

# Efficiently Closing Loops in LiDAR-Based SLAM Using Point Cloud Density Maps

Saurabh Gupta

Tiziano Guadagnino

Benedikt Mersch

Niklas Trekel

Meher V. R. Malladi

Cyrill Stachniss

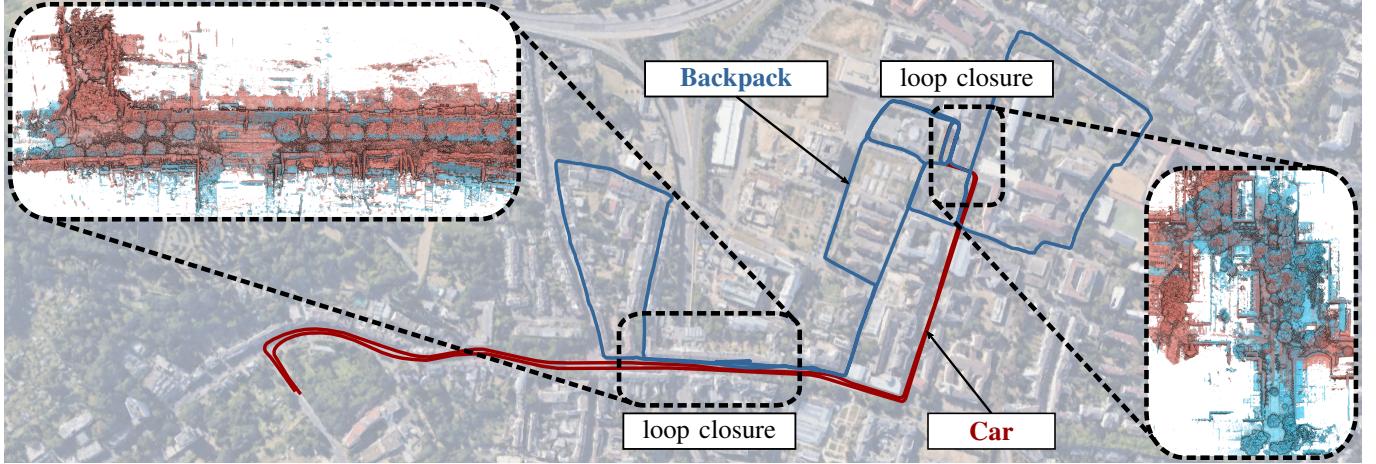


Fig. 1: An example of loop closures detected between two sequences recorded with different LiDAR sensor platforms with a revisit interval of 2 weeks. In blue is the Backpack sequence, recorded in a campus environment with a Hesai Pandar-128 LiDAR mounted on a backpack. In red is the Car sequence, recorded in the city with an Ouster OS1-128 LiDAR. Both trajectories were individually optimized through a pose-graph with in-session loop closure constraints obtained from our pipeline. We used our loop closure pipeline to detect closures between these two sequences, which have little overlap, and align the two trajectories using the multi-session loop closure constraints. The overlapping areas are highlighted by small rectangles, and the corresponding loop closure from these areas is shown in the enlarged rectangles.

**Abstract**—Consistent maps are key for most autonomous mobile robots. They often use SLAM approaches to build such maps. Loop closures via place recognition help maintain accurate pose estimates by mitigating global drift. This paper presents a robust loop closure detection pipeline for outdoor SLAM with LiDAR-equipped robots. The method handles various LiDAR sensors with different scanning patterns, field of views and resolutions. It generates local maps from LiDAR scans and aligns them using a ground alignment module to handle both planar and non-planar motion of the LiDAR, ensuring applicability across platforms. The method uses density-preserving bird’s eye view projections of these local maps and extracts ORB feature descriptors from them for place recognition. It stores the feature descriptors in a binary search tree for efficient retrieval, and self-similarity pruning addresses perceptual aliasing in repetitive environments. Extensive experiments on public and self-recorded datasets demonstrate accurate loop closure detection, long-term localization, and cross-platform multi-map alignment, agnostic to the LiDAR scanning patterns, fields of view, and motion profiles.

**Index Terms**—SLAM, Localization, Mapping, LiDAR-based Place Recognition

All authors are with the Center for Robotics, University of Bonn, Germany. Cyrill Stachniss is additionally with the Department of Engineering Science at the University of Oxford, UK, and with the Lamarr Institute for Machine Learning and Artificial Intelligence, Germany.

This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy, EXC-2070 – 390732324 – PhenoRob, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under STA 1051/5-1 within the FOR 5351 (AID4Crops), and by the European Union’s Horizon Europe research and innovation programme under grant agreement No 101070405 (DigiForest).

## I. INTRODUCTION

Mobile robots must navigate their surroundings safely and efficiently. They need to know their location within the environment to successfully navigate to a desired place or explore new areas. Accurate localization helps the robots generate accurate maps of the environment, which they can use for navigation. Traditionally, robots often localize themselves using sequential data acquired through exteroceptive sensors like cameras and laser range sensors (LiDAR), proprioceptive sensors like wheel odometers and IMUs, or a combination. Depending on the application and availability, outdoor systems often exploit additionally GNSS for global positioning.

LiDAR sensors are frequently used in robotics due to their accurate and dense 3D range data. Many previous studies have advanced the field of sequential pose estimation using LiDAR sensors [9], [11], [12], [15], [58]. Such sequential pose estimation alone, also called sensor odometry, suffers from drifting pose estimates over time due to the inherent noise in the robot motion and sensor data, dynamics in the environment, and non-trivial data association problems.

We can compensate for such drift and improve pose estimation by recognizing places previously seen by the robot. This task is often referred to as loop closing. It allows the use of the geometric information from the LiDAR observations at revisited locations to correct the pose drift, for example, through pose-graph optimization. Robust loop closure detection is paramount in simultaneous localization and mapping (SLAM)

systems. Robots must recognize that they have returned to a previously visited place to close a loop. Place recognition is a data association task between the current view and a map or database of previously seen places. Generating a database of places is a non-trivial task that requires crafting an as unique as possible description of the environment, invariant to the changes in the viewpoint. Perceptual aliasing is another challenge, as distinct places with similar structures can confuse place recognition algorithms. Also, such a database should filter the dynamic objects appearing in the sensor data to achieve robust place recognition across long time intervals.

In our work, we generate the database from the point clouds provided by the sensor. The descriptors of places computed from the point clouds and stored in the database should be invariant to the LiDAR's scanning pattern, spatial density, and field of view (FoV). Furthermore, to perform place recognition for loop closures in a SLAM system, the descriptor should capture the scene's geometry to enable relative pose estimation between local maps and actually compute a loop closure constraint for the pose-graph optimization.

Feature descriptor-based techniques are a common practice in LiDAR-based loop closure detection and place recognition [4], [17], [25], [44], [48], [49]. Global feature descriptors computed over the point clouds are easy to store in a database and allow for quick retrievals of revisit candidates [22], [33], [57], [63]. However, such methods often do not provide an initial geometric alignment between the detected loop closures and require an additional point cloud registration step to perform such an alignment. In contrast, local feature descriptors computed over local 3D patches within a point cloud can aid the geometric alignment of revisited locations [34], [49], [54], [65]. However, they require a nuanced approach to store the feature descriptors efficiently in a database.

Due to the algorithmic complexity of processing 3D point clouds to obtain local or global feature descriptors, several methods perform a bird's eye view (BEV) projection [22], [30], [34], [36] or a cylindrical projection [6], [37], [53] of the point clouds. This results in a 2D representation of the point cloud data, allowing faster feature detection and matching towards online loop closure detection in SLAM.

This manuscript can be seen as an extension of the work by Gupta et al. [16], which proposed loop closure detection using local maps by detecting local feature descriptors from BEV projections. However, they strictly assumed a planar motion of the sensor platform, limiting their application to instrumented vehicular platforms and providing only 2D rigid body alignments of loop closures. Furthermore, the work by Gupta et al. [16] also struggled in scenarios with perceptual aliasing, such as bridges or tunnels, resulting in false loop closures, i.e., situations in which the proposed approach performs better.

The main contribution of this article is a robust loop closure detection pipeline that works with various LiDAR sensors having different motion profiles, invariant to their scanning pattern, field of view, and viewpoint. We generate local maps by aggregating consecutive scans and creating a density-preserving BEV projection of the local maps as an intermediate 2D representation for computing local features describing the structural information in the 3D local maps.

Since the ground planes can be consistently identified in outdoor environments during revisits, we use them as a reference plane for consistent BEV projections across revisits. We propose a ground alignment module to identify such a ground plane in each local map and transform the local map such that this ground plane coincides with the local xy-plane of the reference frame. This simplifies the BEV projection and restricts the loop closure alignment to two dimensions on the common ground plane and also makes our approach applicable to various motion profiles of LiDAR sensors. We store binary ORB [46] feature descriptors from the BEV projection into a Hamming distance embedding binary search tree [50]. We design our approach to be robust against the effects of scene similarity, also known as perceptual aliasing, by performing a self-similarity pruning of the feature descriptors before inserting them in the database. Using a Hamming distance metric, we obtain loop closure candidates by matching feature descriptors from query local maps against this database. Our subsequent RANSAC-based geometric validation step provides us with loop closures along with a 2D alignment between BEV projections of the local maps. When combined with the ground alignment estimate of each local map, we obtain a complete 3D estimate of the global alignment between the local maps. We provide an extensive experimental evaluation on multiple datasets, with sequences recorded with a wide range of LiDARs mounted on different mobile platforms, testing the accuracy and robustness of our pipeline in challenging scenarios.

In sum, we make five key claims which are backed up by the paper and our experimental evaluation:

- 1) Our approach can detect loop closures between local maps generated from various LiDAR sensors with different scanning patterns, FoVs, and resolutions.
- 2) Our approach can perform multi-session loop closure detection and alignment with long-term revisits.
- 3) Our approach also works with handheld platforms having non-planar motion in the LiDAR sensor frame and provides a complete 3D rigid body transform to align the detected loop closures.
- 4) Our approach is robust against perceptual aliasing in environments with repetitive structures.
- 5) Our approach can detect loop closures between sequences having minimal overlap, recorded with different LiDAR sensor platforms, enabling cross-platform multi-map alignment.

The implementation of our approach will be released upon acceptance of the manuscript.

## II. RELATED WORK

Zhang et al. [69] provides a thorough literature review on place recognition with 3D LiDAR and Yin et al. [66] on global localization. This section briefly overviews key approaches for LiDAR-based place recognition and loop closure detection in SLAM, related to our approach.

### A. Place Recognition with 3D LiDAR Scans

The most straightforward approach towards loop closure detection in SLAM is using the individual scans and their

poses obtained from odometry. The pose-similarity between non-consecutive scans within a search radius is used to detect loop closures [45], [52]. S4-SLAM [70] and the approach by Mendes et al. [39] use the odometry information to compute the overlap between point clouds to check if they are recorded from the same location. Chen et al. [6] use convolutional neural networks to calculate the overlap between the range image representations of point clouds to detect loops. These methods primarily perform a place retrieval task, often requiring a subsequent point cloud registration step to validate the loop closures. Furthermore, they can be sensitive to the magnitude of drift present in the odometry pose estimates, requiring a search radius proportional to the drift.

Place recognition has been widely studied in the context of camera images [8], [14], [40], [59], [60]. Consequently, much research on LiDAR place recognition has been motivated by approaches in the camera vision domain. In particular, many approaches have extended the idea of local feature descriptors to 3D point clouds [4], [19], [48], [49], [65], discretizing the neighborhood around 3D features into a geometrical grid. They compute a descriptor from the points in this neighborhood based on their height [4], density [13], [55], or distance and angle [47], [48]. These local feature descriptor-based methods estimate a complete 3D relative pose between the loop closures. However, they have a high computational cost to identify such feature descriptors from 3D point clouds. They are sensitive to the radially increasing sparsity of point clouds obtained from standard spinning LiDAR.

Many methods [26], [38], [44], [57] take an orthogonal approach by computing one global descriptor per LiDAR scan. Magnusson et al. [38] superimpose a voxel grid on the point clouds and approximate a normal distribution over points in each grid cell. They compute a global descriptor for the point clouds by computing a histogram over these normal distributions. More recently, Kim et al. [26] proposed with SOLiD a lightweight global descriptor by representing the point clouds in polar coordinates and discretizing them into range-elevation and azimuth-elevation directions, respectively, with each discrete bin storing the number of points. Such global descriptors are easy to store in a simple database, resulting in a faster matching process to retrieve loop closure candidates. However, global descriptor-based methods typically require revisits within small error margins around the reference pose to limit the variation in the global descriptor. They often do not generalize when performing place recognition across LiDARs with different scanning patterns and resolutions.

### B. Projection-based Place Recognition

A lot of recent works focus on speeding up the recognition by describing 3D point clouds by a 2D projection [16], [22], [30], [34], [35], [36], [53], [63], [67]. Even though such a projection results in losing geometric information, such methods perform comparably and sometimes even better than their full 3D counterparts, especially in outdoor scenes. Among the two-dimensional projection-based methods, the cylindrical projection into range images [6], [51], [53], [54] and the orthogonal projection into BEV images [16], [25], [33], [34], [61], [68] are two widely used 2D projections.

Range image projections are equivariant to azimuthal rotations due to the underlying cylindrical projection. They can recover the relative yaw between loop closures but suffer from scale distortions due to lateral shifts in the LiDAR viewpoint. On the other hand, BEV projections preserve 2D geometry along the local ground plane, which is vital in the case of autonomous ground-operated robots. Among the BEV projection methods, the elevation map is widely used [22], [25], [36], [64], [68]. An elevation map preserves the maximum elevation for each discrete pillar in the BEV representation.

Scan Context by Kim et al. [25] is a widely known LiDAR-based loop closure approach. It computes a global descriptor for each LiDAR scan using the elevation map in polar coordinates. The polar coordinates make the descriptor equivariant to in-plane rotations, thereby allowing the loop closure alignment module to estimate the relative yaw between LiDAR scans from the same location. However, the polar coordinate representation can be sensitive to in-plane translations. Scan Context++ [22] improves upon this drawback of Scan Context and computes elevation maps in Cartesian coordinates to augment the polar elevation maps. However, preserving the maximum elevation of the points after the BEV projection makes the two approaches sensitive to the vertical resolution and field of view of the LiDAR.

In contrast, BVMatch [34] proposes a density-preserving BEV projection of point clouds and a Log Gabor filtering step. It requires training a bag of words model to perform place recognition, making it unsuitable for real-time applications such as SLAM. Our approach also uses a density-preserving BEV projection. It generates a database of local feature descriptors from the BEV image online using the Hamming distance embedding binary search tree [50] for efficient operation.

Yuan et al. [68] propose utilizing a set of geometric primitives, i.e., triangles with unique side lengths, to describe a point cloud. The feature vertices of such triangles are obtained by a local BEV projection of points within voxels that lie at the boundaries of large planar regions in the environment. BTC [67] improves upon this work by proposing a binary descriptor for a detailed representation of local geometry. They combine the triangle descriptors and the newly proposed binary descriptors to perform loop closure retrieval. The works by Yuan et al. [67], [68] use a sub-map representation of the environment by accumulating a fixed number of consecutive scans. This helps them tackle the sparsity issue present with rotating 3D LiDAR. The sub-maps also make their approach generalizable to LiDAR with non-repetitive scanning patterns and a small FoV. We also use a local map representation to detect loop closures. However, unlike STD and BTC, we accumulate scans until a minimum displacement of the platform to ensure a sufficient portion of the scene is captured in the local maps.

### C. Place Recognition with Learning-based Approaches

The recent popularity of deep neural networks, attributed to the widespread availability of specialized hardware and software to train such networks, has generated a lot of interest

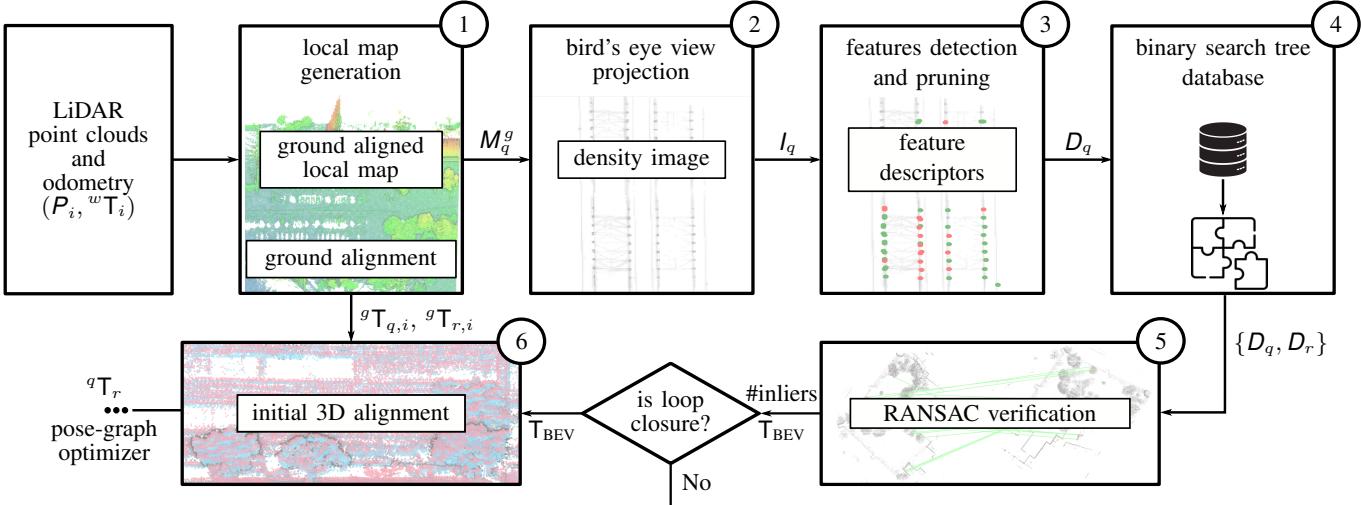


Fig. 2: An overview of our pipeline for loop closure detection and alignment. Given an input stream of point clouds  $P_i$ , with corresponding odometry estimate  $w\mathbf{T}_i$ , (1) we generate a query local map  $M_q^g$  based on travel displacement criteria. This local map is transformed by  ${}^g\mathbf{T}_{q,i}$  such that the ground plane in the local map is aligned with the xy-plane of the reference frame. (2) Then, we project the ground-aligned local map  $M_q^g$  on the local xy-plane, from a bird's eye view, to obtain a 2D density image  $I_q$ . (3) We extract ORB feature descriptors  $D_q$  from the density image and perform a self-similarity pruning to remove the repetitive features. (4) These feature descriptors are incrementally inserted into a binary search tree database. This database returns feature matches between a query local map  $M_q^g$  with feature descriptors  $D_q$  and reference local maps  $M_r^g$  with feature descriptors  $D_r$ . These feature matches serve as the initial set of loop closure candidates. (5) We perform a geometric verification and alignment of these loop closure candidates using a RANSAC scheme, which gives us the number of inlier matches and the corresponding 2D transform  $\mathbf{T}_{\text{BEV}}$ . The candidates are classified as a loop closure if we obtain a minimum number of inliers. (6) Our pipeline also provides a relative 3D transform  ${}^q\mathbf{T}_r$  for the initial alignment of the two local maps  $M_q^g$  and  $M_r^g$ , which a subsequent pose-graph optimizer can use.

from the LiDAR place recognition perspective [6], [10], [27], [36], [37], [57]. PointNet [42] and PointNet++ [43] use multi-layer perceptron networks to detect local feature descriptors from point clouds. PointNetVLAD [57] proposes an end-to-end trained neural network that combines the local feature descriptors obtained from PointNet [42] into a global descriptor using NetVLAD [1]. SegMap [10] segments point clouds using a region-growing algorithm and compute data-driven descriptors of these segments to detect loop closures.

BEV projections are also gaining popularity with learning-based approaches [23], [36], [64]. Kim et al. [23] repurpose the Scan Context [25] image as a three-channel image using a jet colormap over a range of structural heights. They train a convolutional neural network using these three-channel Scan Context images and perform localization as a classification task. Xu et al. [64] propose an encoder-decoder network to extract descriptors from differentiable Scan Context images. BEVPlace [36] performs group convolutions to detect local features on BEV images and computes a global descriptor for place retrieval. Even though learning-based approaches achieve impressive performance regarding place recognition precision and recall metrics, they require a computationally expensive offline pre-training step on a GPU, affecting their application to real-time loop closure detection in SLAM. Furthermore, these methods do not generalize to out-of-distribution data obtained from different environments with different LiDAR sensors.

We use local maps as the primary representation for detecting loop closures with LiDAR sensor data instead of individual scans, similar to Gupta et al. [16]. We generate local maps based on travel displacement criteria, unlike Yuan et

al. [67], [68], which proposes accumulating a fixed number of individual scans. Such maps help us overcome the sparsity issue of rotating LiDAR and make our method agnostic to the LiDAR's scanning pattern, FoV, and resolution. We use a density-preserving BEV projection of the local maps on the local ground plane to reduce their dimensionality for computational efficiency. We identify the local ground plane from the local map. Subsequently, we compute ORB feature descriptors [46] from these BEV images and store them in a binary search tree [50] for place recognition. We mitigate the negative impact of perceptual aliasing on our feature descriptor-matching algorithm by performing a self-similarity pruning of the ORB feature descriptors [3]. Our loop closure pipeline provides a complete 3D global alignment between the local maps.

### III. OUR METHODOLOGY

We propose an approach to detect loop closures between local maps using their BEV representation that works with both vehicle-mounted and handheld LiDAR sensor platforms. It additionally provides a complete 3D initial guess of the relative pose between local maps involved in the closures. We show an overview of our complete pipeline in Fig. 2.

We explain in Sec. III-A the procedure to generate local maps from LiDAR point clouds based on a travel displacement criterion. Then, in Sec. III-B, we present an approach to detect and align the ground plane in the local maps to the xy-plane of the local reference frame. This alignment allows us to compute a BEV representation even in scenarios with a non-planar motion of the LiDAR sensor, as explained in Sec. III-C. We compute binary descriptors for point features

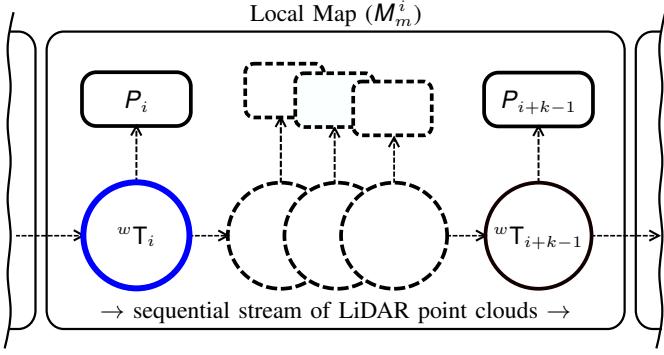


Fig. 3: A block diagram showcasing the composition of a local map  $M_m^i$ , generated using point clouds  $\{P_i, \dots, P_{i+k-1}\}$  registered with corresponding odometry pose estimates  $\{{}^w\mathbf{T}_i, \dots, {}^w\mathbf{T}_{i+k-1}\}$ . The local map is referenced to the local coordinate frame of the first scan in this local map, as highlighted in blue.

on these bird’s eye view images, as presented in Sec. III-D. To enable place recognition using these binary feature descriptors, we store them in a Hamming distance embedding binary search tree [50] for subsequent matching with query maps as presented in Sec. III-E. We obtain a set of candidate loop closures for new feature descriptors from a query local map by matching these feature descriptors to the database. We perform a loop closure validation and geometric alignment in 2D between the matched feature descriptors using a RANSAC scheme described in Sec. III-F. This 2D alignment and the ground alignment for each local map provide the relative 3D transform aligning each pair of loop-closed local maps.

#### A. Creation of Local Maps

Our approach uses local point cloud maps of the environment to perform loop closures. By aggregating downsampled LiDAR scans, we exploit the local consistency of sequential LiDAR odometry to generate these local maps. We use KISS-ICP [58] to obtain LiDAR odometry for our pipeline.

Given an online stream of sequential point clouds  $\{P_1, \dots, P_n, \dots\}$  in the LiDAR frame and their corresponding 3D pose estimates  $\{{}^w\mathbf{T}_1, \dots, {}^w\mathbf{T}_n, \dots\}$  in the odometry frame  $w$ , we first transform the point clouds into the odometry frame, resulting in point clouds  $\{P_1^w, \dots, P_n^w, \dots\}$ .

Starting from a point cloud with index  $i$ , we consider  $k$  subsequent scans until the displacement  $\|{}^w\mathbf{t}_{i+k-1} - {}^w\mathbf{t}_i\|_2$  exceeds a threshold  $\tau_c$ , where  ${}^w\mathbf{t}_i \in \mathbb{SE}(3)$  is the translational component of the pose estimate  ${}^w\mathbf{T}_i \in \mathbb{SE}(3)$ . Once we reach the sequence cut-off threshold  $\tau_c$ , we aggregate this subset of scans  $\{P_i^w, \dots, P_{i+k-1}^w\}$  into a local map  $M_m^w$  with  $m$  being the local map index. Precisely, we use a voxel grid-based downsampling strategy to generate the local map, using a resolution of  $\nu_{\text{map}}$  meters per voxel. Such a downsampling strategy also ensures a uniform spatial distribution of scanned points by retaining a maximum of 20 points per voxel. We repeat this process of local map generation sequentially, thereby generating mutually exclusive sets of consecutive point clouds, as shown in Fig. 3.

We transform each local map  $M_m^w$  into the local reference frame  ${}^w\mathbf{T}_i$  of the first scan in the sub-sequence as follows:

$$M_m^i = {}^w\mathbf{T}_i^{-1} M_m^w. \quad (1)$$

Such aggregation of LiDAR scans into local maps provides more structural details of the environment than the individual scans, as seen along each row of Fig. 4. Moreover, local maps from the same location, even when generated from LiDAR with different resolutions or scanning patterns, often have a higher similarity, as illustrated in Fig. 4. Such multi-LiDAR place recognition with only individual scans is a challenging data association task, even for human-level intelligence. Our approach can detect closures between local maps across multiple sessions recorded with different LiDARs, enabling cross-platform place recognition.

#### B. Ground Plane Detection and Alignment

After generating the point cloud local map  $M_m^i$ , we want to perform a BEV projection of the point cloud. Such a projection is consistent across multiple revisits only if we can perform the projection *on the same physical plane* in the environment. The ground plane is one such plane, which can be identified and computed consistently for the BEV projection. Given a planar motion of our LiDAR sensor and an extrinsic calibration, the ground plane is parallel to the xy-plane in the reference frame of the local map. However, in scenarios with non-planar motion of the LiDAR, the local ground plane will have a misalignment with respect to the xy-plane, as illustrated in Fig. 5. Therefore, we need to identify the ground plane and align it with the xy-plane of the reference frame.

The first step is identifying the ground within the local map to perform such an alignment. One way is to extract semantic information from point clouds using deep neural networks [41], [62]. These networks, which require an offline training step on a GPU, add a significant computational overload. Methods like Patchwork [31] and Patchwork++ [29] can segment ground points in real-time on a CPU. Still, they require a lot of parameter tuning and cannot perform segmentation of local maps, thereby requiring ground segmentation for each LiDAR scan. Furthermore, our pipeline requires only a single approximate ground plane in the scene for ground alignment rather than perfect ground segmentation.

We propose a ground points sampling strategy exploiting the fact that ground points are typically the lowest-lying points scanned by a LiDAR. We discretize the local map  $M_m^i$  along the local X and Y axis using a resolution of 5.0 m per cell and retain only one point per cell with the lowest z-coordinate. We show an example of our sampling strategy in Fig. 5.

After sampling the ground points  $G_m^i$ , we formulate a least-squares problem to align these points with the local xy-plane of the reference frame as follows:

$${}^g\mathbf{T}_{m,i} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_{\forall \mathbf{g}_k \in G_m^i} (\hat{\mathbf{n}}_{xy} \cdot \mathbf{T} \mathbf{g}_k)^2, \quad (2)$$

where,  $\mathbf{T} \mathbf{g}_k = [x'_k \ y'_k \ z'_k]^T$  is the point  $\mathbf{g}_k \in \mathbb{R}^3$  transformed by  $\mathbf{T} \in \mathbb{SE}(3)$ ,  $\hat{\mathbf{n}}_{xy}$  is the normal vector defining the xy-plane, and  $\cdot$  is the vector dot product.

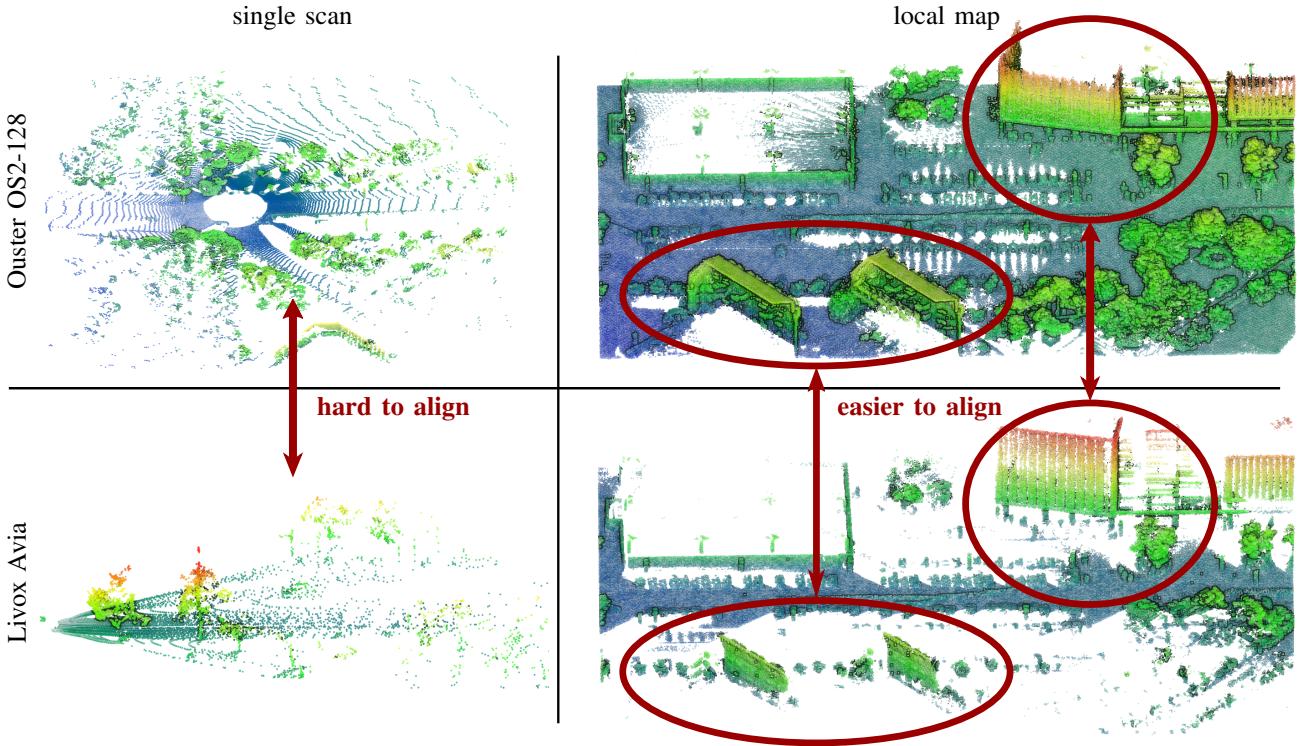


Fig. 4: A comparison of data from LiDAR sensors with different scanning patterns and field of views. The first row shows a single scan from an Ouster OS2-128 LiDAR with a  $360^\circ \times 45^\circ$  FoV and its corresponding local map. The second row shows a Livox Avia LiDAR with a  $70^\circ \times 77^\circ$  FoV and its corresponding local map. The similar structural features across local maps are highlighted with ellipses.

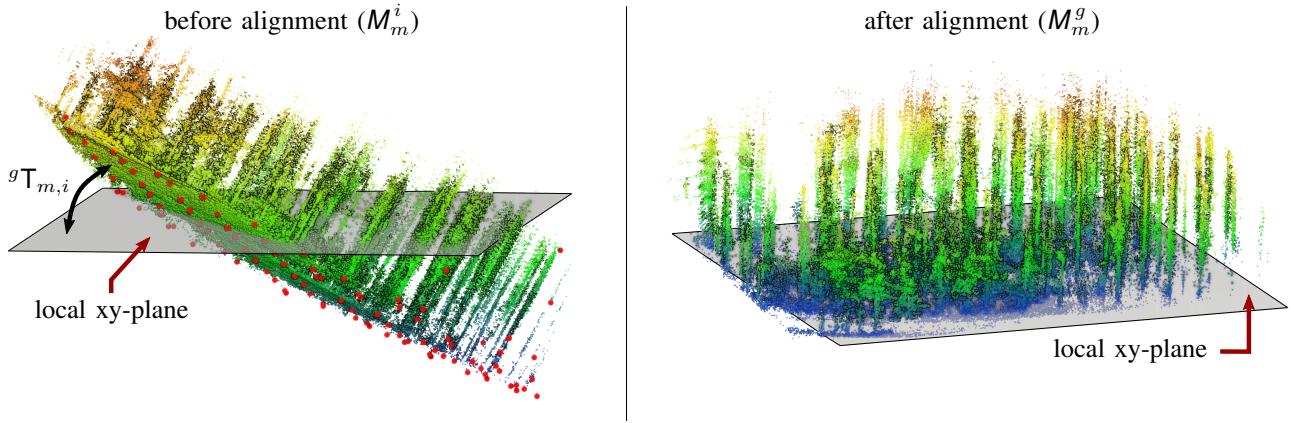


Fig. 5: Effect of ground alignment on a local map generated from a handheld LiDAR sensor with non-planar motion. The colors of the points represent their z-coordinates in the local reference frame. *Left:* Our ground alignment approach samples ground points  $G_m^i$  (highlighted in red) from the local map  $M_m^i$  and computes an alignment  ${}^9T_{m,i}$  to the xy-plane. *Right:* The ground-aligned local map  $M_m^g$ .

This formulation minimizes the sum of squared distances of the ground samples  $G_m^i$  to the local xy-plane. We perform this optimization iteratively for a maximum of 20 iterations or if the pose correction applied between successive iterations is less than some threshold. To reject the outliers arising from non-ground points, we use a binary weighting scheme based on the distance of each sampled point to the xy-plane.

The Jacobian  $J$  of the formulation in Eq. (2), with respect to the 3D transformation  $T \in \mathbb{SE}(3)$  is as follows:

$$J = [0 \ 0 \ 1 \ y'_k \ -x'_k \ 0]^T, \quad (3)$$

which is computed using a perturbation  $\Delta x^T = [\Delta t^T \ \Delta \omega^T]^T$

on the  $\mathbb{SE}(3)$  manifold, where  $\Delta t \in \mathbb{R}^3$  represents the translation and  $\Delta \omega \in \mathbb{R}^3$  represents the rotation in the axis-angle form. The Jacobian indicates that our least-squares formulation of the ground alignment provides a pose correction corresponding only to a translation along the Z-axis and a rotation around X-axis (roll) and Y-axis (pitch), as represented by the non-zero quantities in Eq. (3).

Finally, we transform the local map  $M_m^i$  by the ground aligning transform  ${}^9T_{m,i}$ , resulting in a transformed local map  $M_m^g$  with its ground plane overlapping with the xy-plane. We show an example of the proposed ground alignment strategy in Fig. 5. We provide a detailed analysis and evaluation

of the ground alignment strategy in Sec. V-C.

### C. Density-preserving Bird's Eye View Projection

After establishing the primary representation of the environment in Sec. III-A and Sec. III-B, the next step is to extract informative yet distinct local feature descriptors to perform place recognition for loop closures. Rather than computing features directly on the 3D point clouds of local maps, which is computationally expensive given the spatial extent of such point clouds, we use a 2D BEV projection of the local maps as an intermediate representation for loop closure detection.

We perform the BEV projection by simply dropping the z-coordinate of all the individual points in the ground-aligned local maps  $M_m^g$ . Furthermore, we discretize the  $\mathbb{R}^2$  projection space with a resolution of  $\nu_b$  meters. This results in a 2D Cartesian grid  $N_m(u, v)$  of size  $W_m \times H_m$  where,

$$W_m = \left\lfloor \frac{x_m^u}{\nu_b} \right\rfloor - \left\lfloor \frac{x_m^l}{\nu_b} \right\rfloor + 1, \quad (4)$$

$$H_m = \left\lfloor \frac{y_m^u}{\nu_b} \right\rfloor - \left\lfloor \frac{y_m^l}{\nu_b} \right\rfloor + 1, \quad (5)$$

$$\begin{bmatrix} x_m^u \\ y_m^u \end{bmatrix} = \max_{x,y} M_m^g; \quad \begin{bmatrix} x_m^l \\ y_m^l \end{bmatrix} = \min_{x,y} M_m^g. \quad (6)$$

However, the dimensionality reduction comes at the cost of losing the complete 3D information about the scene. Many traditional BEV projection approaches store the maximum elevation of the points in each cell [22], [25], [30], [35]. The maximum elevation, however, is sensitive to the distance between the scanner and the surface being scanned, as well as the LiDAR's field of view. In our pipeline, we instead store the *density of points* in each discrete 2D cell after the projection, which is less sensitive to viewpoint changes [34].

Therefore, each cell in this grid  $N_m(u, v) \in \mathbb{N}_0$  stores the count of projected points within that cell. The bird's eye view image  $I_m(u, v)$  of the local map  $M_m^g$  is then defined as the relative density of points in each cell of the grid as follows:

$$I_m(u, v) = \frac{N_m(u, v) - N_{\min}}{N_{\max} - N_{\min}} \in \mathbb{R}^{W_m \times H_m}, \quad (7)$$

$$N_{\max} = \max N_m(u, v), \quad N_{\min} = \min N_m(u, v). \quad (8)$$

We set all the image pixels  $N_m(u, v)$  with a relative density lower than 5% to be zero. This removes structures within the environment with a small vertical footprint, such as roads, vehicles, and pedestrians. Such features in the environment are often caused by dynamic or small objects that are unreliable for place recognition.

### D. Feature Detection and Pruning Strategy

The BEV projection makes feature detection for place recognition computationally efficient due to the reduction of the dimensions. The image-like BEV density representation also enables us to leverage established techniques from the computer vision domain to extract distinct feature descriptors for place recognition. Since the BEV density images preserve the geometrical floorplan-like structure of the environment, we use ORB [46] feature descriptors to capture features from

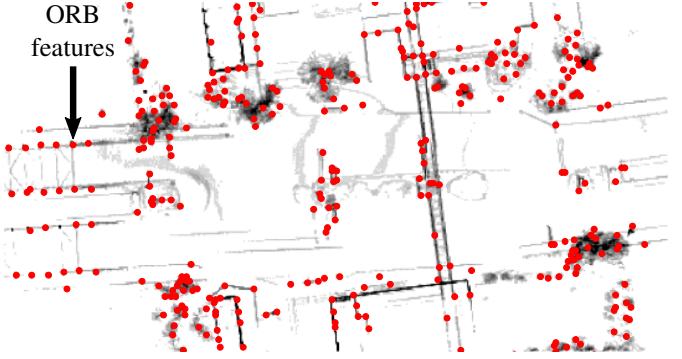


Fig. 6: A BEV density image of a local map with darker pixels corresponding to a higher density. The red dots are the ORB features computed from the density image.

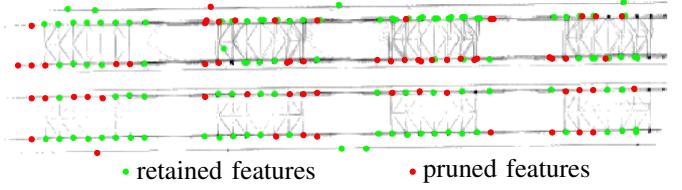


Fig. 7: An example of self-similarity feature pruning on the ORB feature descriptors computed on BEV density map of a bridge with repetitive mechanical structures. Our algorithm prunes the features in red while retaining only the green ones.

these images. These density images, which are an orthographic projection of the 3D world, have no scale ambiguity, unlike regular camera images. We leverage this fact and compute the ORB feature descriptors  $D_m$  without using the scale-invariance factors typically used for camera images, which also improves the computational efficiency of our pipeline. Fig. 6 shows the ORB features computed from a BEV density image.

Furthermore, we perform a pruning of the feature descriptors computed from each BEV density image, inspired by Bosse et al. [3]. This step performs a self-similarity check on each density image using the computed ORB feature descriptors. It prunes features within the same image that have similar descriptors. We set a threshold  $\tau_{pr} = 35$  based on the Hamming distance between the ORB descriptors to perform the self-similarity pruning. This approach is similar to the Lowe's ratio test [32] performed during feature descriptor matching between two images.

Such a pruning strategy removes the features recorded from repetitive structures in the environment, for example, a bridge with repetitive mechanical structures, as shown in Fig. 7. Feature pruning helps to reduce the number of false positive feature matches across density images resulting from perceptual aliasing, thereby reducing false positive loop closure detections in our pipeline. We provide a thorough analysis and evaluation of the effect of feature pruning on loop closure detection in Sec. V-D.

### E. Feature Database

Once we have the unique features within a BEV density image, we create a database to serve as a reference for place recognition. We leverage the binary domain of the

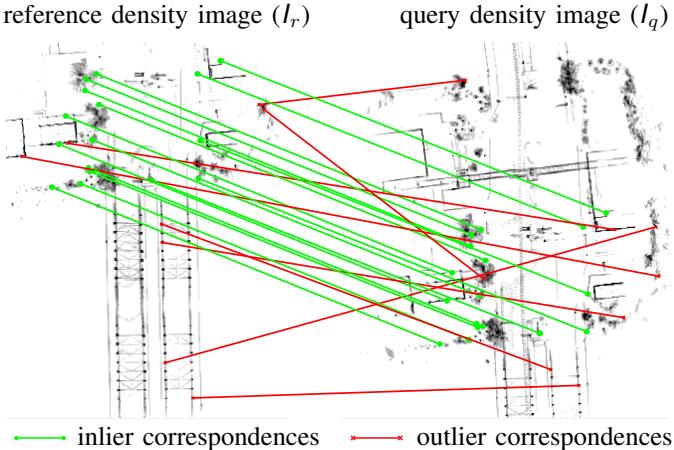


Fig. 8: ORB descriptor matches between features from a reference and a query BEV density image obtained from the HBST database. The inlier correspondences obtained from the RANSAC-based geometric verification are shown with green lines, and the outlier correspondences are shown with red lines.

ORB descriptors, using the Hamming distance embedding binary search tree or HBST [50] to store the set of feature descriptors  $D_m$  obtained from each density image  $I_m$  along with their corresponding map index  $m$ .

The depth of the HBST is limited by the number of bits in the binary descriptor (256). As a result, a query descriptor will require at most 256 bitwise comparisons with the tree’s nodes before reaching a leaf node. Each leaf node can hold a maximum of 100 descriptors, ensuring efficient feature matching. These design choices constrain the computational time for feature matching without imposing practical limitations on the usability of our approach. Unlike a clustering-based bag of words or a neural representation-based database, we use the HBST as an incrementally updated database, requiring no offline pre-training.

After obtaining a set of descriptors  $D_q$  from the query local map’s BEV density image  $I_q$ , we search the HBST database to find the closest match for each descriptor  $d_{j,q} \in D_q$ . Matches are determined using a Hamming distance threshold of 50 bits between ORB descriptors. Since the binary tree links each stored descriptor to an index  $m$  corresponding to its local map, we compile a list of feature matches between the query map and the reference maps in the database. These matches are subsequently verified geometrically. Fig. 8 visualizes the corresponding set of feature matches between two density images, as obtained from the HBST database.

#### F. Loop Detection and Map Alignment

The geometrical verification step entails performing a 2D alignment of the matched features. This involves calculating a 2D rigid body transformation that optimally aligns the matched features from the binary tree using a distance metric. Unlike the general image-alignment problem, this process is constrained to an  $\mathbb{SE}(2)$  transformation rather than a homography. To handle outlier associations caused by the limitations of binary tree-based matching, we employ a RANSAC-based alignment strategy.

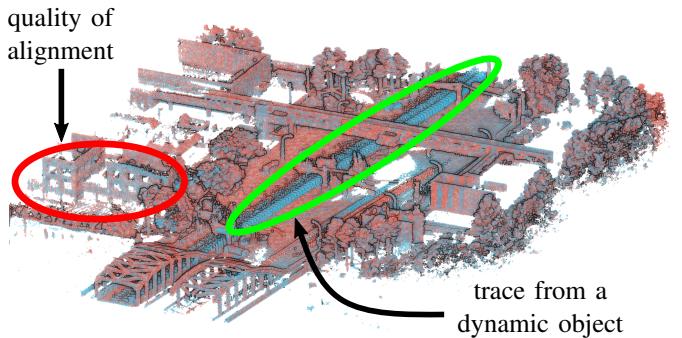


Fig. 9: Two local maps (in red and blue) detected as loop closure and aligned using the initial estimate  ${}^qT_r$  provided by our pipeline. Dynamic objects in the scene, as seen in the area highlighted by a green ellipse, do not affect the alignment quality provided by our approach. The red ellipse highlights the overlap of windows-panes on a building, achieved through our initial alignment  ${}^qT_r$  estimate.

We design a RANSAC-based approach that randomly selects two feature pairs from the set of matches between a query ( $I_q$ ) and a reference ( $I_r$ ) density image. We use the Kabsch-Umeyama algorithm [21], [56] to compute the relative 2D alignment between the feature pairs. We use this alignment to compute the point-wise Euclidean distance between each pair of feature matches. Feature matches with distances larger than 1.5 m (3 pixels for  $\nu_b = 0.5$  m) are considered outliers. The RANSAC process runs for a fixed number of iterations  $N_{\text{ransac}}$ . We require a minimum number of inliers  $\gamma$  from the RANSAC alignment stage to decide whether two local maps belong to the same location.

Finally, after the  $N_{\text{ransac}}$  iterations, we compute a Kabsch-Umeyama 2D alignment over the entire set of inlier correspondences. It provides us with a rotation matrix  $R \in \mathbb{SO}(2)$  and a translation vector  $t \in \mathbb{R}^2$ , which we represent as a homogenous transformation  $T_{\text{BEV}} \in \mathbb{SE}(3)$ , having translation components only along the X and Y axes, and a rotation component only about the Z axis. We also have to scale the translation vector by the voxel size ( $\nu_b$ ) since the features are computed from the discretized density images.

We then compose the associated transform between the two BEV density images with the ground aligning transforms of the two local maps as shown in Eq. (9) to get a complete 3D transform between local maps in their respective local frames.

$${}^qT_r = {}^qT_{q,i}^{-1} T_{\text{BEV}} {}^qT_{r,i}. \quad (9)$$

This 3D alignment is an initial guess for a fine registration of local maps performed during pose-graph optimization for drift correction. A qualitative example of the effectiveness of this 3D initial alignment is shown in Fig. 9.

#### IV. EXPERIMENTAL SETUP FOR EVALUATION

In this section, we provide an overview of our experimental setup used to evaluate our method and compare it to existing methods for detecting loop closures. First, we introduce the datasets we use for the experimental evaluation and benchmarking. Then, we provide information about the implementation details of the baseline methods against which we compare our approach. Finally, we introduce the quantitative metrics for evaluating our loop closure pipeline and other baselines.

## A. Datasets

We thoroughly evaluate our approach on datasets collected with various LiDARs with different resolutions and scanning patterns. These datasets have multiple sequences recorded in diverse urban environments using different mobile platforms.

*1) Public Datasets:* We use the MulRan [24] and HeLiPR [20] public datasets, recorded in urban environments from LiDARs mounted on the rooftop of a car. Both datasets are ideal for single-session loop closure evaluation with multiple in-sequence revisits with varying orientations and lane shifts. Also, MulRan and HeLiPR datasets contain at least 3 sequences in each diverse environment, so they are suitable for multi-session loop closure evaluation.

The MulRan [24] dataset is recorded from a 64 beam rotating LiDAR (Ouster OS1-64) at a frequency of 10 Hz. The LiDAR's horizontal FoV is limited to 290° due to the occlusion from a Radar sensor mounted behind it. We use this dataset's three environments: KAIST, Riverside, and Sejong. The HeLiPR [20] dataset contains multiple LiDARs with different FoVs, scanning patterns, and resolutions mounted on the rooftop of a car, making it suitable for evaluating multi-LiDAR place recognition. We use three of the four available LiDAR from this dataset: (1) a 128 beam spinning LiDAR (Ouster OS2-128) with a 360° × 45° FoV, (2) a hybrid solid-state LiDAR (Livox Avia) with a 70° × 77° FoV and an unusual non-repetitive scanning pattern, and (3) a solid-state LiDAR (Aeva Aeries II) with a 120° × 19.2° FoV. We do not use the fourth Velodyne VLP-16 LiDAR due to its self-occlusion with surrounding sensors [20]. The Bridge sequences suffer from strong perceptual aliasing on two long bridges, which are featureless in specific parts and have repetitive structures in other parts.

Furthermore, the MulRan and HeLiPR datasets have sequences recorded from the same environments, KAIST and Riverside, with a temporal gap of 4 years. This enables long-term place recognition between the MulRan and HeLiPR datasets on the KAIST and Riverside environments. These long-term revisits are also performed with different LiDAR sensors, making them more challenging. The KAIST and Riverside sequences also have some spatial overlap, providing an opportunity to evaluate multi-map alignment.

We also use two sequences from the NCLT [5] dataset recorded with a Velodyne HDL-32E LiDAR mounted on a Segway robot. In these sequences, the LiDAR has a non-planar motion compared to the previous datasets due to the inverted pendulum-like motion of the Segway platform. We select two sequences recorded from the same location but a year apart.

*2) Self-recorded Datasets:* In addition to the public datasets mentioned above, we also demonstrate the effectiveness of our approach on sequences recorded using our own instrumented vehicles and sensor platforms. First, we record a long semi-urban sequence from a rooftop mounted Ouster OS1-128 LiDAR on a Volkswagen Passat instrumented vehicle (Car). We drive through a hilly region, passing through streets with forests on either side, and perform multiple revisits from different viewpoints and elevations.

Furthermore, we use a sensor platform mounted on a backpack and walk around the city to record a sequence with

loop closures. This platform has a Hesai Pandar-128 LiDAR. It has a different view of the surroundings at a height of approximately 2 m to the ground and a non-planar motion of the LiDAR due to the walking motion. This is a challenging sequence in which we evaluate our ground alignment strategy under non-planar sensor motion. The two sequences also have a small physical overlap in their sensed environments. We use this overlap to evaluate multi-session loop closure detection and alignment between the sequences.

We generated the ground truth using an offline LiDAR bundle adjustment method with manual inspection of the results. It also uses RTK-GNSS information, if available, for precise geo-referencing. The required initial pose estimates were obtained by pose-graph fusion of LiDAR odometry [58] for local consistency, and either manual loop closures for the Backpack dataset or RTK-GNSS for the Car dataset for global consistency. The LiDAR bundle adjustment approach refines the initial poses by aligning all scans with each other.

## B. Baseline Methods Used for Comparison

We compare our approach against four baselines. All these baselines provide publicly available implementations that we used for the evaluation. We modified the source code only to return all possible loop closure candidates for each query scan instead of only the best candidate, ensuring a fair comparison for all methods. Unless otherwise mentioned, we set all the other parameters of the baselines to the default values provided in their respective implementations.

*1) Our Approach:* For our approach, we set the maximum range of each individual point cloud to 100 m. The travel displacement threshold  $\tau_c$  used to generate local maps is also set to 100 m. For the KISS-ICP [58] odometry, we use the default parameters provided in their latest open-source release. We use a voxel size of  $\nu_{\text{map}} = 1.0$  m for the voxel grid used to generate the local maps and a resolution of  $\nu_b = 0.5$  m for the density BEV images. We use the default parameters provided for ORB feature descriptors, except for the scale invariance-related parameters, which we deactivate since the BEV images are orthographic projections. We obtain feature matches from the HBST using a Hamming distance threshold of 50 bits. All other parameters of the HBST tree are set to the defaults. Finally, we predict that two local maps as loop closures if we obtain more than  $\gamma = 5$  inliers from the RANSAC-based alignment. For the multi-session experiments, we store the HBST database generated from the reference session as a binary file, and use it as the database for the query session.

Additionally, as our pipeline detects loop closures between local maps, we convert the detected loop closures between local maps to loop closures between individual scans in these local maps. This enables a comparison with the scan-level reference closures and other baselines that work on individual scans. To this end, we follow the approach from Gupta et al. [16]. We first apply the 3D pose correction  ${}^q T_r$  between the local maps to the individual poses comprising these maps, transforming them into a common reference frame. Then, we compute the Euclidean distance between all the pairs of poses between the two local maps. Finally, we use a distance

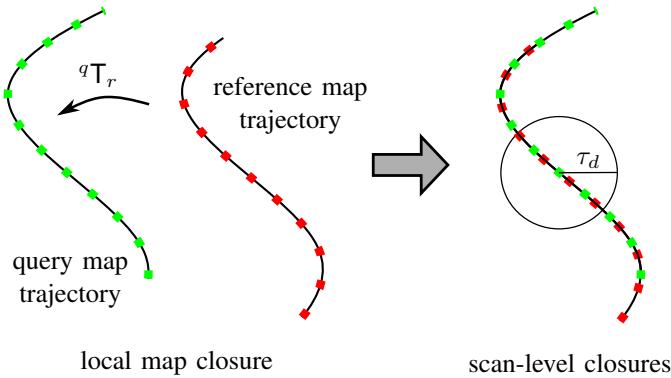


Fig. 10: An illustration of the conversion from loop closures between local maps to loop closures between individual scans in these local maps. *Left:* A pair of local maps identified as a loop closure. The scans within the query map are in green, and those within the reference map are in red. *Right:* We align the local map closure using the initial guess  ${}^q T_r$  from our pipeline. The pairs of scans between the two trajectories (red and green) with their poses within a distance threshold  $\tau_d$  are considered closures, as highlighted by black circles.

threshold  $\tau_d = 10$  m to classify a pair of individual scans as a loop closure. We show an illustration of this process in Fig. 10.

2) *Scan Context (SC)*: Scan Context [25] is a widely used, state-of-the-art approach for LiDAR-based loop closures and place recognition in SLAM. It computes a global descriptor for the BEV projection of each LiDAR scan to perform place recognition. However, it is designed for LiDARs with a large horizontal FoV and a cylindrical scanning pattern. As a result, we skip their evaluation for the Livox Avia and Aeva Aeries II from the HeLiPR dataset.

3) *Stable Triangle Descriptors (STD) and Binary Triangle Combined Descriptors (BTC)*: STD [68] proposes a triangle descriptor for describing point cloud sub-maps and extends it in BTC [67] with a binary descriptor for fast storage and retrieval of closure candidates. They propose accumulating 10 consecutive scans into a sub-map to compute the descriptors. We use KISS-ICP to obtain the pose estimates for creating the sub-maps. Furthermore, we convert the loop closures between sub-maps to loop closures between individual scans using the same approach we use for our pipeline.

4) *Spatially Organized and Lightweight Global Descriptor (SOLID)*: SOLID [26] is a recent approach for place recognition with 3D LiDAR that claims to work on various LiDAR irrespective of their geometric pattern and resolution. It is a lightweight global descriptor generated through point clouds represented in polar coordinates.

### C. Reference Loop Closures

#### 1) Reference Loop Closures Between Individual Scans:

A vast majority of previous works [18], [22], [25], [26], [68] consider a pair of LiDAR scans as a positive loop closure if the distance between their corresponding ground truth positions is below a certain threshold. This criterion, though, makes a wrong assumption that the LiDAR always observes the same objects while being at a similar location. It does not hold when other objects occlude the previously seen area or the sensor has

a restricted FoV and is revisiting the location from a different viewpoint. Furthermore, with the extended range of modern-day LiDAR, two scans not captured from the same location can still have sufficient overlap to perform place recognition.

Gupta et al. [16] suggested using the volumetric overlap of scans to decide if a loop should be closed. In this work, we extend this method for generating reference loop closures. The first step of our reference identification is to sample the reference trajectory at equidistant locations (2 m) to reduce the number of candidates. Then, we accumulate the scans between two sampled locations and generate a dense representation of the local environment in keyframe maps. We select all keyframes within the sensor’s maximum range for each query keyframe as potential reference loop closure candidates. We skip over 100 consecutive keyframes from the query keyframe to avoid short-distance closures.

Instead of directly computing the overlap between each keyframe pair, as proposed by Gupta et al. [16], we first perform an ICP registration [2], [7] to align the two keyframe maps. This overcomes any noise in the ground truth poses provided with the datasets. We use the point-to-point ICP registration pipeline from Open3D [71] with a maximum correspondence distance of 2 m. After the registration step, we treat the relative fitness score from the output of the registration algorithm as the overlap between the two keyframe maps. We consider two keyframes to be a loop closure if their overlap  $o > 0.5$ . Then, we use an exhaustive pairing of the individual scans comprising these keyframes as loop closures at the level of scan indices.

2) *Reference Loop Closures Between Local Maps*: We compute reference closures between local maps to perform ablation studies and evaluate our approach’s effectiveness on multi-session and multi-LiDAR loop closure scenarios. To this end, we generate the local maps in the same way as proposed in Sec. III-A. However, instead of using the odometry poses from KISS-ICP, we use the ground truth LiDAR poses provided with the datasets. We ensure that each ground truth local map contains the same scan indices as in the local maps generated by our approach so that the ground truth local maps directly correlate with the local maps computed using the odometry poses. Then, we consider all pairs of the local maps with a non-zero overlap as reference loop closures. The same criterion is used for generating reference loop closures between multiple sequences. However, in this case, all the sequences must have ground truth poses in a global reference frame.

### D. Evaluation Metrics

1) *Precision, Recall and F1 Score*: We use the reference closures computed in Sec. IV-C to evaluate our approach and other baselines quantitatively. In particular, we compute the precision-recall curve by varying the threshold  $\gamma$  on the number of inlier feature descriptor matches obtained from our pipeline’s RANSAC-based geometric verification stage. Similarly, we generate the precision-recall curve for other baselines by varying their key threshold parameters. Furthermore, we also compute the area under the precision-recall curve, i.e.,

average precision, the recall score at 100% precision, and the maximum F1 score, i.e., the harmonic mean between precision and recall values.

2) *Relative Fitness Score*: We evaluate the effectiveness of the 3D alignment obtained from our loop closure pipeline by computing the overlap between local map pairs involved in the loop closure. We use only the estimate  ${}^qT_r$  to align the local maps and do not perform any ICP registration between them. We use the *evaluate\_registration* function from Open3D [71] library with a maximum correspondence distance of 1 m to compute the overlap between the aligned local maps in terms of the relative fitness score provided by the function.

3) *Absolute Trajectory Error*: Finally, we evaluate our approach's effectiveness in performing drift correction in a SLAM pipeline. To this end, we perform an offline pose-graph optimization using the g2o [28] optimizer. We directly incorporate the detected loops between local maps, using the initial guess provided by our pipeline as the loop closure constraint in the pose-graph. We compute the absolute trajectory error with respect to the ground truth poses before and after the pose-graph optimization. Furthermore, we do not use any robust kernel in the pose-graph for outlier rejection. This allows us to test the detected loop closures' accuracy and alignments.

## V. EXPERIMENTAL EVALUATION

The main focus of this work is an accurate and effective loop closure detection pipeline that works with various LiDAR sensors, invariant to their scanning pattern, field of view, and viewpoint. We present our experiments to show the capabilities of our method. The results of our experiments also support our key claims, which are: (1) our approach can detect loop closures between local maps generated from various LiDAR sensors with different scanning patterns, FoVs, and resolutions; (2) our approach can perform multi-session loop closure detection and alignment with long-term revisits; (3) our approach also works with handheld platforms having non-planar motion in the LiDAR sensor frame and provides a complete 3D rigid body transform to align the detected loop closures; (4) our approach is robust against perceptual aliasing in environments with repetitive structures; (5) our approach can detect loop closures between sequences having minimal overlap, recorded with different LiDAR sensor platforms, enabling cross-platform multi-map alignment.

### A. Single-Session Loop Closure Detection and Alignment

In this first experiment, we evaluate the performance of our approach for single-session loop closure detection. We also compare our performance against other state-of-the-art baselines using the reference closures between individual scans, as computed in Sec. IV-C1. The precision-recall curves of all the methods on multiple datasets are shown in Fig. 11. As can be seen, our method consistently outperforms the other state-of-the-art baselines on a variety of datasets recorded from different LiDAR sensors. Note that the overall recall rates for all methods are relatively low. This is because the overlap-based reference closure computation in Sec. IV-C1 results in a considerable number of scan-level reference closures.

Furthermore, a higher precision is prioritized over a higher recall in the context of loop closures for SLAM.

Performance across all baselines is comparable in less challenging urban scenarios, such as the KAIST01 and Riverside01 sequences from the MulRan dataset. However, for some baselines, a performance drop can be seen in challenging urban sequences from the HeLiPR dataset, such as the Bridge01 and Town03. Due to our feature descriptor pruning strategy, we achieve a very high precision on the Bridge01 sequence, which has many repetitive structural features in the environment that can lead to false positives in place recognition. This high precision is also maintained across LiDAR sensors with different scanning patterns and FoVs, showcasing the versatility of our approach using local maps.

We can make a similar observation for the Town03 sequence, one of the most challenging sequences of the HeLiPR dataset due to the tight alleyways and several dynamic objects. We detect highly precise loop closures and a higher recall than all other baselines. Our approach outperforms all other baselines on the MulRan Sejong01 sequence, recorded on a highway with few structural features and a challenging low-overlap revisit coupled with a lane shift. This is mainly achieved due to data aggregation through local maps, which allows for detecting sufficient features even in such sparse environments. The high precision on the self-recorded Car sequence also showcases the effectiveness of our approach in semi-urban scenes with in-sequence elevation changes.

Furthermore, due to the ground alignment strategy proposed in Sec. III-B, we perform better on datasets with a non-planar motion of the LiDAR sensor, such as the NCLT and the self-recorded Backpack sequence. Scan Context, also a BEV projection-based method, performs worse on these datasets recorded in dense urban environments. This is mainly due to the inconsistent BEV projection of point clouds due to revisits with varying orientations along the LiDAR pitch and roll axis.

We report the average precision (AP), recall at 100% precision (R@1), and maximum F1 score ( $F1_m$ ) for all the baselines in Tab. I. Our loop closure detection and alignment pipeline is robust across various scenes and LiDAR sensors. We have a higher average precision amongst all the baselines and achieve the best maximum F1 score among them.

Our approach and BTC are the better-performing approaches across all the precision-recall evaluations in Fig. 11 and Tab. I. Notably, both approaches use local maps to detect loop closures and perform a 3D alignment using local features. Scan-based approaches such as Scan Context and SOLiD perform poorly in almost all the sequences. Their low precision on sequences such as Sejong01, Town03, and Bridge01 is due to the revisits in these sequences, which have a larger lateral and rotational shift than Riverside01 and KAIST01. Place recognition with significant viewpoint changes upon revisit is challenging to detect using scan-wise descriptors.

This thorough experimental evaluation supports our main claim that our approach can detect and align loop closures using various LiDAR sensors with different scanning patterns, FoVs, and resolutions.

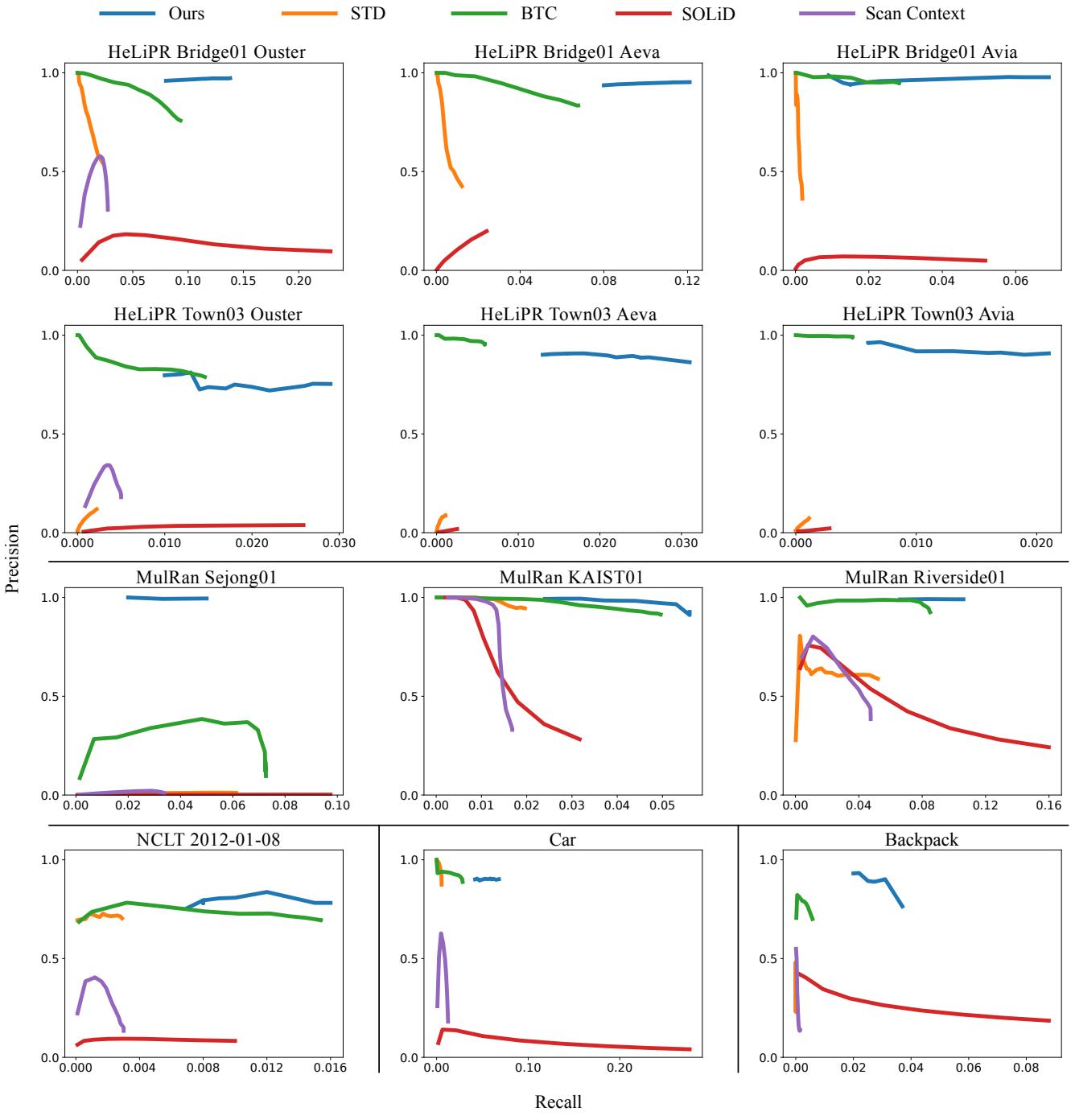


Fig. 11: The precision-recall curves of state-of-the-art baselines and our approach for single-session loop closure detection.

### B. Multi-Session and Multi-LiDAR Loop Closure Detection and Alignment

In this experiment, we evaluate the performance of our approach for multi-session and multi-LiDAR loop closure detection. First, we evaluate the performance of our approach on detecting closures across multiple sessions recorded with the same LiDAR. We provide in Tab. IIa the precision, recall, and F1 scores of the detected closures on the level of local maps. We can identify multi-session loop closures at very high precision, irrespective of the revisit interval and the LiDAR scanning pattern and FoV.

We see that LiDARs with smaller FoVs such as Livox Avia

and Aeva Aries II can detect fewer closures on sequences with a revisit in the opposite direction with respect to the reference session, for example, the Bridge01-Bridge03 sequence from HeLiPR dataset. In comparison, the Bridge02-Bridge03 sequences have a better recall score for all LiDARs due to the same direction of revisit. This effect can also be seen on sequences from the MulRan dataset, which is recorded using an Ouster OS1-64 LiDAR with an obstructed FoV. The recall on the KAIST01-KAIST03 sequences revisited from the same direction is substantially larger than the Sejong01-Sejong03 sequences revisited from the opposite direction.

Among the sequences from the HeLiPR dataset, the Livox

TABLE I: Average precision (AP), recall at 100% precision (R@1), and maximum F1 scores ( $F1_m$ ) of state-of-the-art baselines and our approach for single-session loop closure detection. Larger values are better. The best values are in bold.

(a) MulRan → Ouster OS1-64 LiDAR with  $290^\circ \times 45^\circ$  FoV

Datasets		KAIST01			KAIST02			Riverside01			Riverside02			Sejong01		
Metrics		AP	R@1	$F1_m$												
SC		0.015	0.032	0.003	0.023	<b>0.047</b>	0.003	0.031	0.011	0.085	0.014	0.016	0.059	0.000	0.029	0.024
STD		0.019	0.009	0.038	0.021	0.004	0.042	0.033	0.003	0.096	0.020	0.002	0.054	0.001	0.050	0.019
BTC		0.048	0.005	0.094	0.066	0.002	0.126	0.084	0.003	0.156	0.141	0.108	0.250	0.027	0.048	<b>0.115</b>
SOLiD		0.018	<b>0.057</b>	0.002	0.023	0.002	0.066	0.066	0.008	<b>0.193</b>	0.038	0.054	0.180	0.001	<b>0.097</b>	0.004
Ours		<b>0.055</b>	0.032	<b>0.105</b>	<b>0.088</b>	0.045	<b>0.162</b>	<b>0.105</b>	<b>0.083</b>	0.192	<b>0.251</b>	<b>0.219</b>	<b>0.401</b>	<b>0.050</b>	0.02	0.096

(b) HeLiPR → Ouster OS2-128 LiDAR with  $360^\circ \times 45^\circ$  FoV

Datasets		Bridge01			Bridge02			Roundabout01			Town02			Town03		
Metrics		AP	R@1	$F1_m$												
SC		0.015	0.021	0.051	0.007	0.010	0.028	0.001	0.002	0.006	0.001	0.002	0.007	0.002	0.003	0.010
STD		0.016	0.001	0.043	0.006	0.000	0.023	0.000	0.004	0.007	0.000	0.001	0.001	0.000	0.002	0.004
BTC		0.084	0.000	0.166	0.064	0.000	0.133	0.056	0.041	0.156	0.003	0.008	0.016	0.012	0.000	0.027
SOLiD		0.031	0.043	0.135	0.037	0.049	0.168	0.085	0.222	0.259	0.000	0.012	0.017	0.001	<b>0.026</b>	0.031
Ours		<b>0.134</b>	<b>0.138</b>	<b>0.242</b>	<b>0.112</b>	<b>0.087</b>	<b>0.201</b>	<b>0.313</b>	<b>0.280</b>	<b>0.475</b>	<b>0.027</b>	<b>0.016</b>	<b>0.055</b>	<b>0.023</b>	0.013	<b>0.056</b>

(c) HeLiPR → Aeva Aeries II LiDAR with  $120^\circ \times 19.2^\circ$  FoV

Datasets		Bridge01			Bridge02			Roundabout01			Town02			Town03		
Metrics		AP	R@1	$F1_m$												
STD		0.007	0.000	0.024	0.002	0.000	0.010	0.000	0.002	0.004	0.000	0.000	0.000	0.000	0.001	0.002
BTC		0.062	0.001	0.125	0.028	0.000	0.067	0.041	0.000	0.125	0.000	0.002	0.004	0.006	0.000	0.012
SOLiD		0.005	0.024	0.043	0.007	0.022	0.048	0.004	0.042	0.060	0.000	0.000	0.000	0.000	0.003	0.004
Ours		<b>0.115</b>	<b>0.121</b>	<b>0.215</b>	<b>0.111</b>	<b>0.080</b>	<b>0.201</b>	<b>0.219</b>	<b>0.220</b>	<b>0.360</b>	<b>0.017</b>	<b>0.016</b>	<b>0.034</b>	<b>0.028</b>	<b>0.018</b>	<b>0.059</b>

(d) HeLiPR → Livox Avia LiDAR with  $70^\circ \times 77^\circ$  FoV

Datasets		Bridge01			Bridge02			Roundabout01			Town02			Town03		
Metrics		AP	R@1	$F1_m$												
STD		0.001	0.000	0.004	0.001	0.000	0.007	0.000	0.003	0.005	0.000	0.000	0.000	0.000	0.001	0.002
BTC		0.027	0.001	0.055	0.017	<b>0.019</b>	0.039	0.033	0.065	0.115	0.000	0.001	0.002	0.005	0.000	0.009
SOLiD		0.003	<b>0.013</b>	0.050	0.003	0.012	0.040	0.000	0.005	0.007	0.000	0.000	0.001	0.000	0.003	0.005
Ours		<b>0.068</b>	0.009	<b>0.128</b>	<b>0.060</b>	0.006	<b>0.116</b>	<b>0.254</b>	<b>0.230</b>	<b>0.404</b>	<b>0.013</b>	<b>0.013</b>	<b>0.026</b>	<b>0.020</b>	<b>0.007</b>	<b>0.041</b>

(e) NCLT → Velodyne HDL-32E; Car → Ouster OS1-128; Backpack → Hesai Pandar-128

Datasets		NCLT 2012-01-08			NCLT 2013-04-05			Car			Backpack		
Metrics		AP	R@1	$F1_m$	AP	R@1	$F1_m$	AP	R@1	$F1_m$	AP	R@1	$F1_m$
SC		0.001	0.001	0.006	0.000	0.001	0.003	0.006	0.005	0.024	0.000	0.000	0.003
STD		0.002	0.000	0.006	0.000	0.000	0.002	0.005	0.000	0.011	0.000	0.000	0.000
BTC		0.011	0.003	0.030	0.002	0.000	0.010	0.026	0.000	0.055	0.004	0.001	0.012
SOLiD		0.001	0.003	0.018	0.000	<b>0.005</b>	0.008	0.019	0.006	0.092	0.021	0.001	<b>0.119</b>
Ours		<b>0.013</b>	<b>0.012</b>	<b>0.032</b>	<b>0.007</b>	<b>0.005</b>	<b>0.019</b>	<b>0.062</b>	<b>0.059</b>	<b>0.127</b>	<b>0.033</b>	<b>0.022</b>	0.071

Avia LiDAR has the worst performance in terms of recall rates and F1 scores. This is mainly due to the small horizontal field of view of the LiDAR. It cannot detect features from the urban structures typically located laterally to the direction of motion of the vehicle. However, on the Town01-Town03 sequences from the HeLiPR dataset, the three LiDARs have a comparable recall rate. This is mainly due to the presence of narrow alleyways in these sequences. In such scenarios, the large FoV LiDAR does not provide more information to the local maps on the lateral direction as compared to the small FoV LiDARs due to the close proximity of structures.

We also detect long-term multi-session closures with perfect precision on the NCLT dataset, which was recorded with a low-resolution Velodyne HDL-32E LiDAR. This result also shows the effectiveness of our ground alignment strategy as the LiDAR has a non-planar motion since it is mounted on a Segway platform.

Additionally, we evaluate the performance of our approach detecting closures across multiple sessions recorded with different LiDARs. We provide in Tab. IIb the precision, recall, and F1 scores of the detected closures on the level of local maps. We can detect loop closures with very high precision,

TABLE II: The number of loop closures detected by our approach between multiple sessions with their precision, recall, and F1 scores.

(a) Same LiDAR sensor between multiple sessions. (O → Ouster OS2-128; A → Aeva Aeries II; L → Livox Avia; V → Velodyne HDL-32E)

Datasets	MulRan			HeLiPR						NCLT
Revisit Interval	≈ 2 months		2 weeks		4 weeks			4 weeks		
Reference Session	KAIST01	Sejong01		Bridge02		Bridge01		Town01		2012-01-08
Query Session	KAIST03	Sejong03		Bridge03		Bridge03		Town03		2013-04-05
LiDAR	Ouster OS1-64	O	L	A	O	L	A	O	L	A
# Closures	273	39	363	199	342	277	90	140	195	117
Precision	1.0	1.0	1.0	1.0	0.997	1.0	0.989	0.993	0.944	0.940
Recall	0.273	0.029	0.205	0.181	0.311	0.117	0.063	0.094	0.148	0.172
F1	0.428	0.057	0.341	0.307	0.474	0.209	0.119	0.172	0.255	0.290
										V

(b) Different LiDAR sensors between multiple sessions. (O → Ouster OS2-128; A → Aeva Aeries II; L → Livox Avia)

Datasets	HeLiPR								MulRan × HeLiPR	Self-recorded
Revisit Interval	2 weeks	4 weeks	≈ 2 weeks	2 weeks	4 weeks	> 4 years			2 weeks	
Reference Session	Bridge02	Bridge01	Roundabout01	Town02	Town01	MulRan	KAIST01		Backpack	
Query Session	Bridge03	Bridge03	Roundabout02	Town03	Town03	HeLiPR	KAIST05		Car	
Reference LiDAR	O	O	O	O	O	O	L	Ouster OS1-64	Hesai Pandar-128	
Query LiDAR	A	A	A	A	A	A	A	O	Ouster OS1-128	
# Closures	55	21	62	5	21	22	20	23	3	12
Precision	1.0	1.0	1.0	1.0	0.917	0.955	0.95	1.0	1.0	1.0
Recall	0.039	0.011	0.052	0.006	0.020	0.030	0.023	0.029	0.022	0.003
F1	0.075	0.022	0.098	0.013	0.040	0.058	0.044	0.056	0.043	0.006
										0.112

although with a relatively low recall due to the comparison between LiDARs with vastly varying FoVs across sequences with a minimum revisit interval of 2 weeks.

The effect of the direction of revisit in the context of low FoV LiDARs can also be seen in the multi-session scenario. The Bridge02-Bridge03 sequences have a higher recall compared to the Bridge01-Bridge03 sequences. For the Roundabout01-Roundabout02, Town01-Town03, and Town02-Town03 sequences, we can even detect loop closures between two low FoV LiDARs with a very high precision.

We also evaluate our approach to detecting loop closures between sequences from the MulRan and HeLiPR datasets with a large revisit interval of 4 years and recorded with different LiDARs. Multi-session closures across such sequences are very challenging due to structural changes in the environment. As seen in the second to last column of Tab. IIb, our approach can detect closures between the KAIST01 and the KAIST05 sequences with a 1.0 precision. These sequences' recall rates are low due to the widely varying FoVs of the LiDARs used in the two sequences.

Finally, we also perform multi-session loop closure detection on the self-recorded datasets, which were recorded two weeks apart from mobile platforms having different motion profiles, different LiDARs, and different viewpoints of the environment. Our approach successfully detects closures between these two sequences, also with a 1.0 precision.

This evaluation showcases the versatility of our approach in detecting loop closures between multiple sequences, recorded with different mobile platforms, different LiDAR sensors, and having large revisit intervals, thereby supporting our second key claim.

### C. Analysis and Evaluation of the Ground Alignment Module

In this section, we provide a qualitative and quantitative evaluation of the effectiveness of the ground alignment stage for detecting and aligning loop closures. As explained in Sec. III-B, we need the ground alignment stage in scenarios where the LiDAR has a non-planar motion to have a consistent reference plane for the BEV projection across revisits.

Without the ground alignment module, our approach performs a BEV projection on the xy-plane, assuming a planar motion. As a result, it can only align the detected loop closures in 2D along the xy-plane using the RANSAC-based geometric verification. We show an example at the top-left image of Fig. 12 where two local maps from a loop closure are aligned from a bird's eye view. However, this initial 2D alignment, as seen from the top view, can be misleading.

Instead, there can exist a significant misalignment between the local maps, evident from another perspective, as seen on the top right of Fig. 12. Such a misalignment occurs when the local maps have a relative pitch or roll with respect to each other due to the non-planar motion of the LiDAR frame. However, with the explicit ground alignment of the local maps, we can now perform loop closures along with a complete 3D global alignment. We illustrate this in the second row of Fig. 12. This results in a better initial overlap between the local map pairs involved in a loop closure without a point cloud registration.

We also quantitatively evaluate the relative alignment between detected loop closures. In particular, we compute the relative fitness of the local maps involved in each closure after aligning them with our initial guess  ${}^q T_r$ . To this end, we choose two sequences with a non-planar motion in the LiDAR frame: one sequence (2013-04-05) from the NCLT [5]

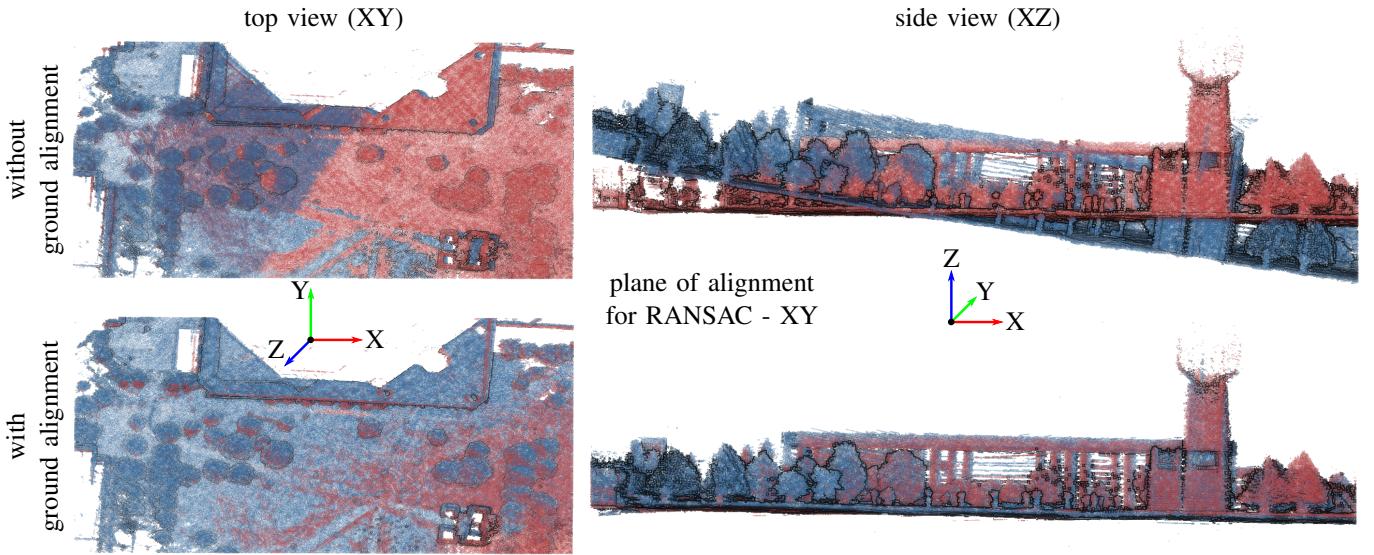


Fig. 12: Effect of the ground alignment strategy on the quality of alignment  ${}^gT_r$  between a pair of loop-closed local maps. The two local maps look aligned in the top view irrespective of the ground alignment, as seen on the left. However, the side view shows the misalignment between the loop-closed local maps when not ground-aligned.

TABLE III: Quantitative evaluation of the effect of ground alignment on loop closure detection and alignment. We report the number of loop closures detected by our approach with and without the ground alignment strategy. We also report the average relative fitness score of the loop-closed local maps aligned using the pose estimate  ${}^gT_r$  from our approach and the ground truth (GT) aligned local maps.

Dataset	NCLT 2013-04-05		Backpack		
	Ground Alignment Module	Off	On	Off	On
# Closures		7	9	7	23
Avg. Rel. Fitness (with ${}^gT_r$ ) $\uparrow$		0.24	0.49	0.27	0.68
Avg. Rel. Fitness (with GT) $\uparrow$		0.44	0.53	0.75	0.70

TABLE IV: Quantitative evaluation of ground alignment for varying magnitudes of misalignments between the ground plane and xy-plane in the local maps from sequences in the MulRan dataset. All values are in degrees.

Sequence	KAIST01	Riverside01	Sejong01
No. of Local Maps	58	62	231
Input Rotation [°]	Average Error in Predicted Rotation [°]		
10	0.01	0.03	0.07
20	0.04	0.05	0.13
30	0.07	0.07	0.31
40	0.11	0.10	0.58
50	0.41	0.29	1.51
60	2.96	0.89	4.53

dataset and one sequence recorded from the Backpack setup. In Tab. III, we show the number of closures detected and their average relative fitness scores for the configurations with and without the ground alignment. We also provide the average ground truth relative fitness score computed from the ground truth local maps of the detected closures. All the closures reported in Tab. III have a 1.0 precision concerning the respective dataset’s local map reference closures.

It can be seen in Tab. III that using the ground alignment approach, we doubled the overlap score between loop closures. This improvement in relative fitness, achieved without any

fine point cloud registration, is very close to the ground truth relative fitness. Furthermore, we improve our recall rate regarding closures between local maps. Notably, in datasets with exaggerated non-planar motions, like the Backpack with approximately a  $-30^\circ$  to  $30^\circ$  variation in ground truth roll and pitch angles, we detect significantly more closures using the proposed ground alignment module. This is because, in the presence of non-planar motion of the LiDAR, the BEV density images are consistent across revisits when the projection is on the physical ground plane in the environment as against the local xy-plane in the local map frame.

Finally, we show the ability of our ground alignment module to compensate for varying magnitudes of misalignments between the ground plane and the local xy-plane of the map. We manually induce a misalignment to each local map by applying rotations around ten randomly chosen axes lying on the xy-plane with magnitudes within a range ( $10^\circ$  -  $60^\circ$ ). Then, we compute the transform  ${}^gT_{m,i}$  to align the ground plane with the xy-plane using our approach. We perform this experiment on the MulRan KAIST01, Riverside01, and Sejong01 sequences, which have a locally-planar motion, to ensure that the reference local maps have their ground plane parallel to the xy-plane. In Tab. IV, we report the quality of the ground alignment module in terms of the error in predicted rotation magnitudes, averaging over all the ten rotations for all local maps in each sequence. Our ground alignment strategy works reliably up to  $50^\circ$  misalignment between the ground plane and the xy-plane. Even for a  $60^\circ$  input rotation, the maximum average error in misalignment is less than  $5^\circ$ . This performance is sufficient for any practical application that will not have such high magnitudes of misalignment between the ground plane and the xy-plane.

This experiment supports our third claim that our approach works with handheld platforms having a non-planar motion in the LiDAR frame and provides a 3D rigid body transform to align the loop closures.

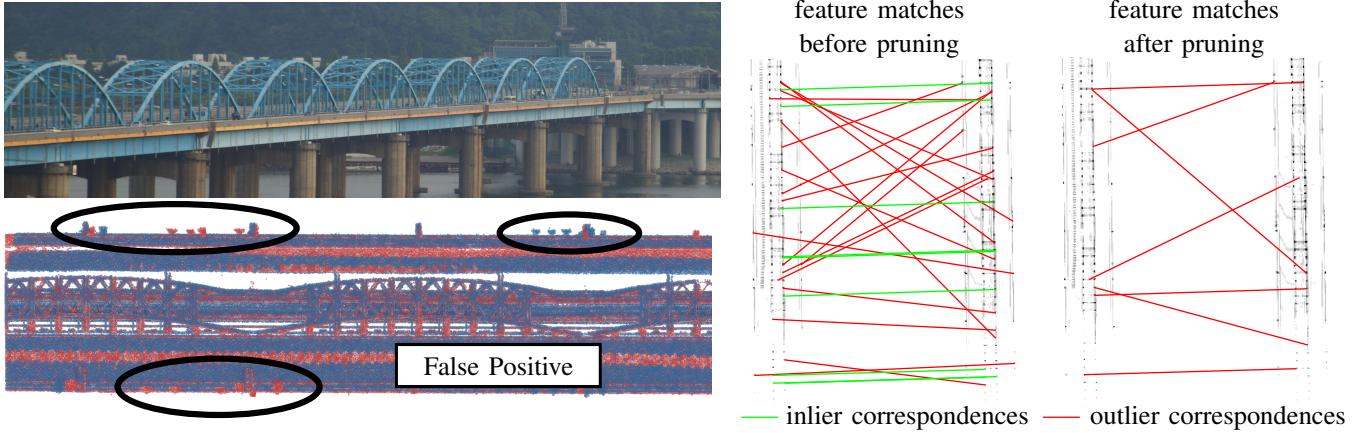


Fig. 13: Perceptual aliasing from the HeLiPR Bridge sequence. *Left:* The semi-circular arches of the bridge (top) from a repetitive pattern leading to a false loop closure (below) by our approach without the feature pruning strategy. As highlighted with black ellipses, the two local maps from this closure have very high structural similarity and only a few non-overlapping structures. *Right:* Without feature pruning, the RANSAC-based geometric verification process detects sufficient inlier matches (green lines), resulting in a false closure. In contrast, no inlier matches are detected when we prune the features from each BEV density image before inserting them into the HBST database.

TABLE V: Quantitative evaluation of the effect of feature pruning on loop closure detection, in terms of the precision (P) and recall (R) values of closures at the level of local maps. A higher value indicates better performance on the metric.

Sequence	HeLiPR Bridge01						HeLiPR Bridge02					
	Ouster OS2-128		Aeva Aeries II		Livox Avia		Ouster OS2-128		Aeva Aeries II		Livox Avia	
Metric	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
w/o Pruning	0.96	<b>0.22</b>	0.98	<b>0.25</b>	0.93	<b>0.21</b>	0.99	<b>0.28</b>	0.96	<b>0.32</b>	0.91	<b>0.22</b>
w/ Pruning	<b>1.0</b>	0.20	<b>1.0</b>	0.21	<b>1.0</b>	0.12	<b>1.0</b>	0.25	<b>1.0</b>	0.29	<b>1.0</b>	0.16

#### D. Analysis and Evaluation of the Self-Similarity Pruning Strategy

In this section, we analyze and evaluate the effectiveness of the self-similarity pruning strategy for detecting loop closures in environments with highly repetitive structures. We perform this study on the Bridge01 and Bridge02 sequences from the HeLiPR [20] dataset. These sequences contain two long bridges with repetitive mechanical structures leading to perceptual aliasing. One such bridge, a part of these sequences, can be seen at the top-left of Fig. 13.

Our approach detects false loop closures on such sequences without the pruning strategy. An example of such a false positive closure is seen at the bottom-left of Fig. 13, which poses a significant challenge in place recognition due to the almost perfect alignment between local maps. This is because of the sufficient inlier matches after the 2D RANSAC alignment between their BEV features, as seen on the right-hand side of Fig. 13. These matches are mainly between the repetitive structures in the scene. We successfully eliminate these inlier feature matches by the feature pruning strategy proposed in Sec. III-D, thereby removing the false loop closure.

We compare our pipeline’s performance with and without the feature pruning strategy in Tab. V. We report the precision and recall values over all the detected loop closures at the level of local maps. With the activated feature pruning stage, we have a perfect 1.0 precision on all the Bridge sequences from the HeLiPR dataset for all three LiDAR sensors. In contrast, some false positives are always detected when deactivating the feature pruning stage. We have a slightly lower recall when

using the pruning strategy. This is expected as the pruning of features will eliminate all loop closures in areas with perceptual aliasing since it is hard to differentiate between a true and a false closure in such situations. As a result, we miss some of the true loop closures on the bridges.

Finally, through a pose-graph optimization study, we emphasize the benefits of having a 1.0 precision, even if at the cost of lower recall. We perform a pose-graph optimization using the pose constraints from the detected loop closures, with and without the pruning strategy. It can be clearly observed in Tab. VI that loop closures obtained with the pruning strategy lead to a significantly better absolute trajectory estimate after pose-graph optimization despite the lower number of detected closures. On the other hand, the false loop closures obtained by deactivating the pruning strategy lead to a larger error in the final trajectory estimate, even more considerable than the odometry-only error. This study supports our fourth claim that the feature pruning strategy proposed in Sec. III-D prevents our pipeline from failing in scenes with high self-similarity.

#### E. Multi-Map Alignment

In this final experiment, we detect loop closures between the Riverside03 sequence and the three KAIST sequences, respectively, from the MulRan dataset. These two trajectories have a minimal overlap, allowing them to align with each other. We first compute the optimized trajectory for each sequence using the in-session loop closures constraints provided by our approach through a pose-graph optimization. Then, we align the two pose-graph optimized sequences using the loop

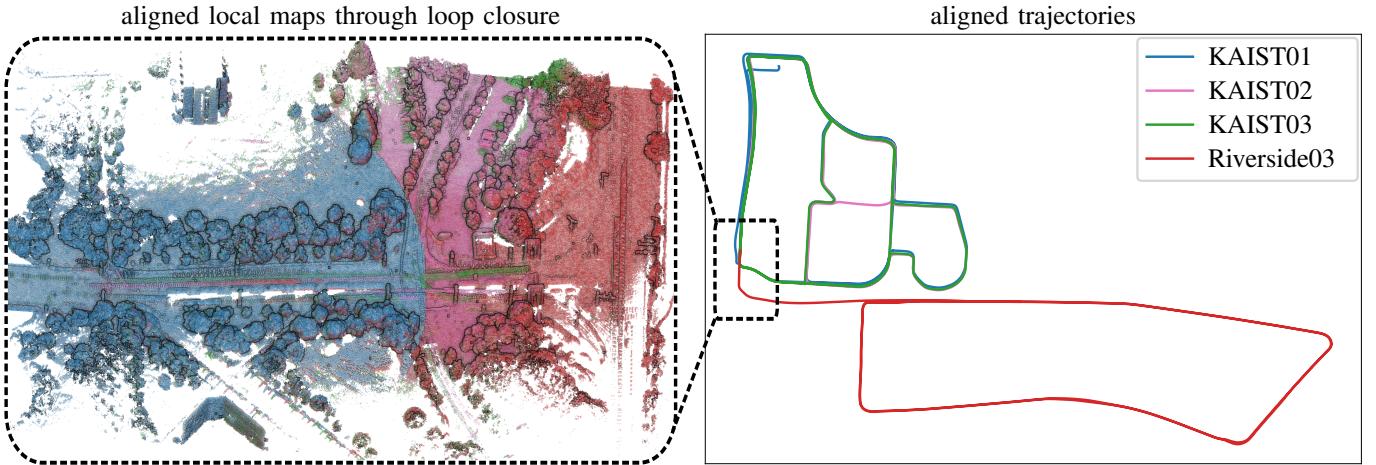


Fig. 14: We use our loop closure pipeline to align the three KAIST sequences respectively to the Riverside03 sequence from MulRan dataset. The overlap between these sequences is minimal, as seen in the area highlighted with a dashed rectangle. Our approach successfully detects closures in this area with a good alignment between the local maps.

TABLE VI: Quantitative evaluation of the effect of feature pruning on loop closure detection, in terms of the root-mean-square absolute trajectory error in translation after pose-graph optimization. All quantities are in meters. A lower value indicates better performance on the metric.

Dataset	HeLiPR Bridge01			HeLiPR Bridge02		
	Ouster	Aeva	Livox	Ouster	Aeva	Livox
Metric	RMS Absolute Trajectory Error [m]					
w/o g2o	183.8	104.9	198.9	59.7	136.9	87.7
w/o Pruning	587.2	413.1	731.6	185.8	422.6	485.5
w/ Pruning	<b>18.4</b>	<b>46.4</b>	<b>54.1</b>	<b>12.1</b>	<b>18.3</b>	<b>29.2</b>

TABLE VII: Precision, recall, and F1 score for multi-session loop closure between the Riverside03 and KAIST sequences from the MulRan dataset recorded with an Ouster OS1-64 LiDAR. We also report the average relative fitness of the loop-closed local maps aligned using the pose estimate  ${}^q T_r$  from our approach as well as the ground truth (GT) aligned local maps.

Reference Session	Riverside03		
	KAIST01	KAIST02	KAIST03
Query Session	2 Months	Same Day	1 Week
Revisit Interval	2 Months	Same Day	1 Week
Precision	1.0	1.0	1.0
Recall	0.019	0.024	0.028
F1	0.037	0.047	0.054
Avg. Rel. Fitness (with ${}^q T_r$ ) $\uparrow$	0.503	0.585	0.665
Avg. Rel. Fitness (with GT) $\uparrow$	0.520	0.599	0.685

closures constraints detected by our pipeline between multiple sessions in another pose-graph optimization step.

We show the aligned trajectories in Fig. 14 along with the aligned local maps from the detected loop closures at the overlapping area. We also provide the precision, recall, and F1 scores of the identified loop closures between local maps in Tab. VII. Our approach detects closures between the low-overlap sequences with 1.0 precision. The recall values are low due to the non-zero overlap criteria used to compute reference closures between local maps. We evaluate the quality of the predicted alignment  ${}^q T_r$  between the detected loop closures by computing the average relative fitness score between the aligned local maps. We also provide the average relative fitness

score between the local maps generated using ground truth poses for comparison. It can be seen from the last two rows of Tab. VII that our pipeline provides a near-ground truth alignment of loop closures without a point cloud registration.

We also perform a similar alignment between the two self-recorded Car and Backpack datasets, which have overlapping areas in their trajectories. These two sequences are also recorded with different LiDAR sensors having different motion profiles since they are mounted on different mobile platforms. The resulting aligned trajectories and the aligned loop closures are shown in Fig. 1.

This experiment supports our last claim that our approach can successfully perform loop closures in challenging scenarios with little overlap. This capability is helpful for tasks such as cross-platform multi-map alignment and fusion for multi-robot collaborations or for long-term change detection.

#### F. Runtime Evaluation of Our Approach

In this additional experiment, we evaluate the runtime of our approach. All the experiments are done on an Intel i9-10980XE @ 3.00 GHz CPU and 64 GB RAM. Our approach is implemented in C++ with a single-thread design. We present the runtime of different components of our pipeline in Tab. VIII. We report the maximum runtimes for a local map to perform ground alignment, loop closure detection, and loop closure alignment. We also provide the total number of local maps, the average number of scans in each local map, and the average number of points in each local map. The runtime for generation of local maps directly corresponds to the LiDAR odometry algorithm we use. We conclude from this analysis that our algorithm is capable of real-time loop closure detection and alignment since it takes approximately a second at worst to process a local map containing millions of points from approximately hundreds of LiDAR scans.

In summary, the experiments presented above support all our claims and showcase the applicability of our approach across various scenarios. The first experiment shows the robustness of our approach to various scanning patterns, FoVs, and

TABLE VIII: Runtime evaluation of our approach.

Sequence	No. of Local Maps	Avg. #Scans / Local Map	Avg. #Points / Local Map	Maximum Execution Time / Local Map [ms]		
				Ground Alignment	Loop Closure Detection	Loop Closure Validation
MulRan Sejong01	231	119	1,405,417	83.5	338.1	0.6
HeLiPR Bridge01	223	96	2,277,130	222.9	832.1	1.0
NCLT 2012-01-08	42	664	3,848,588	225.9	854.5	0.7
Car	190	137	3,190,308	303.4	1080.1	1.0

resolutions of the LiDAR sensor. The second experiment illustrates the applicability of our approach to detect loop closures between multiple sessions with different LiDARs with short-term and long-term revisits through the same environment. The following two experimental evaluations support the newly proposed modules in our loop closure pipeline. We show that our approach can be used on LiDAR platforms with non-planar motion and is also robust towards perceptual aliasing. Finally, we demonstrate that our approach can also align multiple trajectories with minimal physical overlap in the environment.

## VI. CONCLUSION

In this paper, we presented a novel and robust approach to detect loop closures for LiDAR-based SLAM and provide a 3D alignment between the detected closures. Our method relies on the density-preserving BEV projection of local maps generated using local odometry estimates. We treat the local ground plane as a common reference plane across revisits and align the local maps such that the ground plane coincides with the xy-plane of the local map's reference frame. This allows a consistent BEV projection across multiple mobile platforms having different motion profiles of the LiDAR. We detect ORB feature descriptors on the BEV projections and perform a self-similarity pruning of these feature descriptors to avoid false closures due to scene similarities in repetitive environments.

We implemented and extensively evaluated our approach on several public and self-recorded datasets, having various LiDAR sensors recorded from different mobile platforms. We also compare our approach against other state-of-the-art baselines. These experiments demonstrate the accuracy and effectiveness of our approach in detecting loop closures in various urban environments across LiDARs with different scanning patterns and FoVs. We also show the usefulness of our approach in detecting long-term loop closures across multiple sessions recorded with different LiDARs. We release our software as open-source for further use by the community to drive forward the research on place recognition and loop closures in SLAM.

## REFERENCES

- R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- P. Besl and N. McKay. A Method for Registration of 3D Shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 14(2):239–256, 1992.
- M. Bosse, P. Newman, J. Leonard, and S. Teller. Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework. *Intl. Journal of Robotics Research (IJRR)*, 23(12):1113 – 1139, 2004.
- M. Bosse and R. Zlot. Place Recognition Using Keypoint Voting in Large 3D Lidar Datasets. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2013.
- N. Carlevaris-Bianco, A. Ushani, and R. Eustice. University of Michigan North Campus Long-term Vision and LiDAR Dataset. *Intl. Journal of Robotics Research (IJRR)*, 35(9):1023–1035, 2016.
- X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, and C. Stachniss. OverlapNet: Loop Closing for LiDAR-based SLAM. In *Proc. of Robotics: Science and Systems (RSS)*, 2020.
- Y. Chen and G. Medioni. Object Modelling by Registration of Multiple Range Images. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 1991.
- M. Cummins and P. Newman. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *Intl. Journal of Robotics Research (IJRR)*, 27(6):647–665, 2008.
- P. Dellenbach, J. Deschaud, B. Jacquet, and F. Goulette. CT-ICP Real-Time Elastic LiDAR Odometry with Loop Closure. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.
- R. Dubé, A. Cramariuc, D. Dugas, J. Nieto, R. Siegwart, and C. Cadena. SegMap: 3D Segment Mapping using Data-Driven Descriptors. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.
- S. Ferrari, L.D. Giannarino, L. Brizi, and G. Grisetti. MAD-ICP: It Is All About Matching Data-Robust and Informed LiDAR Odometry. *IEEE Robotics and Automation Letters (RA-L)*, 9(11):9175–9182, 2024.
- S. Fontana, G. Agamennoni, R. Siegwart, and D. Sorrenti. Point Clouds Registration with Probabilistic Data Association. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.
- A. Frome, D. Huber, R. Kulkuri, T. Bülow, and J. Malik. Recognizing Objects in Range Data Using Regional Point Descriptors. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2004.
- D. Galvez-Lopez and J.D. Tardos. Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Trans. on Robotics (TRO)*, 28(5):1188–1197, 2012.
- T. Guadagnino, X. Chen, M. Sodano, J. Behley, G. Grisetti, and C. Stachniss. Fast Sparse LiDAR Odometry Using Self-Supervised Feature Selection on Intensity Images. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7597–7604, 2022.
- S. Gupta, T. Guadagnino, B. Mersch, I. Vizzo, and C. Stachniss. Effectively Detecting Loop Closures using Point Cloud Density Maps. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024.
- L. He, X. Wang, and H. Zhang. M2DP: A Novel 3D Point Cloud Descriptor and Its Application in Loop Closure Detection. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.
- B. Jiang and S. Shen. Contour Context Abstract Structural Distribution for 3D LiDAR Loop Detection and Metric Pose Estimation. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.
- A. Johnson and M. Hebert. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 21(5):433–449, 1999.
- M. Jung, W. Yang, D. Lee, H. Gil, G. Kim, and A. Kim. HeLiPR: Heterogeneous LiDAR Dataset for inter-LiDAR Place Recognition under Spatiotemporal Variations. *Intl. Journal of Robotics Research (IJRR)*, 43(12):1867–1883, 2024.
- W. Kabsch. A Solution for the Best Rotation to Relate Two Sets of Vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.
- G. Kim, S. Choi, and A. Kim. Scan Context++: Structural Place Recognition Robust to Rotation and Lateral Variations in Urban Environments. *IEEE Trans. on Robotics (TRO)*, 38(2):21–27, 2021.
- G. Kim, B. Park, and A. Kim. 1-day learning, 1-year localization: Long-term LiDAR Localization Using Scan Context Image. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1948–1955, 2019.
- G. Kim, Y. Park, Y. Cho, J. Jeong, and A. Kim. Mulran: Multimodal Range Dataset for Urban Place Recognition. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

- [25] G. Kim and A. Kim. Scan Context: Egocentric Spatial Descriptor for Place Recognition within 3D Point Cloud Map. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.
- [26] H. Kim, J. Choi, T. Sim, G. Kim, and Y. Cho. Narrowing Your FOV With SOLiD: Spatially Organized and Lightweight Global Descriptor for FOV-Constrained LiDAR Place Recognition. *IEEE Robotics and Automation Letters (RA-L)*, 9(1):9645–9652, 2024.
- [27] J. Komorowski. MinkLoc3D: Point Cloud Based Large-Scale Place Recognition. In *Proc. of the IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2021.
- [28] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A General Framework for Graph Optimization. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2011.
- [29] S. Lee, H. Lim, and H. Myung. Patchwork++: Fast and Robust Ground Segmentation Solving Partial Under-segmentation Using 3D Point Cloud. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.
- [30] Y. Li and H. Li. LiDAR-Based Initial Global Localization Using Two-Dimensional (2D) Submap Projection Image (SPI). In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.
- [31] H. Lim, O. Minho, and H. Myung. Patchwork: Concentric Zone-based Region-wise Ground Segmentation with Ground Likelihood Estimation Using a 3D LiDAR Sensor. *IEEE Robotics and Automation Letters (RA-L)*, 6(4):6458–6465, 2021.
- [32] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Intl. Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.
- [33] S. Lu, X. Xu, H. Yin, Z. Chen, R. Xiong, and Y. Wang. One RING to Rule Them All: Radon Sinogram for Place Recognition, Orientation and Translation Estimation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.
- [34] L. Luo, S. Cao, B. Han, H. Shen, and J. Li. BVMatch: Lidar-Based Place Recognition Using Bird's-Eye View Images. *IEEE Robotics and Automation Letters (RA-L)*, 6(3):6076–6083, 2021.
- [35] L. Luo, S. Cao, Z. Sheng, and H. Shen. LiDAR-Based Global Localization Using Histogram of Orientations of Principal Normals. *IEEE Trans. on Intelligent Vehicles (TIV)*, 7(3):771–782, 2022.
- [36] L. Luo, S. Zheng, Y. Li, Y. Fan, B. Yu, S. Cao, J. Li, and H. Shen. BEVPlace: Learning LiDAR-based Place Recognition using Bird's Eye View Images. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023.
- [37] J. Ma, J. Zhang, J. Xu, R. Ai, W. Gu, and X. Chen. OverlapTransformer: An Efficient and Yaw-Angle-Invariant Transformer Network for LiDAR-Based Place Recognition. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):6958–6965, 2022.
- [38] M. Magnusson, H. Andreasson, A. Nuechter, Achim, and J. Lilienthal. Appearance-Based Loop Detection from 3D Laser Data Using the Normal Distributions Transform. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2009.
- [39] E. Mendes, P. Koch, and S. Lacroix. ICP-based Pose-Graph SLAM. In *Proc. of the IEEE Intl. Symp. on Safety, Security, and Rescue Robotics (SSRR)*, 2016.
- [40] R. Mur-Artal, J. Montiel, and J.D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. on Robotics (TRO)*, 31(5):1147–1163, 2015.
- [41] A. Paigwar, Ö. Erkent, D. Sierra-Gonzalez, and C. Laugier. GndNet: Fast Ground Plane Estimation and Point Cloud Segmentation for Autonomous Vehicles. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.
- [42] C.R. Qi, H. Su, K. Mo, and L.J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [43] C. Qi, K. Yi, H. Su, and L.J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2017.
- [44] T. Röhling, J. Mack, and D. Schulz. A Fast Histogram-Based Similarity Measure for Detecting Loop Closures in 3-D LIDAR Data. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.
- [45] N. Rottmann, R. Bruder, A. Schweikard, and E. Rueckert. Loop Closure Detection in Closed Environments. In *Proc. of the Europ. Conf. on Mobile Robotics (ECMR)*, 2019.
- [46] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2011.
- [47] R. Rusu, N. Blodow, Z. Marton, and M. Beetz. Aligning Point Cloud Views using Persistent Feature Histograms. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 09 2008.
- [48] R. Rusu, N. Blodow, and M. Beetz. Fast Point Feature Histograms (FPFH) for 3D Registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2009.
- [49] S. Salti, F. Tombari, and L. Stefano. SHOT: Unique Signatures of Histograms for Surface and Texture Description. *Journal of Computer Vision and Image Understanding (CVIU)*, 125:251–264, 2014.
- [50] D. Schlegel and G. Grisetti. HBST: A Hamming Distance Embedding Binary Search Tree for Visual Place Recognition. *IEEE Robotics and Automation Letters (RA-L)*, 3:3741–3748, 2018.
- [51] T. Shan, B. Englot, F. Duarte, C. Ratti, and D. Rus. Robust Place Recognition using an Imaging Lidar. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.
- [52] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.
- [53] B. Steder, G. Grisetti, and W. Burgard. Robust Place Recognition for 3D Range Data Based on Point Features. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2010.
- [54] B. Steder, M. Ruhnke, S. Grzonka, and W. Burgard. Place Recognition in 3D Scans Using a Combination of Bag of Words and Point Feature Based Relative Pose Estimation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011.
- [55] F. Tombari, S. Salti, and L.D. Stefano. Unique Shape Context for 3d Data Description. In *Proceedings of the ACM Workshop on 3D Object Retrieval*, 2010.
- [56] S. Umeyama. Least-squares Estimation of Transformation Parameters Between Two Point Patterns. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 13(4):376–380, 1991.
- [57] A. Uy and G. Lee. PointNetVLAD: Deep Point Cloud Based Retrieval for Large-scale Place Recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [58] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):1029–1036, 2023.
- [59] O. Vysotska and C. Stachniss. Lazy Data Association For Image Sequences Matching Under Substantial Appearance Changes. *IEEE Robotics and Automation Letters (RA-L)*, 1(1):213–220, 2016.
- [60] O. Vysotska and C. Stachniss. Effective Visual Place Recognition Using Multi-Sequence Maps. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1730–1736, 2019.
- [61] Y. Wang, Z. Sun, C. Xu, S. Sarma, J. Yang, and H. Kong. LiDAR Iris for Loop-Closure Detection. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.
- [62] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu. RPVNet: A Deep and Efficient Range-Point-Voxel Fusion Network for LiDAR Point Cloud Segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.
- [63] X. Xu, S. Lu, J. Wu, H. Lu, Q. Zhu, Y. Liao, R. Xiong, and Y. Wang. RING++: Roto-Translation Invariant Gram for Global Localization on a Sparse Scan Map. *IEEE Trans. on Robotics (TRO)*, 39(6):4616–4635, 2023.
- [64] X. Xu, H. Yin, Z. Chen, Y. Li, Y. Wang, and R. Xiong. DiSCO: Differentiable Scan Context With Orientation. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):2791–2798, 2021.
- [65] J. Yang, Q. Zhang, Y. Xiao, and Z. Cao. TOLDI: An Effective and Robust Approach for 3D Local Shape Description. *Pattern Recognition*, 65:175–187, 2017.
- [66] H. Yin, X. Xu, S. Lu, X. Chen, R. Xiong, S. Shen, C. Stachniss, and Y. Wang. A Survey on Global LiDAR Localization: Challenges, Advances and Open Problems. *Intl. Journal of Computer Vision (IJCV)*, 132:1–33, 03 2024.
- [67] C. Yuan, J. Lin, Z. Liu, H. Wei, X. Hong, and F. Zhang. BTC: A Binary and Triangle Combined Descriptor for 3-D Place Recognition. *IEEE Trans. on Robotics (TRO)*, 40:1580–1599, 2024.
- [68] C. Yuan, J. Lin, Z. Zou, X. Hong, and F. Zhang. STD: Stable Triangle Descriptor for 3D place recognition. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.
- [69] Y. Zhang, P. Shi, and J. Li. LiDAR-Based Place Recognition For Autonomous Driving: A Survey. *arXiv preprint*, arXiv:2306.10561, 2023.
- [70] B. Zhou, Y. He, K. Qian, X. Ma, and X. Li. S4-slam: A real-time 3d lidar slam system for ground/watersurface multi-scene outdoor applications. *Autonomous Robots*, 45(1):77–98, 2021.
- [71] Q. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv preprint*, arXiv:1801.09847, 2018.