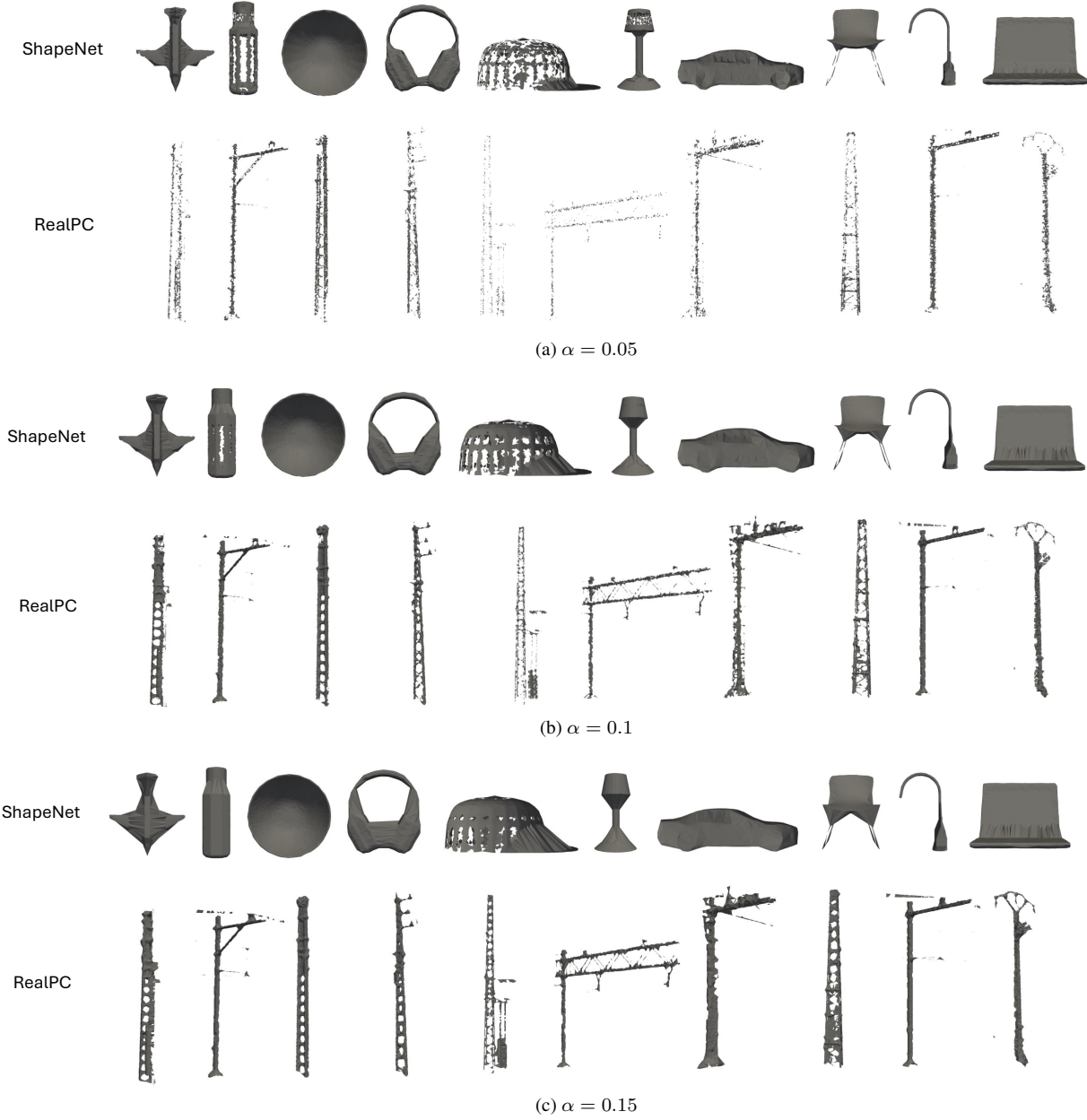


Figure 2. Surface reconstruction of some instances from different categories of ShapeNet and RealPC using alpha shapes.

$$\mathcal{L}_{homo} = \sum_{i=0}^n 1\{i > k\}(b_i - d_i) = \sum_{i=k+1}^n (b_i - d_i) \quad (1)$$

The parameter k holds significance here. As explained in the main paper, the input partial point cloud may consist of multiple disconnected components as demonstrated in Figure 6 in the main paper. In such cases, the above loss formulation ensures that the skeleton can have three skele-

ton components that can **(a)** cater to the three disconnected components and **(b)** merge together to form the final complete point cloud skeleton.

Setting the value of k requires manual inspection of the partial point clouds. In cases where the majority of point clouds have single components, setting $k=1$ works fairly well.

This formulation assists in the formation of the 0-dim \mathcal{PH} skeleton easily when the partial point clouds are not continuous (points are in disconnected clusters).

5.2. PH priors for KITTI scene understanding

We test these topological priors for scene reconstruction as well as completion on the real-world KITTI Odometry dataset (data). We cannot use **TopODGNet** and its modules (dictionary, seed generation, orthogonality constraint) are optimized for shape completion [2]. For KITTI scenes, we develop a generative model (inspired by DGCNN) and test it with and without 0-dim \mathcal{PH} priors. We use DGCNN [13] because it can easily be used for integrating topological priors as it satisfies the required criterion for integrating \mathcal{PH} priors. It transforms each point of a point cloud using Edge Convolutions into higher dimensions and retains the number of points across each layer. Therefore, sparse seeds point clouds of higher dimensions can easily be extracted from these layers and used for computation of the Vietoris Complex and persistence information. We use Sequence 08 for testing, 03 for validation, and the rest for training DGCNN with the KITTI dataset. The results for the experiments are shown in Table 1.

With respect to the KITTI-cars object level point cloud dataset, it consists of only partial point clouds, hence lacks paired information. Our model is designed on the assumption that paired information is available. We use a simple point cloud-based backbone to extract sparse to dense seed point clouds/backbone features, unlike existing works that use sophisticated and powerful backbones, but ours still works better on supervised real-world completion owing to 0-dim \mathcal{PH} features. We do not claim our work to be better than existing works in terms of sim-2-real transfer.

6. BOSHNet

The focus of **BOSHNet** is to extract a global topological backbone feature that is representative of the point cloud. The loss formulation ensures that all these significant topological features are combined into a single connected component - a single global backbone, instead of multiple disconnected components. Moreover, there are several high dimensional topological features (1-dim, 2-dim, etc.) that are richer representations of topological structures which still stay intact, because we work only with 0-dim \mathcal{PH} . Our method does not lead to the loss of important structural information or suppress significant topological features.

The Homology Sampler can be implemented in multiple ways (a): Direct application of the 0-dim \mathcal{PH} loss function on the complete point cloud for several epochs and extracting the updated point cloud after certain epochs as the sampled backbone ([1]), (b): Direct sampling of points from the entire surface of point cloud at different sparsity levels using a uniform distribution. We use the latter due to its time efficiency.

FPS-based sampling can also be used. We also present results with FPS sampling: CD-L2: 4.6, CD-L1: 64.6.

	Without \mathcal{PH} priors	With \mathcal{PH} priors
Scene Completion	1.18	1.04
Scene Reconstruction	1.50	1.18

Table 1. Benchmarking result for scene reconstruction and completion using the KITTI dataset using the Chamfer Distance \downarrow metric. All numbers are ($\times 10^{-3}$).

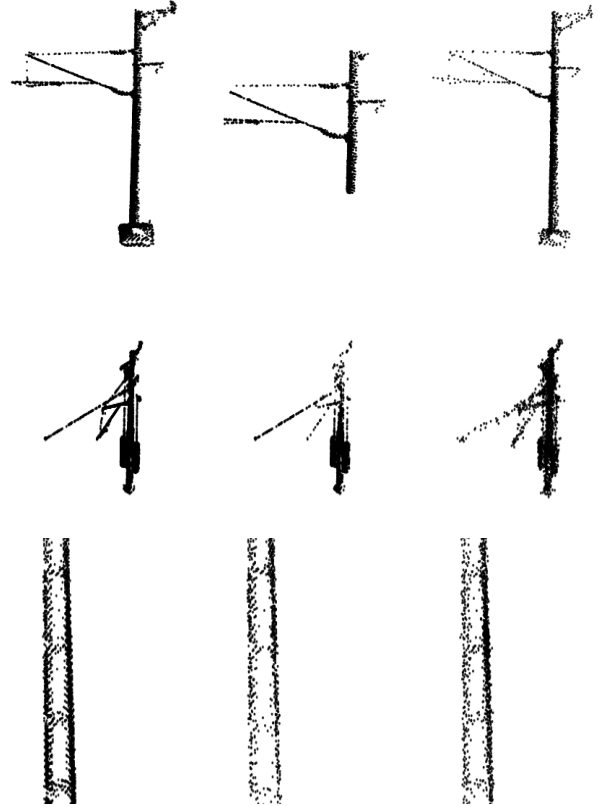


Figure 3. **Left:** Complete GT **Middle:** Partial Input to Homology Sampler based Model **Right:** Output. Our Homology Sampler model, on account of multiple 0-dim \mathcal{PH} priors is able to accurately reconstruct the incomplete PC.

7. Experimental Details

We provide the experimental details here. Our models are trained using an NVIDIA A100 GPU. For training the **TopODGNet** we use the same parameters and model structure as ODGNet [2]. We plug in the topology module on the sparse seeds generated at the decoder as demonstrated in Figure 6 in the main paper.

For Homology Sampler based network, we train the network for 1000 epochs with Adam optimizer using an initial learning rate of $5e-4$. We use a standard Point Cloud Autoencoder backbone based on PointNet.

We also provide the details of the computational training

costs before and after including \mathcal{PH} in the model. **Without** \mathcal{PH} : 6.96s/epoch; **With** \mathcal{PH} : 17.76s/epoch.

8. Analysis

We discuss the results of **BOSHNet** and highlight the reasons why **BOSHNet** works so well on the task. (a) The multiple proxy backbone point clouds extracted at gradual sparsity levels (sparser to denser) imitate a curriculum strategy. **BOSHNet** is taught with examples of increasing difficulty. It first learns to work well with extremely hard sparse backbone point clouds and reconstructs them well, followed by easier denser point cloud backbones. The initial hard examples help the model to work well with easier samples later. (b) **BOSHNet** has access to these proxy backbones as inputs to the model right from the beginning, while **TopODGNet** needs to *learn* to generate the seeds, which in turn help to generate the backbones (happens during later phases of training). This allows more robust learning for **BOSHNet**, right from the start. Sparse point cloud force **BOSHNet** to learn local structure robustly in absence of sufficient nearby points.

9. Demonstrations

We visually demonstrate nine point clouds from our dataset in Table 2 and 3. These demonstrate the non-uniform sparsity and noise that are natural and intrinsic characteristics of real-world datasets.

We demonstrate the Persistence Diagrams of these nine point clouds in Figure 4. For most of the point clouds we observe non-trivial persistence (most points are far from the diagonal). This indicates that real-world point clouds captured in challenging settings exhibit significant zero and one-dimensional topological features which are absent in synthetic point clouds (refer to Table 1 in the main paper). These persistence features are not observed in the persistence diagram of the synthetic datasets as shown in Table 5. Almost all the 0-dimensional topological features (indicated by dot) are along or very close to the diagonal for the synthetic datasets (Table 5) as opposed to **RealPC** (Table 4).

We also show some more examples of **BOSHNet** completions using **RealPC** in Figure 3. **BOSHNet** picks up precise topology of the point clouds and generates decent complete point clouds.

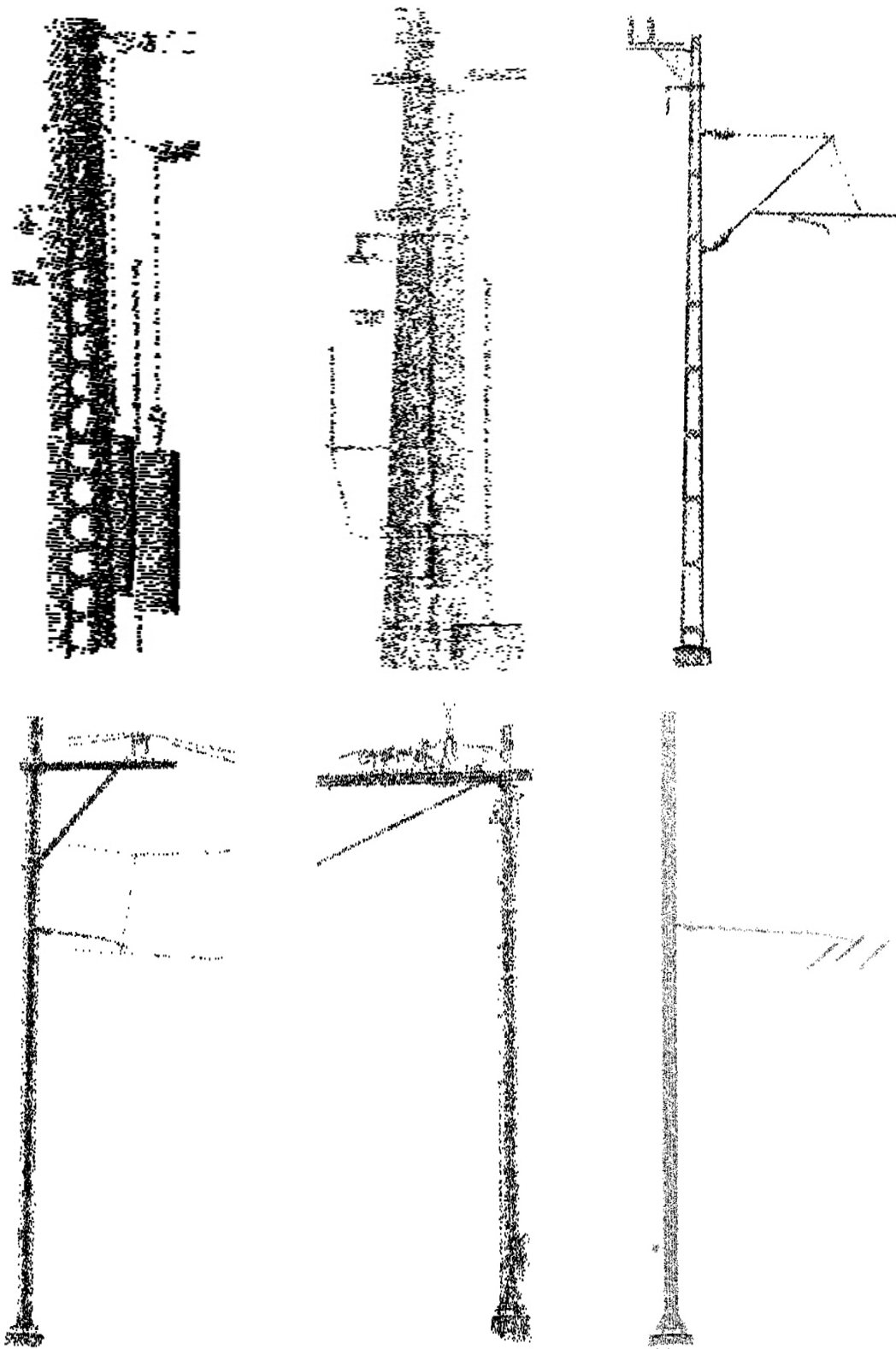


Table 2. Visual Demonstration of some examples from our dataset. For a video demonstration please refer to the video in the supplementary.

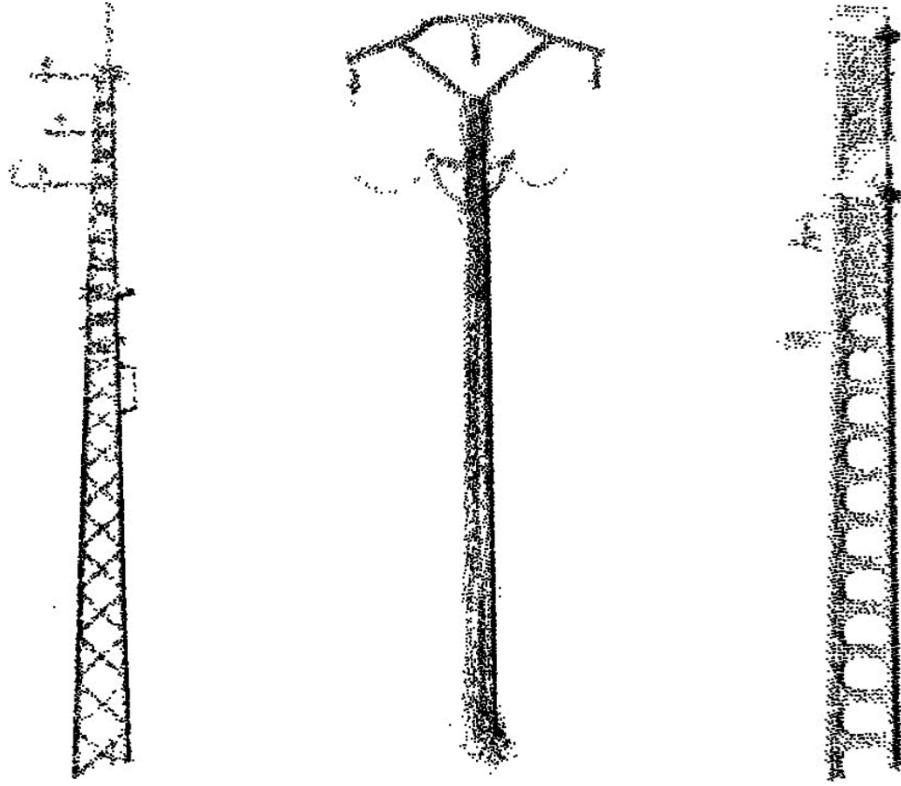


Table 3. Visual Demonstration of some examples from our dataset. For a video demonstration please refer to the video in the supplementary.

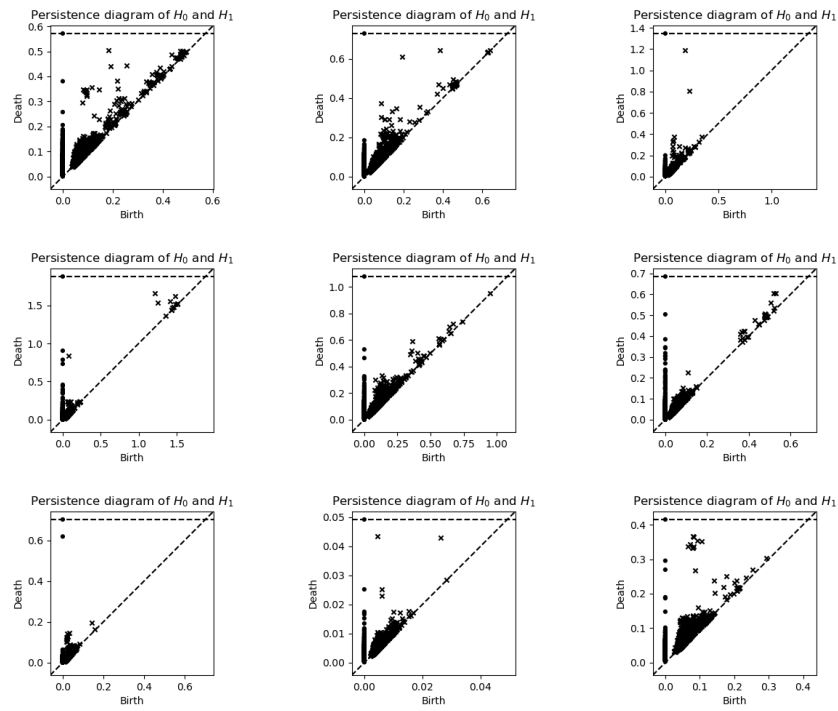


Table 4. Persistence Diagram of nine point clouds (of Table 2 and 3) of the **RealPC** dataset.

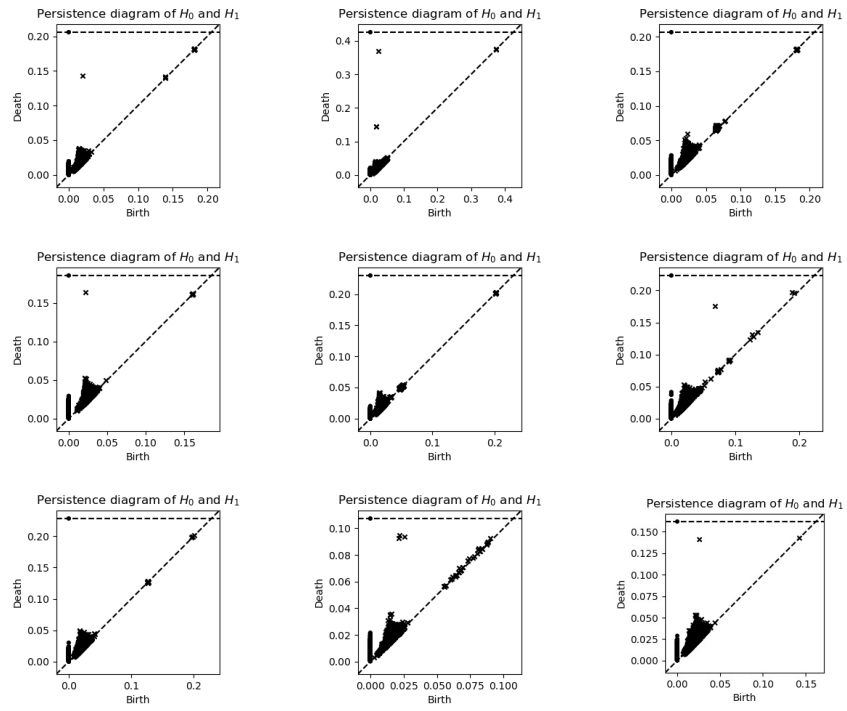


Table 5. Persistence Diagram of nine point clouds from the synthetic datasets - ShapeNet and PCN.

References

- [1] SNCF Réseau railway dataset. <https://ressources.data.sncf.com/explore/dataset/nuage-points-3d/table/>, 2017. 2
- [2] Pingping Cai, Deja Scott, Xiaoguang Li, and Song Wang. Orthogonal dictionary guided shape completion network for point cloud. In *AAAI*, 2024. 5
- [3] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013. 2
- [4] M Cserep. Hungarian mls point clouds of railroad environment and annotated ground truth data. *Mendeley Data*. DOI: <https://doi.org/10.17632/ccxpzhx9dj>, 1, 2022. 2
- [5] Herbert Edelsbrunner. Computational topology an introduction, 2008. 1, 2
- [6] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on information theory*, 29(4):551–559, 1983. 3
- [7] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM transactions on graphics (TOG)*, 28(5):1–7, 2009. 3
- [8] Hui Huang, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri Ascher, and Hao Zhang. Edge-aware point set resampling. *ACM transactions on graphics (TOG)*, 32(1):1–12, 2013. 3
- [9] Yaron Lipman, Daniel Cohen-Or, David Levin, and Hillel Tal-Ezer. Parameterization-free projection for geometry reconstruction. *ACM Transactions on Graphics (ToG)*, 26(3): 22–es, 2007. 3
- [10] Michael Moor, Max Horn, Bastian Rieck, and Karsten Borgwardt. Topological autoencoders. In *International conference on machine learning*, pages 7045–7054. PMLR, 2020. 1
- [11] Bo Qiu, Yuzhou Zhou, Lei Dai, Bing Wang, Jianping Li, Zhen Dong, Chenglu Wen, Zhiliang Ma, and Bisheng Yang. Whu-railway3d: A diverse dataset and benchmark for railway point cloud semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 2024. 2
- [12] B Ton. Labelled high resolution point cloud dataset of 15 catenary arches in the netherlands. *4TU Res. data, The NetherlandsTech. Rep*, 2022. 2
- [13] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 5