Supplementary of GLiDR: Topologically Regularized Graph Generative Network for Sparse LiDAR Point Clouds

Prashant Kumar[1] Kshitij Madhav Bhat[2] Vedang Bhupesh Shenvi Nadkarni[3] Prem Kalra[1]

[1]IIT Delhi [2]IIT Indore [3]BITS Pilani

prashantk.nan@gmail.com kshitijmbhat@gmail.com vedang.nadkarni@gmail.com pkalra@cse.iitd.ac.in

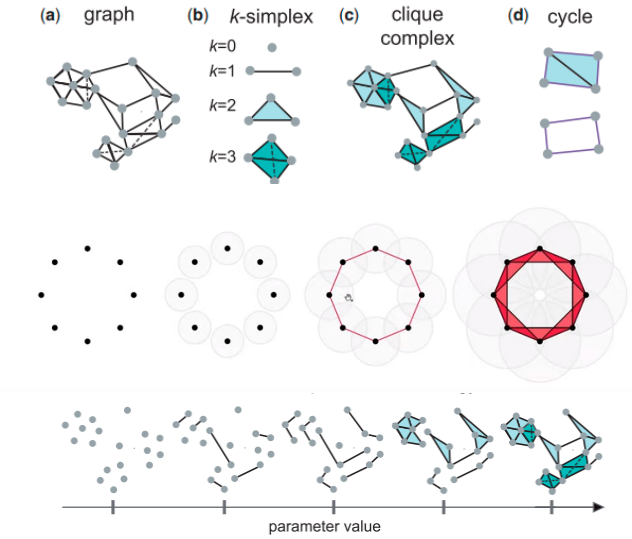


Figure 1. **Row 1** denotes k-simplexes and simplicial complexes. These are generalizations of graphs to higher dimensions. **Row 2**: $\mathcal{PH}$ discovers the global shape of a dataset. **Row 3** denotes the progression of filtration on a point cloud. As the filtration progresses, new simplicies are added to the simplicial complex. Image taken from Lia *et al*. [8].

1. Persistent Homology

We provide a detailed description of Persistent Homology ($\mathcal{PH}$) and its application on point clouds and images. A topological space can be encoded as cell complexes - a collection of k-dim simplicies ($k = (0, 1, 2...)$ (Figure 1 - row 1). *Homology*, an algebraic invariant of a topological space, uses local computation to capture global shape information (k-dim holes) of a topological space (Figure 1- row 2 and 3). These holes, generalized to various dimensions, form the basis of homology [1].

Persistent Homology ($\mathcal{PH}$) is an algebraic method to discover topological features of datasets. It converts a dataset (e.g., point cloud) to a simplicial complex and studies the change of homology across an increasing sequence of simplicial complexes $\phi \subseteq \mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \mathcal{C}_3....\mathcal{C}_i....\mathcal{C}_n = \mathcal{C}$, known as filtration [1]. Topological features are computed at different spatial resolutions across the subsequence. Examining the persistence of the features over a range of scales reveals insights about the underlying patterns in datasets. For point clouds, the filtration is defined on the edges of the complex. We define a sub-level filtration over $\mathcal{C}$. Every simplicial complex in the subsequence can be mapped to a number using a filtration function $f((v_0, v_1...v_n)) = \max_{i<j; i,j \in 0,1,2,3...n} f(v_i, v_j))$. This filtration, known as flag filtration, is based on the pairwise distance between points and is monotonic (every subsequent simplicial complex has a value higher than the previous).

For example - given a set of points in a 3D space, the filtration can be generated by an increasing α-neighbourhood ball for each point (See Table 1). For the given α, two balls intersect when two points are no further apart than distance 2α. At this moment, the two points are connected by an edge. For a given α, we connect all points that are no further than 2α. Assuming k edges are introduced at a given α, an ordering is given to these edges (i.e., the edges appear in that sequence). Further, the introduction of an edge (1-D simplex) can lead to the creation of a higher-order simplex (triangle, tetrahedron, and so on). These are also a part of the filtration - lower dimension simplicies are added to the filtration before the higher dimensions. The progress of filtration leads to the construction or destruction of homology (k-*dim* holes - connected components, cycles, voids, and so on) based on the following principles-

- The appearance of an edge leads to the creation of a k-*dim* hole that is not part of an existing in the previous sequence. This leads to the birth of a feature.
- The appearance of an edge leads to the completion of a k-*dim* hole that was discovered previously (i.e., the k-*dim* hole is completely filled with equal and lesser dimension simplicies). This leads to the death of the feature that was born at the birth of the hole. For example - the occurrence of four points connected to form a rectangle creates a *1-d* hole (cycle). When the diagonal edge is introduced at a

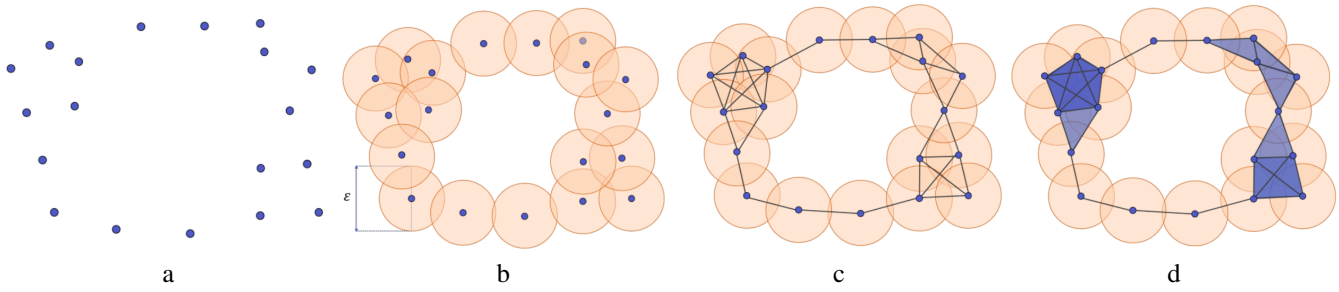


Table 1. Left to Right. Progression of filtration on a point cloud over different spatial resolutions. Image taken from Wright *et al.* [10].

later time in the filtration (for a higher value of α), it leads to the rectangle filled with two triangles (2-simplex). This leads to the death of the *1-d* hole feature.

Each 1-dim hole appears at a particular value of α and disappears at another value of α. The addition of an edge either creates or destroys a homology. For our case, given a set of points in 2D space (a point cloud or an image with pixel values), we do not know what $\alpha's$ to use to obtain the most important features. Therefore, we consider all values and observe the change in homology for different values of α. This leads to a nested sequence of increasing subsets of simplicial complexes referred to as a filtration. Each hole appears at a particular value of α (b) and disappears at another value of α (d). The persistence of the hole is represented as a birth-death pair (b, d). Visualizing all such pairs for all holes in the form of bars leads to a *barcode*. Short bars might represent noise, while long bars represent important features.

Unlike point clouds, images do not have a concept of pairwise distance. Instead, the intensity value of the pixels of an image help to create a filtration. For images, the final simplicial complex, $\mathcal{C}$ can be taken to be the triangulation of the image grid where the intersection refer to the pixels of the image. Like point clouds, sub-level set filtrations are used here. The filtration function for the sublevel filtrations is $f((v_0, v_1 ... v_n)) = \max_{i=0,1,2,3...n} f(v_i))$, which translates to the maximum intensity value of a pixel in a simplicial complex. The only difference is - instead of using pairwise distance for generating sub-level set filtrations, we use the pixel intensity values. α is initialized to the minimum intensity value. As it increases, the sub-level set increases, and more pixels with intensity $\leq \alpha$ are included in the filtration. New simplicies are added to the existing complex, and the filtration progresses.

2. Paired Scan Generation

We have described the method for generating correspondence pairs for datasets already existing in the literature. Assume we have a sequence of scans K = $\{k_i$: 1,2,3...$\}$. We generate pairs K_D, K_S — K_D is the set of dynamic, and K_S denotes the static scans by utilizing semantic segmentation information.

We divide the LiDAR scan and the range image into eight sectors. We identify a source and a target sector. Dynamic objects from the source sector are introduced into the target sector. We may have multiple target sectors depending on the availability of probable regions where new objects can be augmented. We depict the division of the LiDAR into eight sectors in Figure 2a. We further demonstrate the augmentation process for a correspondence pair generation in Figure 2b.

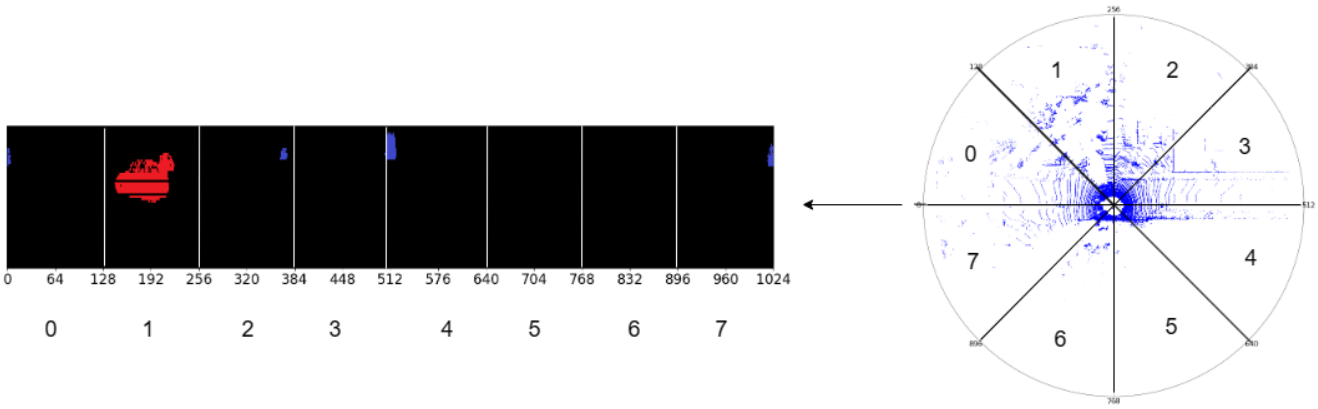
We do not claim that the augmented generated dynamic LiDAR would be totally realistic and follow all rules of the road. For instance, in a certain case with two vehicles: one being the already present and the other being the augmented one in the opposite lane of the AV, both may appear to overlay over each other. However, the data is sufficient to train **GLiDR** accurately. These issues are minor, and a large percentage of LiDAR frames are augmented with realistic dynamism at realistic locations. This is also demonstrated by the effect of augmentation of static structures on SLAM performance as shown in Table 2, 4 and Figure 3.

3. Segmentation mask generated by GLiDR

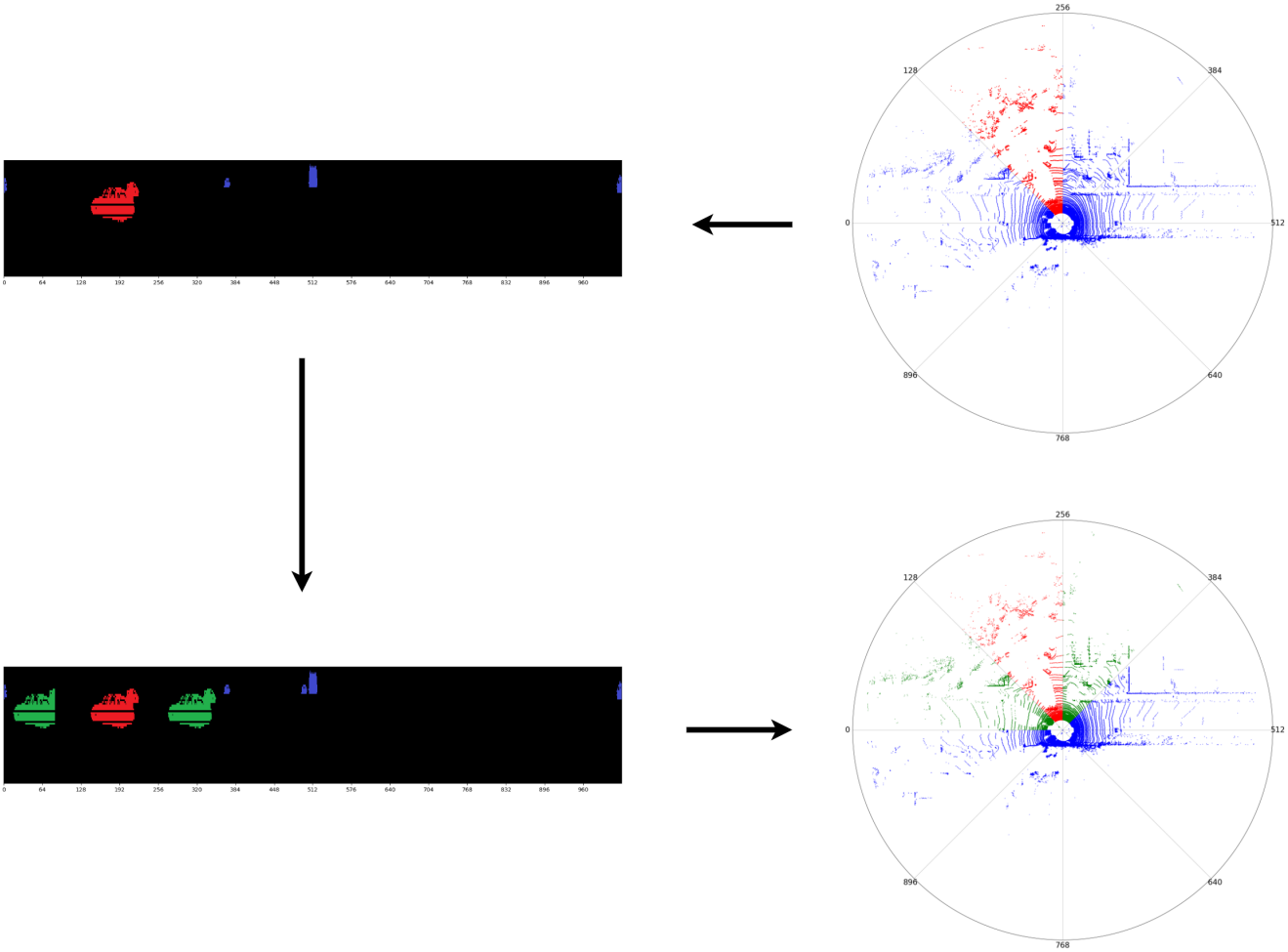
We demonstrate the segmentation masks for dynamic and static objects generated by **GLiDR**. We provide a video in the Supplementary material that demonstrates the accuracy of the binary segmentation mask generated by **GLiDR**. We remove the ground points from the LiDAR before generating the mask. Ground points are not useful for navigation and are also removed by SLAM algorithms before navigation. We classify all the above-ground points into static and dynamic objects. Dynamic objects - whether moving or stationary are marked with blue, while static objects are marked with red.

4. Baselines

CP3 Xu et al. [13] present an architecture for shape completion which is inspired by the prompting methods



(a) A LiDAR scan is divided into eight sectors that translate to corresponding regions in the range image segmentation mask.



(b) View in the clockwise direction from top right. We identify the source sector (highlighted in red in the LiDAR scan) and extract the segmentation mask (also in red) in the range image mask. The source sector mask is inserted in target sectors (0 and 3 - indicated by green) in the new mask. Overlaying the new mask over the LiDAR scan augments the LiDAR with dynamic objects (in the sectors indicated by the green.)

Figure 2. The corresponding pair pipeline results in the creation of a static-dynamic pair for KITTI. Static scan - original scan with dynamic object removed using segmentation mask. Dynamic scan - the output of the pipeline with the augmented new dynamic objects.

in NLP. The approach involves a Pretrain-Prompt-Predict paradigm, treating point cloud generation and refinement as the prompt and predict steps. They use IOI pretraining to achieve higher robustness in point cloud generation by

learning a pretext task in a self-supervised manner. The authors also introduce a novel Semantic Conditional Refinement (SCR) at the prediction step to discriminatively modulate refinement with semantic guidance.

Coase Net Wang et al. [12] is a two-stage approach for dynamic to static image translation applicable for 2D images. The authors formulate the problem as an image inpainting problem and aim to mitigate the loss of detail in the original static reconstructions. In the initial stage, a basic encoder-decoder network is used to generate a coarse representation of the static image, with which dynamic objects and their shadows are identified. In the second stage, the missing static pixels in the estimated dynamic regions are recovered based on their coarse predictions, which are improved by a mutual texture-structure attention module, allowing dynamic regions to incorporate textures and structures from distant areas with similar contexts.

Topological Autoencoders Moor et al. [7] introduces an approach of constraining autoencoders to preserve topological structures from the input space in the latent space of the autoencoder. The authors employ persistent homology (with the Vietoris-Rips complex) to formulate a differentiable loss function considering the topological signatures of both input and latent space. This approximation of the persistent homology loss calculations is combined with backpropagation on the level of mini-batches.

DSLR Kumar et al. [5] tackles the challenge of translating dynamic LiDAR data into static representations using range images. They employ an adversarial training approach involving an autoencoder consisting of a pre-trained generator and a discriminator network to produce static reconstructions of dynamic scans. This approach is further extended to work in a real-world setting using Unsupervised Domain Adaptation. Furthermore, the authors also introduce a variation that enhances the quality of reconstruction by integrating segmentation information into the process.

MOVES Kumar et al. [6] employs a GAN-based adversarial model that segments out movable and moving objects in LiDAR scans. The network consists of a generator and a discriminator, that is coupled with a contrastive loss on a LiDAR scan triplet using hard-negative mining. The authors also introduce an approach that integrates Unsupervised Domain Adaptation into the network for datasets lacking static-dynamic correspondence. This method integrates the domain discrepancy loss between paired and unpaired latent space domains.

5. LiDAR Reconstruction Evaluation Metrics

We evaluate the difference between model reconstructed static and ground truth static LiDAR scan using the following metrics.

- **CD:** Chamfer Distance tries to capture the average mismatch between points in two given point clouds $S_i, \overline{S}_i \in \mathbb{R}^3$ and is given by:

$$d_{CD}(S_i, \overline{S}_i) = \sum_{x \in S_i} \min_{y \in \overline{S}_i} ||x-y||_2^2 + \sum_{y \in \overline{S}_i} \min_{x \in S_i} ||x-y||_2^2$$

- **JSD:** Jenson Shannon Divergence is a measure of the distance between two empirical distributions P and Q, defined as follows.

$$JSD(P||Q) = \frac{1}{2}(D(P||M) + D(Q||M))$$

where $M = \frac{1}{2}(P+Q)$ and $D(\cdot||\cdot)$ represents the Kullback-Leibler-divergence between the two marginal distributions

- **MMD:** Suppose $x \in S_i$ and $y \in \overline{S}_i$ and ϕ is a function used to map the data to a Reproducing Kernel Hilbert Space (RKHS). The Minimal Matching Distance is approximated as follows.

$$d_{MMD}(S_i, \overline{S}_i) = \left|\left| \frac{1}{|S_i|} \sum_{x \in S_i} \phi(x) - \frac{1}{|\overline{S}_i|} \sum_{y \in \overline{S}_i} \phi(y) \right|\right|$$

- **RMSE:** The Root Mean Squared Error between S_i and $\overline{S}_i$ is given by

$$RMSE(S_i, \overline{S}_i) = \big(MSE(S_i, \overline{S}_i)\big)^{\frac{1}{2}}$$

where $MSE(S_i, \overline{S}_i) = \sum_{x \in S_i} \frac{(x-y)^2}{|S_i|}$ and $y \in \overline{S}_i$

- **EMD:** If $S_i, \overline{S}_i \in \mathbb{R}^3$ have the same size, i.e., $s = |S_i| = |\overline{S}_i|$ the Earth Mover Distance between the two point clouds is defined as follows.

$$d_{EMD}(S_i, \overline{S}_i) = \min_{\phi: S_i \to \overline{S}_i} \sum_{x \in S_i} ||x - \phi(x)||_2$$

where $\phi : S_i \to \overline{S}_i$ is a bijection.

6. Navigation Results using SLAM

6.1. Experimental Setup

We use Google Cartographer [3], a widely known LiDAR-based SLAM algorithm, to test the LiDAR sequences for navigation. We use an Intel Core i5 processor with 16 GB RAM running ROS Noetic Distribution.

We also evaluate our work against a recent, popular LiDAR-based SLAM algorithm - KISS-ICP [11]. We show the results in segmentation-devoid baseline (Column 4) and the segmentation assisted baseline (Column 5) in Table 3. We provide the translation error(ATE) on a set of seven sparse version of seven KITTI sequences in Table 3. These sequences have significant variation across the scenes they navigate and are longer.

6.2. Datasets

(a) KITTI Odometry dataset [2] is a 64-beam LiDAR dataset. It has 11 sequences with ground truth poses. For SLAM, we test **GLiDR** on all the ten sequences.
(b) ARD-16 is a 16-beam industrial dataset collected using an UGV. We follow the SLAM protocol mentioned at Kumar et al. [4] and test on the available single SLAM sequence.
(c) CARLA-64 is an extensive simulated 64-beam urban dataset with correspondence information. Is has four available SLAM sequnences with ground truth poses. We use these sequences for navigation.

6.2.1 Baselines

We use the augmented points generated by **GLiDR** along the static LiDAR backbone to assist navigation. We study the performance of **GLiDR** against segmentation-assisted and devoid baselines in sparse and dense settings.

GLiDR always works in label-devoid settings. To show its benefits over segmentation-assisted navigation, we use segmentation labels to remove dynamic objects from KITTI sequences before using them for SLAM. We call this baseline - KITTI-Seg. We compare the SLAM performance of KITTI-Seg with **GLiDR** in Table 2. For segmentation-free settings, we use the best baseline from Table 1 in the main paper for SLAM and compare the results against **GLiDR** in Table 4. We also evaluate **GLiDR** against the original dynamic LiDAR scans on SLAM in Table 4 and Section 6.3.2.

6.3. SLAM Evaluation Metrics

We evaluate the difference between pose estimates generated using model-reconstructed static and ground truth static LiDAR scans using the following metrics.

- **ATE:** Absolute Trajectory Error is used for measuring the overall global consistency between two trajectories by measuring the difference between the translation components of the two trajectories. The two trajectories specified in arbitrary coordinate frames are aligned in closed form. Subsequently, the absolute trajectory error matrix at time i is defined as $\mathbf{E}_i := \mathbf{Q}_i^{-1}\mathbf{S}\mathbf{P}_i$, where $\mathbf{S}$ is the rigid-body transformation using the least squares solution that maps $\mathbf{P}$ (the estimated trajectory) onto $\mathbf{Q}$ (the ground truth trajectory). The ATE is defined as the root mean square error from error matrices over all indices of time.

$$ATE_{RMSE} = \left(\frac{1}{n}\sum_{i=1}^{n}||trans(\mathbf{E}_i)||^2\right)^{\frac{1}{2}}$$

  where $trans(E_i)$ refers to the translational components of the absolute trajectory error matrix E_i

- **RPE:** Relative Pose Error, similar to the ATE, measures the pose error between estimated and ground truth trajectories. It quantifies the local accuracy of the trajectory over a fixed time interval δ. The relative pose error at time step i is defined as follows.

$$\mathbf{E}_i = (\mathbf{P}_i^{-1}\mathbf{P}_{i+\delta})^{-1}(\mathbf{Q}_i^{-1}\mathbf{Q}_{i+\delta})$$

  The RMSE over all time indices is computed for the translational and rotational components as follows.

$$RPE_{trans} = \left(\frac{1}{m}\sum_{i=1}^{m}||trans(\mathbf{E}_i)||^2\right)^{\frac{1}{2}}$$

$$RPE_{rot} = \left(\frac{1}{m}\sum_{i=1}^{m}||rot(\mathbf{E}_i)||^2\right)^{\frac{1}{2}}$$

  For further information on these metrics, please refer to Sturm et al., [9]

6.3.1 Comparison with Segmentation-based Baselines

KITTI has segmentation information available. We compare **GLiDR** against the segmentation-based KITTI baseline - KITTI-Seg. It uses segmentation information to remove dynamic points from LiDAR sequences before navigation. We also compare **GLiDR** against the original dynamic LiDAR sequences of all three datasets in Table 2 and Figure 3. **GLiDR** performs better than both methods by a fair margin without the assistance of segmentation labels for almost every sequence of KITTI and CARLA-64. Our trajectory estimates are on par with or better than KITTI-Seg as shown in Figure 3. The results demonstrate **GLiDR**'s capability to reinforce accurate points along existing static structures as well as newer points along occluded static structures by simply following the 0-dim $\mathcal{PH}$ based static backbone. It strengthens our premise that **(a)** augmenting static structures - both visible and occluded with newer points is superior to existing pre-processing approaches for SLAM in sparse LiDAR settings and **(b)** 0-dim $\mathcal{PH}$ based constraint and the LiDAR graph representation are highly effective at preserving global shape of LiDAR topology while generating newer static points with high precision.

We notice that for certain sequences, KITTI-Seg has higher navigation errors compared to the original dynamic sequences. Our investigation yields the following reasons. These sequences consist of certain dynamic objects that are stationary and help navigation. KITTI-Seg removes all dynamic objects, which may also include objects that are movable but stationary in the sequence. In such cases, it is devoid of several stationary points that are available in the original dynamic sequence. In these scenarios, KITTI-Seg performs inferior to the original dynamic scans.

Dataset	Sequence	Sparse LIDAR						Dense LiDAR					
		Original Dynamic		Segmented Out		Ours		Original Dynamic		Segmented Out		Ours	
		RPE Trans	RPE Rot	RPE Trans	RPE Rot	RPE Trans	RPE Rot	RPE Trans	RPE Rot	RPE Trans	RPE Rot	RPE Trans	RPE Rot
KITTI	0	1.10	**0.060**	**1.096**	**0.060**	1.1	**0.060**	**1.10**	**0.060**	**1.100**	**0.060**	**1.10**	**0.060**
	1	1.73	0.106	2.047	**0.039**	**1.56**	**0.039**	1.595	**0.039**	1.542	**0.039**	**1.54**	**0.039**
	2	**1.48**	**0.488**	1.501	0.049	1.482	0.049	1.549	**0.049**	**1.547**	**0.049**	1.55	**0.049**
	4	114.587	0.008	**1.924**	0.007	2.06	**0.005**	**2.04**	**0.005**	2.05	**0.005**	**2.04**	**0.005**
	5	**1.210**	**0.043**	1.214	0.0433	**1.21**	**0.043**	**1.21**	**0.043**	1.215	**0.043**	**1.21**	**0.043**
	6	1.66	**0.052**	**1.643**	**0.052**	1.66	**0.052**	**1.655**	**0.051**	1.670	0.052	1.66	**0.051**
	7	**1.04**	**0.057**	**1.040**	**0.057**	**1.04**	**0.057**	**1.040**	**0.0567**	**1.040**	0.057	**1.04**	0.057
	8	53.339	**0.109**	**25.274**	0.134	30.85	0.146	**46.170**	**0.107**	55.603	0.129	60.09	0.112
	9	**1.768**	**0.046**	**1.768**	**0.046**	1.77	**0.046**	**1.773**	**0.046**	**1.773**	**0.046**	1.78	0.048
	10	1.366	**0.450**	1.363	**0.045**	**1.36**	**0.045**	**1.355**	**0.04488**	1.363	0.045	1.36	0.045
CARLA-64	0	**0.099**	0.451	-	-	**0.099**	**0.446**	**0.103**	**0.414**	-	-	0.083	0.438
	1	0.068	**0.408**	-	-	**0.057**	0.410	**0.051**	0.451	-	-	0.056	**0.406**
	2	0.100	**0.547**	-	-	**0.099**	0.55	0.078	**0.527**	-	-	**0.066**	0.557
	3	**0.079**	**0.204**	-	-	0.117	0.395	**0.137**	**0.375**	-	-	0.212	0.380
ARD-16	0	**0.166**	5.07	-	-	0.171	**4.955**	0.178	**4.878**	-	-	**0.171**	4.934

Table 2. **GLiDR** comparison against segmentation-assisted baseline on KITTI-Seg and against original dynamic LiDAR sequence based on the RPE metric.

Seq	Length	GLiDR	Best Baseline	KITTI-Seg	Original
0	4390	**7.53**	190.67	7.98	7.7
2	4485	**12.17**	292.65	13.57	12.42
5	2750	**4.27**	152.03	4.41	4.41
6	1091	**1.65**	126.23	1.67	1.67
7	1111	0.98	85.83	0.9	**0.7**
8	5149	**4.19**	188.5	4.21	4.31
9	1599	**4.45**	212.24	4.82	4.48

Table 3. ATE numbers with KISS-ICP SLAM algorithm

We notice that for sequences 6,7,9 and 10 in the dense settings, **GLiDR** performs equivalent to but not better than the two settings. We observe that in dense settings, augmenting existing static structures that are already dense has no major impact on navigation. However, augmenting occluded static structures along the static backbone brings newer points that are not available in the original scan and can help navigation. We observe that sequences 6, 7, 9, and 10 have very few dynamic occlusions (and hence fewer occluded static structures - 8, 15, 13, 10, respectively), resulting in negligible effect on navigation performance.

A natural question arises: What levels of sparsity can **GLiDR** handle? We perform several experiments similar to Table 2 for KITTI with sparser LiDAR scans - 8-beam and 4-beam scans. The results of the experiments are available in Table 5 and Section 6.3.3. Our investigations suggest that high sparsity levels in the above cases destroy the LiDAR topology to the extent that 0-dim $\mathcal{PH}$ can no longer recover an accurate backbone of the LiDAR scene. Due to this, **GLiDR** navigation results perform poorly against the original dynamic and the KITTI-Seg baseline. **GLiDR** does not perform well with such sparsity.

6.3.2 Comparison with Segmentation Devoid Baseline

We compare the navigation results of **GLiDR** against the best baseline for static point augmentation in Table 4, Figure 3. **GLiDR** performs extremely well and consistently outperforms the baseline by a large margin for KITTI and CARLA-64 datasets in the sparse settings. In Figure 3, we observe that for the sparse case, the best baseline (blue) misses the ground-truth trajectory (dotted line) by a heavy margin. **GLiDR** is able to navigate accurately and performs better for most sequences. **GLiDR** generates consistent points along the sparse structures by following the static topology-based backbone unlike the best baseline, which in the sparse case fails to generate consistent points along static structures. The 0-dim $\mathcal{PH}$ prior ensures that **GLiDR** reinforces static structures only along the single connected component outlined by the static backbone. This allows accurate scan matching using the predicted static points, leading to better navigation performance. It also performs well against the baseline in dense settings.

We observe that for the ARD-16 dataset (which only works in segmentation devoid settings), **GLiDR** performs comparable but not better than the original dynamic scan for navigation in Table 4. The reasons lie in the semantics and the structure of the ARD-16 dataset. The improvement in navigation for ARD-16 is marginal in both settings. ARD-16 is collected in a closed industrial environment, unlike KITTI and CARLA-64. Every LiDAR scan in the sequence observes a large part of the navigation environment. Unlike urban settings, the collected LiDAR scans overlap significantly, even at turns. Consecutive scans have sufficient static points for accurate scan matching despite sparsity, even in the absence of augmented static points. The static points introduced by **GLiDR** do not provide any significant improvement to navigation performance. However, the newly introduced static points along occluded static struc-

Dataset	Sequence	Sparse LIDAR				Dense LiDAR			
		Best Baseline		Ours		Best Baseline		Ours	
		RPE Trans	RPE Rot	RPE Trans	RPE Rot	RPE Trans	RPE Rot	RPE Trans	RPE Rot
KITTI	0	1.17	**0.059**	**1.1**	0.060	1.10	0.060	**1.10**	**0.060**
	1	2.07	**0.037**	**1.56**	0.039	2.40	0.039	**1.54**	**0.039**
	2	**1.38**	**0.048**	1.48	0.049	**1.54**	0.049	1.55	**0.049**
	4	**1.68**	**0.006**	2.06	0.005	2.04	0.0052	**2.04**	**0.005**
	5	**1.10**	0.043	1.21	**0.043**	1.21	0.043	**1.21**	**0.043**
	6	**1.42**	**0.051**	1.66	0.0516	**1.64**	0.051	1.66	**0.051**
	7	**0.93**	**0.056**	1.04	0.057	1.04	0.057	**1.04**	**0.057**
	8	**20.85**	**0.129**	30.85	0.146	**33.075**	**0.071**	60.00	0.112
	9	**1.24**	**0.045**	1.77	0.046	**1.77**	**0.046**	1.78	0.048
	10	**1.01**	**0.044**	1.36	0.045	**1.31**	**0.044**	1.36	0.045
CARLA-64	0	**0.099**	**0.439**	**0.099**	0.446	0.090	**0.414**	**0.083**	0.438
	1	0.156	0.411	**0.057**	**0.41**	0.065	**0.394**	**0.056**	0.406
	2	0.125	**0.546**	**0.099**	0.55	0.141	**0.553**	**0.066**	0.557
	3	**0.106**	0.397	0.117	**0.395**	**0.110**	0.399	0.120	**0.393**
ARD-16	0	**0.170**	**4.947**	0.171	4.955	0.182	**4.85**	**0.171**	4.934

Table 4. Navigation performance comparison of **GLiDR** against the best baseline in segmentation-devoid settings using RPE metric in sparse and dense settings. For RPE, lower is better.

tures help to generate accurate binary segmentation masks (Section 6.1.1 in the main paper), which are vital for safe navigation.

6.3.3 Experiments on more sparse datasets

What level of sparsity can be handled by **GLiDR**? To answer this question, we perform experiments using more sparse LiDAR scans - 8 and 4-beam LiDAR dataset and test them on navigation. We perform experiments with the same baselines as Table 3 in the main paper - Original Dynamic, KITTI-Seg. Original Dynamic is the LiDAR range image without any pre-processing. KITTI-Seg uses segmentation labels to remove dynamic objects from the LiDAR range image before using them for navigation. The experiments are performed using the KITTI dataset. The navigation results are available in Table 5.

We observe that, unlike 64 and 16 beam cases, for the more sparse cases (8 and 4-beam), **GLiDR** does not perform well on the ATE metric for most of the sequences. ATE is a strong measure of the global consistency of the predicted trajectory with the groundtruth. We conclude that with sparsity beyond 16 beam LiDAR scans, **GLiDR** does not outperform the existing baselines. Our investigation suggests that for 8 and 4-beam LiDAR datasets, the 0-dim $\mathcal{PH}$ based backbone is not able to calculate an accurate global backbone and misses many details that were captured in the 16-beam datasets. We demonstrate these findings in Table 6 and 7. In the figures, column 1 denotes the original LiDAR, while Column 2, 3, and 4 denotes the 0-dim $\mathcal{PH}$ backbone for 16, 8, and 4 beam LiDAR sparse LiDAR scans, respectively. We observe that while the 16-beam LiDAR backbone captures the global outline of the static structures accurately, the 8 and 4-beam LiDAR miss a lot of details and are unable to capture the global static LiDAR structure accurately. This results in insufficient and sub-optimal augmentation of static points, which translates into inferior navigation performance for 8 and 4 LiDAR beam based scans.

References

[1] Herbert Edelsbrunner. Computational topology an introduction, 2008. 1

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 5

[3] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278. IEEE, 2016. 4

[4] Prashant Kumar, Sabyasachi Sahoo, Vanshil Shah, Vineetha Kondameedi, Abhinav Jain, Akshaj Verma, Chiranjib Bhattacharyya, and Vinay Vishwanath. Dslr: Dynamic to static lidar scan reconstruction using adversarially trained autoencoder, 2021. 5

[5] Prashant Kumar, Sabyasachi Sahoo, Vanshil Shah, Vineetha Kondameedi, Abhinav Jain, Akshaj Verma, Chiranjib Bhattacharyya, and Vinay Vishwanath. Dynamic to static lidar scan reconstruction using adversarially trained auto encoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1836–1844, 2021. 4

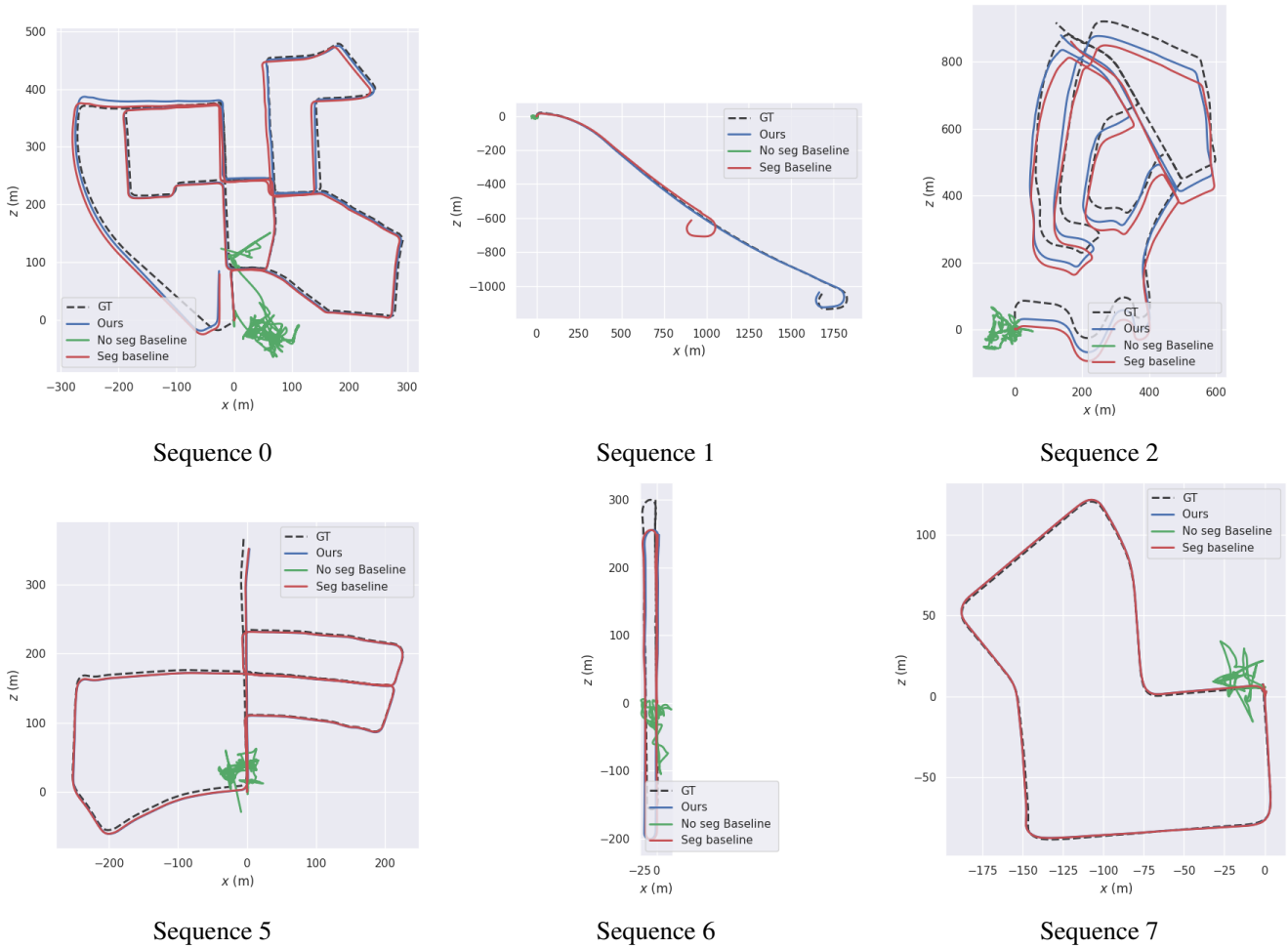
[6] Prashant Kumar, Onkar Susladkar, Dhruv Makwana, Anurag Mittal, and Prem Kumar Kalra. Movese: Movable and moving lidar scene segmentation with improved navigation in seg-label free settings. *arXiv preprint arXiv:2306.14812*, 2023. 4

[7] Michael Moor, Max Horn, Bastian Rieck, and Karsten Borgwardt. Topological autoencoders. In *Proceedings of the 37th International Conference on Machine Learning*, pages 7045–7054. PMLR, 2020. 4

[8] Lia Papadopoulos, Mason A Porter, Karen E Daniels, and Danielle S Bassett. Network analysis of particles and grains. *Journal of Complex Networks*, 6(4):485–565, 2018. 1

Sequence	8-beam			4-beam		
	Original Dynamic	Segmented Out	Ours	Original Dynamic	Segmented Out	Ours
	ATE/RPE Trans/RPE Rot			ATE/RPE Trans/RPE Rot		
0	11.08/**1.10**/**0.06**	**10.98**/**1.10**/**0.06**	100.61/**1.10**/**0.06**	23.46/1.09/**0.06**	**21.63**/1.09/**0.06**	183.36/**1.01**/**0.06**
1	440.63/2.30/**0.04**	**290.35**/**2.00**/**0.04**	732.18/2.82/**0.04**	**715.32**/3.07/0.04	718.71/3.18/0.04	734.16/**2.89**/**0.03**
2	113.12/1.75/**0.05**	**29.84**/1.52/**0.05**	223.09/**1.40**/**0.05**	273.32/1.67/**0.05**	**252.56**/1.65/**0.05**	303.64/**1.34**/**0.05**
4	117.61/2.26/**0.01**	**117.38**/2.29/**0.01**	119.44/**2.03**/**0.01**	**117.41**/2.20/0.01	118.31/**2.15**/0.01	118.62/2.42/**0.003**
5	**3.15**/1.21/**0.04**	4.69/1.21/**0.04**	150.11/**0.97**/**0.04**	12.16/1.21/**0.04**	**12.12**/1.20/**0.04**	162.72/**1.02**/**0.04**
6	22.07/1.66/**0.05**	**13.25**/1.66/**0.05**	132.87/**1.38**/**0.05**	80.59/1.66/**0.05**	**61.30**/1.70/**0.05**	134.75/**1.54**/**0.05**
7	2.80/1.04/**0.06**	**2.21**/ 1.04/ **0.06**	4.95/**1.03**/**0.06**	2.04/1.03/**0.06**	**1.59**/1.03/**0.06**	81.92/**0.87**/**0.06**
8	**106.79**/44.14/**0.09**	157.50/43.88/**0.09**	202.36/**13.22**/0.15	195.29/39.43/0.14	**164.25**/40.34/0.18	200.01/**21.36**/**0.012**
9	13.19/1.76/**0.05**	**12.51**/1.75/**0.05**	15.22/**1.67**/**0.05**	**20.50**/**1.70**/**0.05**	32.83/**1.70**/**0.05**	50.0/1.74/**0.05**

Table 5. GLiDR comparison against a segmentation-assisted baseline - KITTI-Seg and against original dynamic LiDAR sequence for 8 beam and 4 beam LiDAR scans. Lower is better. **GLiDR** does not perform better for these sparsity levels.



Sequence 0 Sequence 1 Sequence 2

Sequence 5 Sequence 6 Sequence 7

Figure 3. SLAM Trajectory comparison of **GLiDR** for sparse LiDAR scans against segmentation assisted and segmentation devoid baseline.

[9] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580. IEEE, 2012. 5

[10] Velodyne. Visualizing multi-dimensional persistent homology, industrial and applied mathematics seminar, university of oxford, 2014. 2

[11] Ignacio Vizzo, Tiziano Guadagnino, Benedikt Mersch, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Kiss-icp: In defense of point-to-point icp–simple, accurate, and robust registration if done the right way. *IEEE Robotics and Au-*

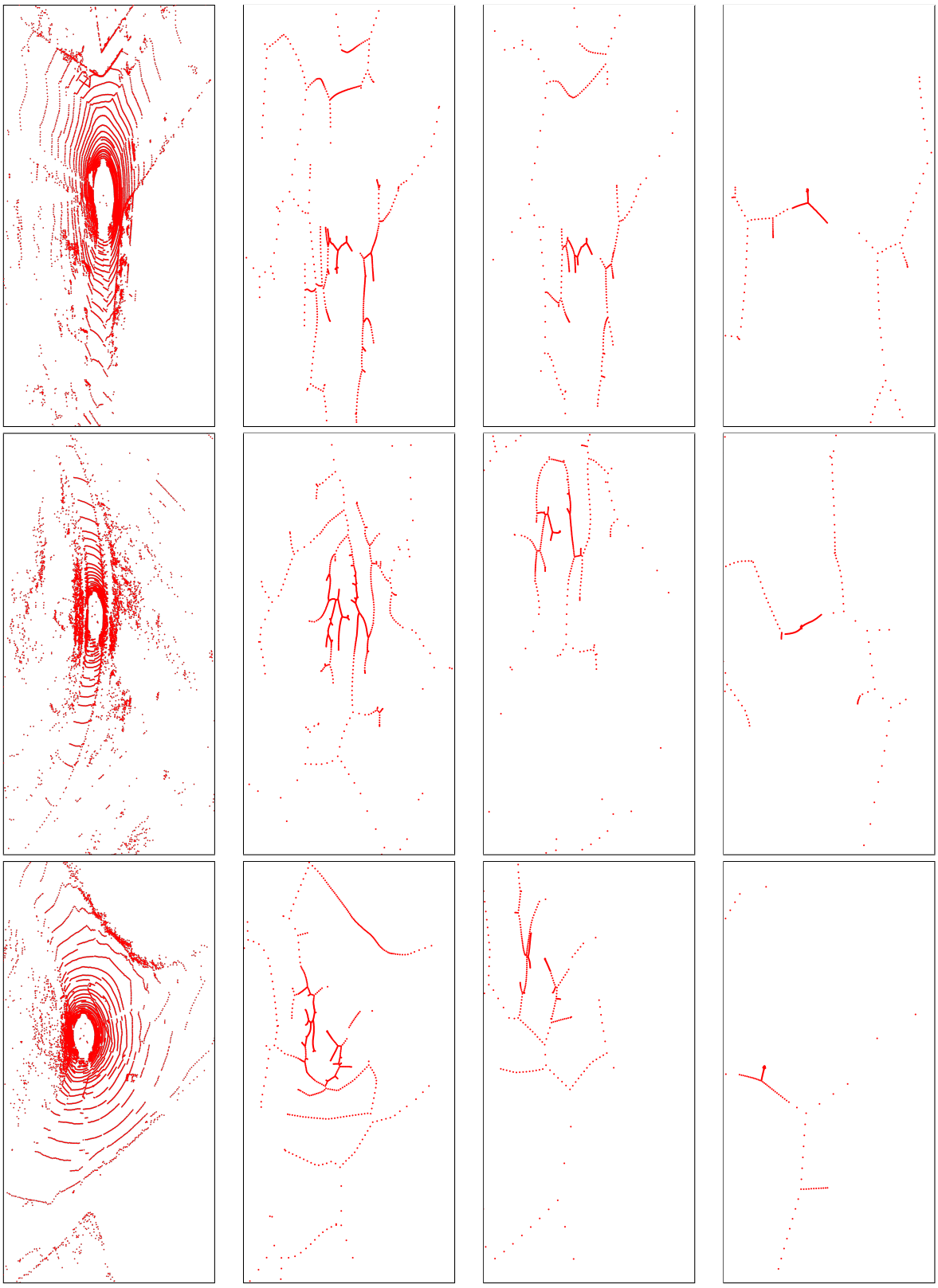


Table 6. 0-dim $\mathcal{PH}$ backbone results for several LiDAR scans. View from left to right. **Left**: Original 16-beam LiDAR scans **Second from left**: Backbone for 16-beam LiDAR scan. **Third from left**: Backbone for 8-beam LiDAR scan. **Right**: Backbone for 4-beam LiDAR scan. While the backbone for a 16-beam LiDAR scan is precise and accurate, the backbone for an 8 and 4-beam LIDAR scan misses several details of the original LiDAR. Thus, **`GLiDR`** is unable to handle sparsity levels of 8 and 4-beam LiDAR scans.

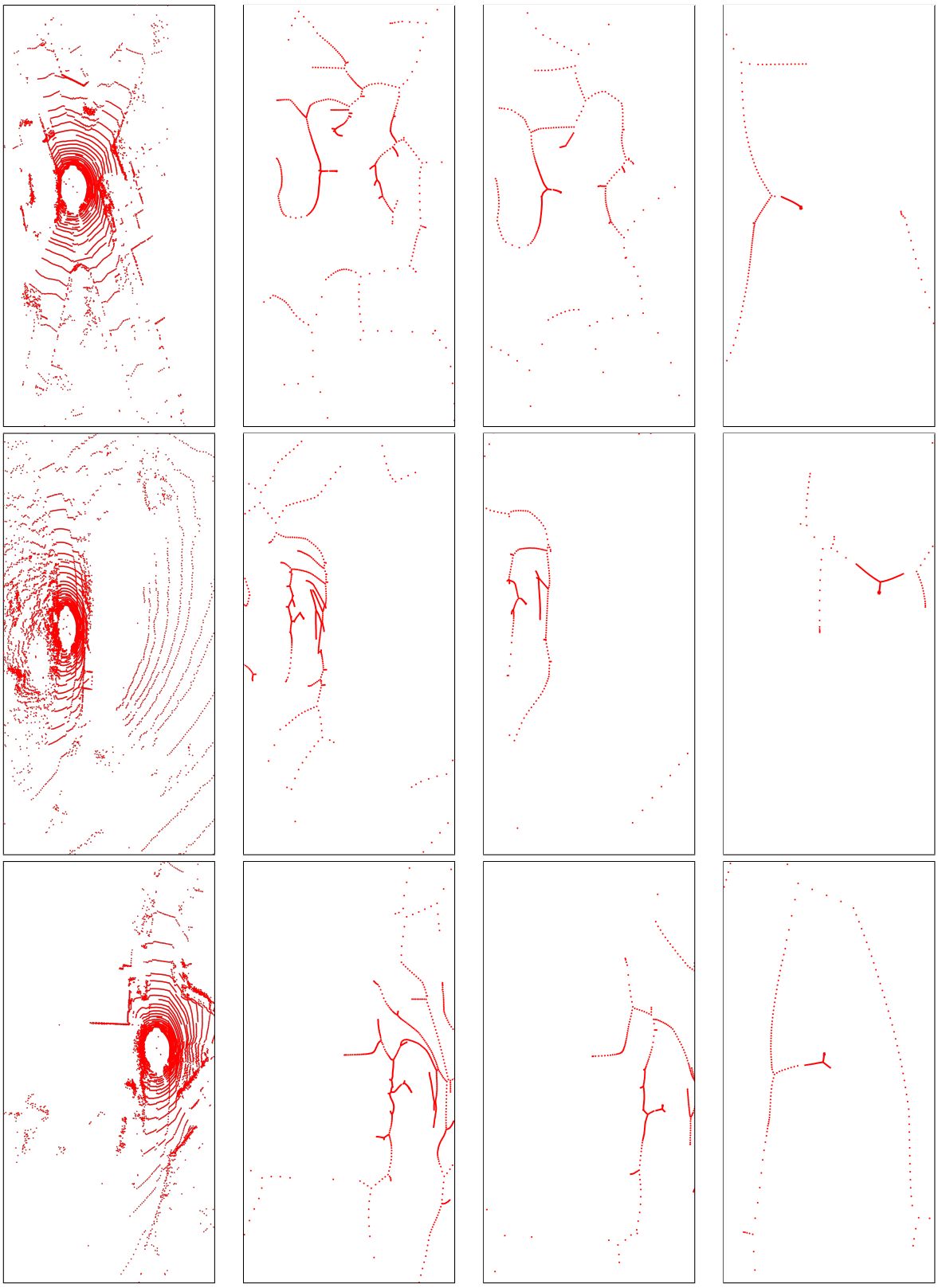


Table 7. 0-dim $\mathcal{PH}$ backbone results for several LiDAR scans. View from left to right. **Left**: Original 16-beam LiDAR scans **Second from left**: Backbone for 16-beam LiDAR scan. **Third from left**: Backbone for 8-beam LiDAR scan. **Right**: Backbone for 4-beam LiDAR scan. While the backbone for a 16-beam LiDAR scan is precise and accurate, the backbone for 8 and 4-beam LIDAR scan misses several details of the original LiDAR. Thus, **`GLiDR`** is unable to handle sparsity levels of 8 and 4-beam LiDAR scans.

tomation Letters, 8(2):1029–1036, 2023. 4

[12] Teng Wang, Lin Wu, and Changyin Sun. A coarse-to-fine approach for dynamic-to-static image translation. *Pattern Recognition*, 123:108373, 2022. 4

[13] Mingye Xu, Yali Wang, Yihao Liu, Tong He, and Yu Qiao. Cp3: Unifying point cloud completion by pretrain-prompt-predict paradigm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2