Appendix

A. Parameters setup and pseudo code

Here, we summarize the parameter values in Table A1. While some values were set empirically based on prior work, others were tuned to maintain generalizability without dataset-specific fine-tuning. In addition, we also provide pseudo code in Algorithm 1 to outline the core steps of our approach, making it easier to understand the implementation details and facilitate reproducibility.

B. Detailed experimental setups and datasets

Here, we elaborate on the experimental setups that are briefly outlined in Table A2. For widely used datasets, we followed the existing training and testing protocols. The details of the datasets employed in our generalizability benchmark are described as follows.

- `3DMatch` follows conventional protocol proposed by Zeng *et al.* [87].
- `3DLoMatch` follows conventional protocol proposed by Huang *et al.* [31]. This dataset is derived from `3DMatch` by selectively extracting pairs with low overlap (*i.e.,* 10–30%), allowing for the evaluation of robustness to low-overlap scenarios.
- `ScanNet++F` is from `ScanNet++` [80]. For each sequence, a pre-merged PLY file representing the entire space is provided, along with the poses of the stationary FARO LiDAR sensor used to measure it. That is, the dataset does not provide individual scans. For that reason, using the given scanner poses, we generate per-scanner point clouds by sampling along raycasting paths from each position within the full map, simulating virtual scans. The sampling process respects the scanner's horizontal angular resolution as well as its vertical resolution between consecutive rays [63], ensuring a realistic approximation of the original scans. For each ray, the nearest intersecting point in the merged point cloud is selected to simulate the original scan.
- `ScanNet++i` is also from `ScanNet++` [80]. There are depth images captured using a LiDAR sensor attached to an iPhone 13 Pro. These depth images were converted into point clouds using the toolbox provided by 3DMatch [87]. To generate dense point cloud fragments, 50 consecutive frames were accumulated. Finally, pairs with an overlap ratio of at least 0.4 (*i.e.,* 40% overlap between two fragments) were selected as the final test pairs.
- `TIERS` consists of `Indoor06`, `Indoor08`, `Indoor09`, `Indoor10`, and `Indoor11` sequences in the TIERS dataset [59]. To reduce redundant test pairs, we exclude `Indoor07` because it was acquired in the same room as `Indoor06`.

Param.	Description	Value
κ_{spheric}	Coefficient for voxel size when sphericity is high	0.10
κ_{disc}	Coefficient for voxel size when sphericity is low	0.15
τ_v	Threshold in Eq. (3)	0.05
τ_l	Threshold for the local (l) search radius	0.005
τ_m	Threshold for the middle (m) search radius	0.02
τ_g	Threshold for the global (g) search radius	0.05
δ_v	Sampling ratio for sphericity-based voxelization	10%
N_r	Number of sampling points for radius estimation	2,000
$r_{\max}$	Maximum radius	5.0 m
N_{FPS}	Number of sampled points by FPS	1,500
N_{patch}	Maximum number of points in each patch	512
δ	Truncation threshold for Huber loss	1.0
H	Height of the cylindrical map in Mini-SpinNet	7
W	Sector size of the cylindrical map in Mini-SpinNet	20
D	Feature dimension of the cylindrical map in Mini-SpinNet	32

Table A1: Parameters of each module in our BUFFER-X. Note that with this parameter setup, our approach operates in our generalizability benchmark in an out-of-the-box manner without any human intervention.

Algorithm 1: BUFFER-X pipeline

Input: Source cloud $\mathcal{P}$ and target cloud $\mathcal{Q}$; User-defined parameters τ_v, δ_v, δ_r, $[\tau_l, \tau_m, \tau_g]$, and N_{FPS}
Output: 3D inliers $\mathcal{I}$
$\mathcal{P}_r \leftarrow$ `select_larger_cloud`$(\mathcal{P}, \mathcal{Q})$
$\mathcal{P}_{\text{sampled}} =$ `sample`$(\mathcal{P}_r, \delta_v)$ % Sample δ_v% of cloud points
% Step 1. Geometric bootstrapping
$v =$ `calc_voxel_size`$(\mathcal{P}_{\text{sampled}}, \tau_v)$ % See Eq. (3)
$\mathcal{P} \leftarrow f_v(\mathcal{P}), \mathcal{Q} \leftarrow f_v(\mathcal{Q})$ % Downsample the point clouds
$\mathcal{P}_r \leftarrow$ `select_larger_cloud`$(\mathcal{P}, \mathcal{Q})$
$\mathcal{R} =$ `estimate_radii`$($`sample`$(\mathcal{P}_r, N_r), [\tau_l, \tau_m, \tau_g])$,
    where $\mathcal{R} = [r_l, r_m, r_g]$ % See Eq. (5)
% Step 2. Multi-scale patch embedder
$\mathcal{M}^{\mathcal{P}} = \varnothing, \mathcal{M}^{\mathcal{Q}} = \varnothing$ % Containers of embedding output
for r_ξ in $\mathcal{R}$ **do**
    $\mathcal{P}_\xi =$ `farthest_point_sampling`$(\mathcal{P}, N_{\text{FPS}})$
    $\mathcal{Q}_\xi =$ `farthest_point_sampling`$(\mathcal{Q}, N_{\text{FPS}})$
    $\mathcal{F}^{\mathcal{P}}_\xi, \mathcal{C}^{\mathcal{P}}_\xi =$ `MiniSpinNet`$(\mathcal{P}_\xi, \mathcal{P}, r_\xi)$
    $\mathcal{F}^{\mathcal{Q}}_\xi, \mathcal{C}^{\mathcal{Q}}_\xi =$ `MiniSpinNet`$(\mathcal{Q}_\xi, \mathcal{Q}, r_\xi)$
    $\mathcal{M}^{\mathcal{P}}$`.append`$((\mathcal{P}_\xi, \mathcal{F}^{\mathcal{P}}_\xi, \mathcal{C}^{\mathcal{P}}_\xi))$
    $\mathcal{M}^{\mathcal{Q}}$`.append`$((\mathcal{Q}_\xi, \mathcal{F}^{\mathcal{Q}}_\xi, \mathcal{C}^{\mathcal{Q}}_\xi))$
end
% Step 3. Hierarchical inlier search
$\mathcal{D} = \varnothing, \mathcal{T} = \varnothing$
for i in `range(size`$(\mathcal{M}^{\mathcal{P}}))$ **do**
    $(\mathcal{P}_\xi, \mathcal{F}^{\mathcal{P}}_\xi, \mathcal{C}^{\mathcal{P}}_\xi) = \mathcal{M}^{\mathcal{P}}[i]$
    $(\mathcal{Q}_\xi, \mathcal{F}^{\mathcal{Q}}_\xi, \mathcal{C}^{\mathcal{Q}}_\xi) = \mathcal{M}^{\mathcal{Q}}[i]$
    % Step 3-1. Nearest neighbor-based intra-scale matching
    $\mathcal{A}_\xi =$ `mutual_matching`$(\mathcal{F}^{\mathcal{P}}_\xi, \mathcal{F}^{\mathcal{Q}}_\xi)$
    $(\widehat{\mathcal{P}}_\xi, \widehat{\mathcal{Q}}_\xi, \widehat{\mathcal{C}}^{\mathcal{P}}_\xi, \widehat{\mathcal{C}}^{\mathcal{Q}}_\xi) =$ `filter`$(\mathcal{M}^{\mathcal{P}}[i], \mathcal{M}^{\mathcal{Q}}[i], \mathcal{A}_\xi)$
    % Step 3-2. Pairwise transformation estimation
    $\mathcal{T}_\xi =$ `calc_pairwise_R_and_t`$(\widehat{\mathcal{C}}^{\mathcal{P}}_\xi, \widehat{\mathcal{C}}^{\mathcal{Q}}_\xi)$
    $\mathcal{D}$`.append`$(\widehat{\mathcal{P}}_\xi, \widehat{\mathcal{Q}}_\xi)), \mathcal{T}$`.append`$(\mathcal{T}_\xi)$
end
% Step 3-3. Cross-scale consensus maximization
$\mathcal{I} =$ `consensus_maximization`$(\mathcal{D}, \mathcal{T})$ % See Eq. (9)

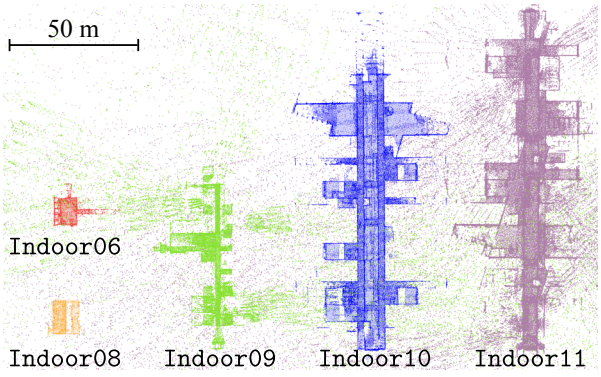
Dataset/Sequence Name	3DMatch, 3DLoMatch	ScanNet++i	ScanNet++F	TIERS
Environment	Room	Room, Interior	Room, Interior	Room, Campus Interior
Acquisition	Handheld	Handheld	Tripod	Cart
Acquisition Site	N/A	N/A	N/A	Univ. Turku, Turku, Finland
Measurement Type	Structured Light	Laser	Laser	Laser
Employed Sensor(s)	Microsoft Kinect, Structure Sensor, Asus Xtion Pro Live, and Intel RealSense	Iphone RGB-D Sensor*	Faro Focus Premium	Velodyne VLP-16, Ouster OS1-64, Ouster0-128
Approx. Range [m]	3.5	3.5	7.0	110
# of Points/Frame	336,274	444,876	1,381,013	34,777
# of Test Pairs	1,623 / 1,781	2,074	2,016	870
Success Criteria	Follow Zeng *et al.* [87]'s criteria (Point-wise RMSE 2 m)	RTE 0.3 m, RRE 15°	RTE 0.3 m, RRE 15°	RTE 2.0 m, RRE 5°

Dataset/Sequence Name	WOD	KITTI	ETH	KAIST	MIT	Oxford
Environment	Urban	Urban	Forest	Campus	Campus	Campus
Acquisition	Vehicle	Vehicle	Tripod	Vehicle	Wheeled	Quadruped / Handheld
Acquisition Site	Phoenix, USA	Karlsruhe, Germany	N/A	KAIST campus, Daejeon, South Korea	MIT campus, Cambridge, USA	Oxford campus, London, UK
Measurement Type	Laser	Laser	Laser	Laser	Laser	Laser
Employed Sensor(s)	Laser Bear Honeycomb*	Velodyne HDL-64E	Rotating Hokuyo UTM-30LX	Livox Avia*, Aeva Aeries II*, Ouster2-128	Velodyne VLP-16	Ouster OS-1 64
Approx. Scale [m]	120	80	85	240	100	100
# Points/Frame	143,960	123,518	96,884	68,790	24,792	55,914
# of Test Pairs	130	555	713	1,991	230	301
Success Criteria	RTE 2.0 m, RRE 5°	RTE 2.0 m, RRE 5°	RTE 0.3 m, RRE 2°	RTE 2.0 m, RRE 5°	RTE 2.0 m, RRE 5°	RTE 2.0 m, RRE 5°

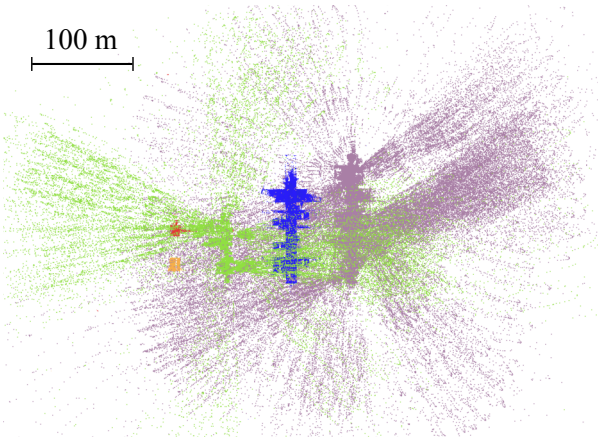
Table A2: Overview of the datasets used in our experiments, including their environments, acquisition sites, measurement types, employed sensors, approximate ranges or scales, and the number of test pairs. The datasets cover a variety of indoor and outdoor environments, spanning different geographic and cultural regions. The superscript ✻ highlights solid-state LiDAR sensors to especially emphasize that our evaluation is not limited to conventional omnidirectional spinning LiDAR sensors, but contains different scanning patterns.

In particular, we used data obtained from the Velodyne VLP-16, Ouster OS1-64, and Ouster OS0-128. While more sensors are available, we observed that point clouds from the Livox Horizon and Livox Avia contain too few points. Specifically, when a human surveyor moves close to a wall in an indoor environment, the narrow field of view of these feed-forwarding LiDAR sensors causes only partial wall surfaces to be captured, unlike omnidirectional LiDAR sensors. Additionally, during rotation, there exist segments where the overlap between consecutive scans becomes completely zero, making them unsuitable for the registration problem.

- WOD follows protocol proposed by Liu *et al.* [43]. This dataset is from the Waymo Open Dataset [68] Perception dataset by extracting LiDAR sequences and corresponding pose files, which are then converted into the KITTI format to ensure compatibility with standard benchmarking pipelines. We set $\tau_{\text{dist}} = 10\,\text{m}$
- KITTI follows conventional protocol proposed by Yew *et al.* [81]. In test scenes, 08, 09, and 10 sequences are employed. Originally, $\tau_{\text{dist}} = 10\,\text{m}$.
- ETH is from the gazebo_summer, gazebo_winter, wood_autmn, and wood_summer sequences of the dataset proposed by Pomerleau *et al.* [55]. The original dataset contains a wider various of scenes; however, following the existing protocol proposed by Ao *et al.* [5], we used four sequences.
- KAIST is from the KAIST05 sequence of the HeLiPR dataset [33]. Originally, HeLiPR contains multiple sequences, but each sequence in the HeLiPR is much longer than those in MIT and Oxford, resulting in many more test pairs compared to other campus scenes (*i.e.,* 1991 vs. 230 or 301 in Table A2). For this reason, we balanced the datasets by using only one sequence. We set $\tau_{\text{dist}} = 10\,\text{m}$.



(a)



(b)

Fig. A1: Visualized map clouds of the TIERS dataset in our experiments. (a) Different scales of the sequences used in our experiments. While each scan is utilized for our evaluation, we build and then visualize map clouds using LiDAR point cloud scans and corresponding poses to illustrate the different scales of the surroundings. (b) A zoomed-out visualization of (a), highlighting the noisy characteristics inherent to indoor LiDAR scanning. Unlike RGB-D sensors, LiDAR sensors emit laser rays and calculate distance by measuring time-of-flight. However, in indoor environments, materials with high reflectivity, such as marble and glass, are highly likely to cause specular and multiple reflections. As a result, these reflections increase the time-of-flight, leading to noisy and incorrect range measurements.

Instead of emphasizing that HeLiPR is a heterogeneous LiDAR sensor dataset, we refer to a subset of it as KAIST in our paper to highlight that our dataset was curated with consideration for geographic and cultural environments. Similarly, we use MIT (from the Kimera-Multi dataset [70]) and Oxford (from the NewerCollege dataset [60]) to emphasize the institutions for the same reason. Further details are provided in Appendix C.

- MIT is from 10_14_acl_jackal sequence of from the Kimera-Multi dataset [70], which is a multi-robot multi-session SLAM dataset. We could have used more scenes, but we chose to use only one sequence to a) match Oxford's frame count as closely as possible (*i.e.,* 230 vs. 301 in Table A2) and b) reduce the redundant test pairs, as multi-robot SLAM datasets often observe the same space multiple times.

  Note that the data was taken from a wheeled robot and acquired using a Velodyne VLP 16 sensor, so we observed that registration almost fails due to point cloud sparsity when $\tau_{\text{dist}} = 10\,\text{m}$ like KITTI, WOD, and KAIST. Therefore, we decided to set $\tau_{\text{dist}} = 5\,\text{m}$.

- Oxford is from the 01, 05, and 07 sequences of the NewerCollege dataset [60]. An interesting aspect is that 01 and 07 were acquired by handheld setup, while the 05 sequence was acquired using a quadruped robot. As shown in Fig. A2, the campus scale was relatively small, so if we use only a single sequence, it only generates few test pairs. For that reason, to ensure at least a similar number of test pairs as MIT, unlike KAIST and MIT, we used three sequences and set $\tau_{\text{dist}} = 5\,\text{m}$.

C. Rationale behind dataset selection

Here, we explain our detailed rationale for why we chose the aforementioned data as our comprehensive dataset.

Variation in environmental scales. First, we want to construct sufficient domain generalization between indoor and outdoor scenes. For existing approaches, only 3DMatch and KITTI were used for indoor-to-outdoor (and vice versa) evaluations. Unfortunately, these experimental setups could not assess how well the methods would perform indoors when using a non-RGB-D camera. Specifically, the maximum range was set to 5 m for 3DMatch and 80–100 m for KITTI.

However, even in indoor environments, LiDAR sensors can measure much farther, as presented in Fig. 2(b) and Fig. A1. Therefore, we aimed to challenge the bias that indoor and outdoor settings can be strictly distinguished by maximum range and decided to include TIERS. That is, in TIERS, although all sequences were collected within the same indoor building, they were captured in diverse environments such as rooms, classrooms, and hallways. As a result, the scale of the surroundings captured by the sensor varies significantly. This was intended to evaluate whether methods could still perform well on indoor scenes beyond the room level, as opposed to the conventional settings in 3DMatch. Moreover, in the TIERS, when data is acquired in a real corridor environment using LiDAR, the point cloud becomes noisy due to the diffuse reflection of laser rays; see Fig. A1(b).

Likewise, one might think that the scales of campus environments are similar; however, as shown in Fig. A2, even

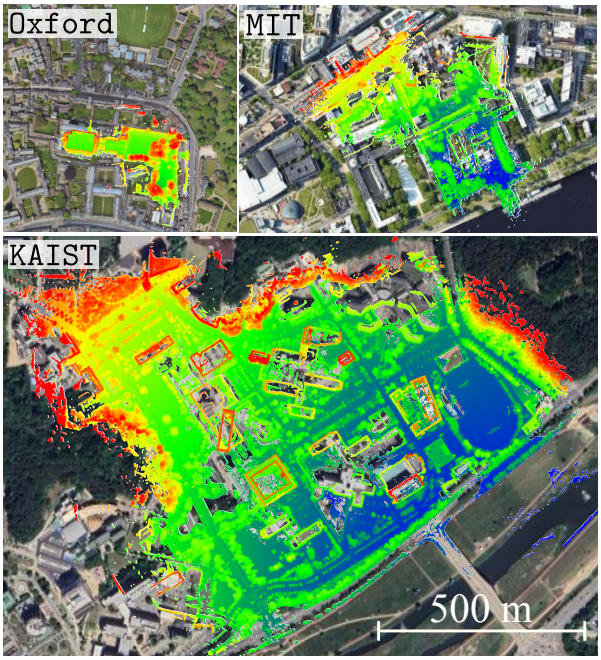


Fig. A2: Scale comparison of three sequences: `Oxford` from the NewerCollege [60], `MIT` from the Kimera-Multi [70], and `KAIST` from the HeLiPR dataset [33] at the same scale (*i.e.,* 500 m). Note that although these sequences fall under the same campus category, their scales differ. For clarity, the map clouds are visualized with respect to their z values.

within the campus category, variations in campus size can lead to differences in the distribution of LiDAR sensor data. Therefore, by incorporating datasets with varying environmental scales, we aim to ensure that our generalizability benchmark includes the full spectrum of scale variations, enabling a more comprehensive assessment of generalization across different settings.

Different scanning patterns with different sensor types. In addition to using the `TIERS` for the reasons mentioned above, we also aimed to evaluate whether the same space remains robust to different scanning patterns. To this end, we employed `KAIST` and `TIERS`. As shown in Fig. A3, even when the same space is captured, variations in the number of laser rays and sensor patterns result in different representations.

Acquisition setups. We also considered that acquisition setups vary in multiple ways. A common bias in previous evaluations in the state-of-the-art approaches is that indoor scanning is performed using handheld devices, whereas outdoor scanning is conducted using vehicles (*i.e.,* the assumption that indoor scanning is performed using a handheld setup, while outdoor scanning is conducted using a vehicle). Thus, we aimed to evaluate whether registration remains robust to different acquisition setups to challenge this assumption.

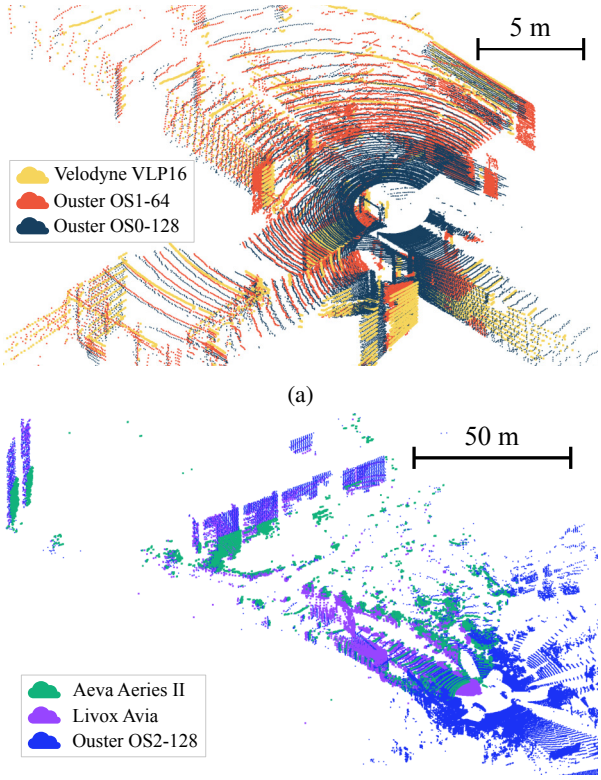


Fig. A3: Examples of visualized LiDAR scans from different LiDAR sensors in (a) the `Indoor10` of `TIERS` dataset [59] and (b) `KAIST05` sequence of the HeLiPR dataset [33]. Note that even in the same environment, differences in the number of LiDAR rays and field of view result in point clouds with different patterns.

For this reason, we included `TIERS`, which was acquired using a sensor cart, and `Oxford`, which was acquired using both a handheld device and a quadruped robot. Additionally, `MIT` was captured using a mobile robot, which performs planar motion similar to a vehicle. However, due to its smaller size, the robot's body experiences significantly more roll and pitch motion, potentially introducing greater motion noise compared to a vehicle.

Diversity of geographic and cultural environments. Lastly, we also aimed to assess whether the acquired data could be generalized across geographic and cultural differences. Some studies [13] have claimed that training on `KITTI` and testing on `KITTI-360` could demonstrate domain generalization. However, since both datasets were collected in Germany using the same vehicle platform, they fall short of demonstrating actual cultural differences.

To address this issue, we leveraged datasets collected from campuses in Asia, Europe, and the USA to evaluate whether such geographic and cultural variations could be accounted for in our proposed benchmark.

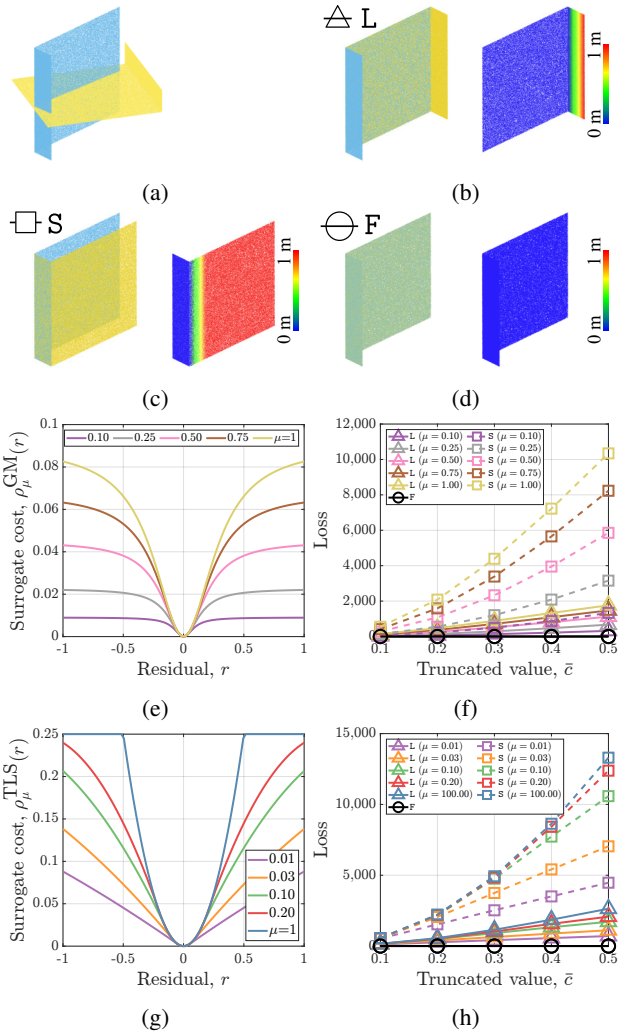


Fig. A4: Illustration of the *global optimum ambiguity* problem in point cloud registration. (a) Example of two misaligned L-shaped point clouds and (b)–(d) possible registration cases. (b) An overlapped case along the longer segment (expressed as `L`), (c) an overlapped case along the shorter segment (expressed as `S`), and (d) a fully overlapped case (expressed as `F`). (e) The behavior of the surrogate cost function of German-McClure (GM) with varying control parameter μ and (f) corresponding loss values for various μ with the user-defined threshold parameter $\bar{c}$. (g) The behavior of the surrogate cost function of truncated least squares (TLS) with μ and (f) corresponding loss values for various μ with the user-defined threshold parameter $\bar{c}$. Note that, regardless of the given μ and $\bar{c}$, the fully overlapped case (*i.e.,* (d)) always results in the lowest loss (since the loss value is zero for any μ and $\bar{c}$, for simplicity, only a single `F` is presented in (f) and (h)).

D. Trade-off between generalization and robustness against partial overlaps

Here, as a supplement to the explanation in Sec. 5.4, we explain with an example why our methodology performed relatively poorly on 3DLoMatch in detail [31]. Suppose we have two arbitrary L-shaped point clouds, as presented in Fig. A4(a). Then, these two L-shaped point clouds can be registered in the following cases:

- Case A: overlapped along the longer segment (Fig. A4(b))
- Case B: overlapped in the short direction (Fig. A4(c))
- Case C: fully overlapped (Fig. A4(d)).

Obviously, the lowest value of the loss function is the fully overlapped case (*i.e.,* Case C); however, in the partial overlap problem, the other local optima (*i.e.,* the partially overlapped case along the longer or shorter segment) can be the actual global optimum. This phenomenon implies that when partial overlap is severe, the relative pose with the smallest loss value is not necessarily the true global optimum.

Therefore, in terms of the optimization problem, we can say that the state with the actual global optimum is not necessarily equal to the state with the global optimum in the cost function. Furthermore, this problem cannot be solved mathematically without prior knowledge of how the two point clouds should be aligned. We refer to this phenomenon as *global optimum ambiguity*.

In addition, even with a learnable non-linear robust kernel, this problem cannot be perfectly resolved. For instance, we examine two renowned non-convex cost functions: a) German-McClure (GM) function and b) truncated least squares (TLS) function, and use them as surrogate cost functions $\rho_\mu(r)$ that adjust their non-linearity by changing the control parameter μ [74]. Formally, by letting the r be the residual and $\bar{c}$ be the user-defined parameter that determines the shape of a kernel, the GM function with μ can be expressed as follows:

$$\rho_\mu^{\mathrm{GM}}(r) = \frac{\mu\bar{c}^2 r^2}{\mu\bar{c}^2 + r^2}, \tag{A12}$$

and the TLS function with μ can be expressed as follows:

$$\rho_\mu^{\mathrm{TLS}}(r) = \begin{cases} r^2 & \text{if } r^2 \in \left[0, \frac{\mu}{\mu+1}\bar{c}^2\right] \\ 2\bar{c}|r|\sqrt{\mu(\mu+1)} - \mu\left(\bar{c}^2 + r^2\right) & \text{if } r^2 \in \left[\frac{\mu}{\mu+1}\bar{c}^2, \frac{\mu+1}{\mu}\bar{c}^2\right] \\ \bar{c}^2 & \text{if } r^2 \in \left[\frac{\mu+1}{\mu}\bar{c}^2, +\infty\right) \end{cases}. \tag{A13}$$

As shown in Fig. A4, even if we vary the non-linearity of the kernel by adjusting the shape of the surrogate cost via μ; see Figs. A4(e) and (g), the fully overlapped case still yields the lowest loss; see Figs. A4(f) and (h). This supports our claim that the global optimum ambiguity problem cannot be easily resolved, no matter how much we formulate it as a non-linear function.

	Env.	Indoor					Outdoor						Average rank
	Dataset	3DMatch	3DLoMatch	ScanNet++i	ScanNet++F	TIERS	KITTI	WOD	KAIST	MIT	ETH	Oxford	
Conventional	FPFH [62] + FGR [93] + 👁	62.53	15.42	77.68	92.31	80.60	98.74	**100.00**	89.80	74.78	91.87	99.00	8.55
	FPFH [62] + Quatro [42] + 👁	8.22	1.74	9.88	97.27	86.57	99.10	**100.00**	91.46	79.57	51.05	91.03	10.82
	FPFH [62] + TEASER++ [76] + 👁	52.00	13.25	66.15	97.22	73.13	98.92	**100.00**	89.20	71.30	93.69	99.34	8.82
Deep	FCGF [21]	8.04	0.17	19.96	23.07	77.82	98.92	95.38	88.34	82.17	6.59	75.08	14.82
	+ 👁	34.97	4.12	31.00	25.10	77.93	98.92	96.92	94.22	89.13	39.97	86.05	12.00
	+ 👁 + 📐	34.97	4.12	33.37	25.10	77.93	98.92	99.23	94.22	90.43	39.97	90.03	11.27
	Predator [31]	N/A (Err)	N/A (Err)	N/A (Err)	N/A (Err)	69.43	99.82	**100.00**	N/A (OOM)	54.78	55.68	89.04	14.09
	+ 👁	16.47	0.00	9.40	3.72	69.77	99.82	**100.00**	71.3	76.52	56.67	89.04	12.00
	+ 👁 + 📐	23.2	3.31	9.40	3.72	69.77	99.82	**100.00**	94.02	86.08	71.95	95.02	9.91
	GeoTransformer [58]	N/A (Err)	N/A (Err)	N/A (Err)	N/A (Err)	N/A (Err)	99.82	**100.00**	63.84	93.91	77.00	73.42	12.73
	+ 👁	5.94	0.30	15.91	34.18	20.57	99.82	**100.00**	63.84	93.91	77.56	73.42	10.90
	+ 👁 + 📐	62.17	14.38	76.52	90.63	87.36	99.82	**100.00**	96.84	96.52	81.77	98.01	5.55
	BUFFER [5]	N/A (Err)	N/A (Err)	17.60	88.84	93.34	99.64	**100.00**	99.50	95.22	98.18	99.34	8.27
	+ 👁	91.19	64.51	93.15	97.81	93.57	99.64	**100.00**	**99.55**	97.39	99.86	99.34	3.27
	+ 👁 + 📐	91.19	64.51	93.15	97.81	93.57	99.64	**100.00**	**99.55**	97.39	99.86	99.34	3.27
	PARENet [79]	N/A (Err)	N/A (Err)	N/A (Err)	N/A (Err)	N/A (Err)	99.82	97.69	57.51	75.22	68.30	66.11	15.82
	+ 👁	0.77	0.10	3.04	12.00	19.20	99.82	98.46	57.51	75.22	68.44	66.11	15.00
	+ 👁 + 📐	22.09	4.98	29.99	42.91	52.99	99.82	**100.00**	89.50	87.39	72.65	94.02	9.18
	Ours with only r_m	91.96	63.59	92.38	99.45	94.37	99.82	**100.00**	**99.55**	**99.13**	**100.00**	**99.67**	1.82
	Ours	**93.79**	**65.89**	**95.13**	**99.65**	**94.83**	99.82	**100.00**	**99.55**	**99.13**	**100.00**	**99.67**	**1.00**

Table A3: Quantitative comparison of generalization performance in terms of success rate (unit: %). Deep learning-based models were trained only on KITTI [27] and RANSAC was used with a maximum iteration of 50K. Icons represent oracle tuning (👁) for voxel size and radius, and scale alignment (📐) to match dataset scales (e.g., adjusting 3DMatch's scale to KITTI by multiplying $\frac{0.3}{0.025}$). N/A (Err) indicates failure due to an insufficient number of points remaining after voxelization with the voxel size typically used for outdoor settings, making keypoint extraction or descriptor generation infeasible. N/A (OOM) indicates an out-of-memory error caused by excessive memory usage.

E. Quantitative results using KITTI

One may wonder how the model performs when trained on KITTI, so we also present the results of training on KITTI in Table A3. Interestingly, the success rates in 3DMatch and 3DLoMatch become slightly lower, while those in TIERS, KAIST, MIT, ETH improve when our BUFFER-X was trained on KITTI instead of 3DMatch. We speculate that because the model was trained on LiDAR data, it is better optimized for the distribution of LiDAR point clouds. Additionally, training on sparse point clouds results in exposure to relatively less diverse patch patterns, which may explain the performance drop when testing on 3DMatch.

Specifically, because the cloud points in 3DMatch are much denser than those in KITTI, randomly sampling N_{patch} points from these clouds for each patch results in more varied local coordinate patterns. Thus, training in 3DMatch enables the model to learn more diverse local neighborhood patterns during training by randomly adjusting the size of N_{patch}. In contrast, in the case of KITTI, the points are relatively sparse because they are acquired using a LiDAR sensor. As a result, even when randomly sampling N_{patch} points within the local neighborhoods, the diversity of local patterns remains relatively limited.

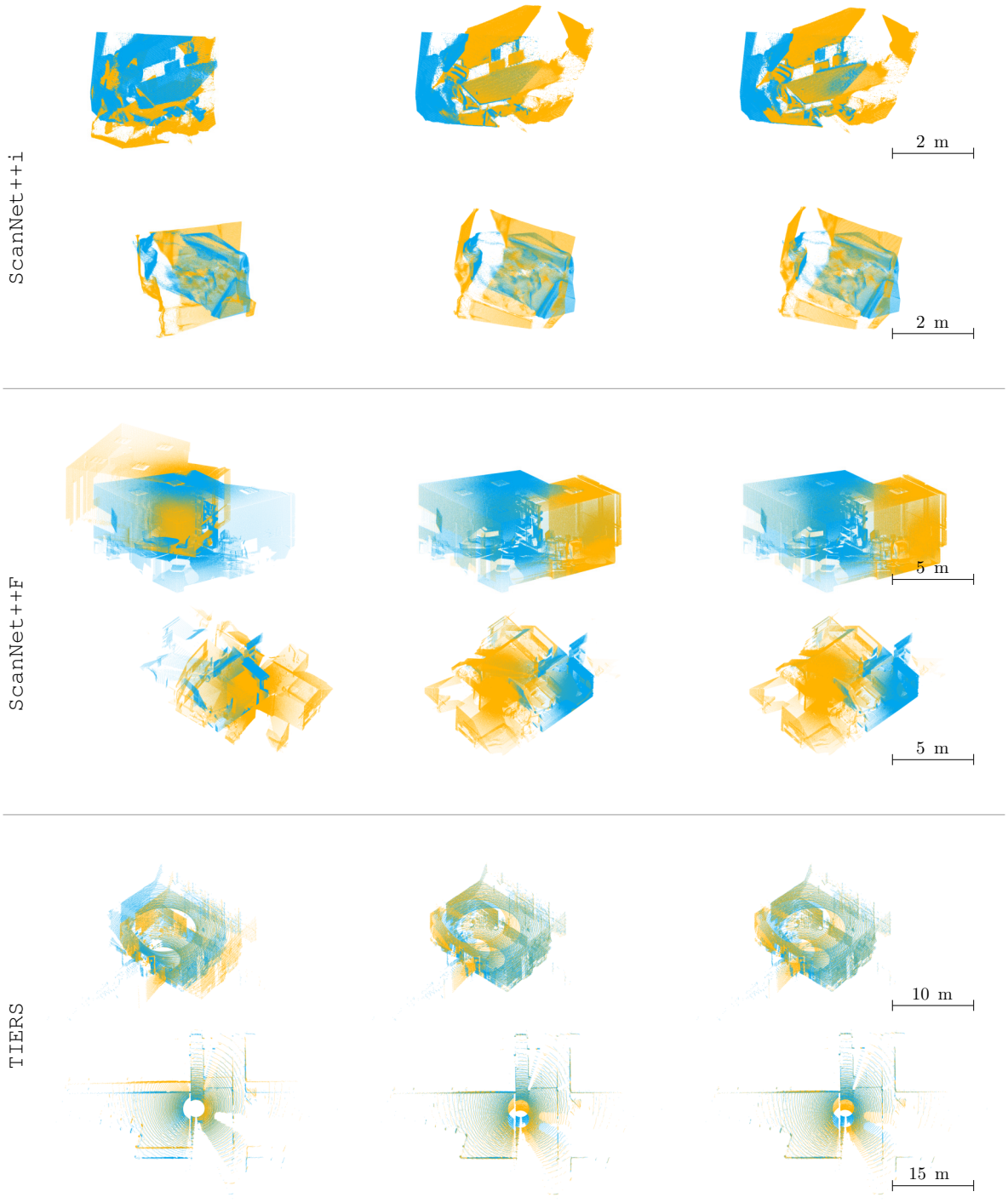
Moreover, our method demonstrated a higher rank compared to the state-of-the-art approaches. In particular, we observed that applying the voxel size used for outdoor training to indoor environments resulted in too few points remaining, leading to unexpected errors (referred to as "N/A (Err)" in Table A3). For example, when downsampling a $3 \times 3 \times 6\,\text{m}^3$ space with the 0.3 m voxel size, which is a typical size used for outdoor settings, only 2,000 points remain. In addition, we found that networks requiring large memory, such as Predator, encountered out-of-memory errors when processing denser point clouds (referred to as "N/A (OOM)" in Table A3). This means that an out-of-memory issue occurs when handling a 128-channel LiDAR point cloud without manual tuning of the parameters used for a 64-channel LiDAR point cloud.

Therefore, while training on KITTI led to some variations in overall performance, our approach remains more robust than other state-of-the-art methods.

F. Qualitative results in diverse scenes

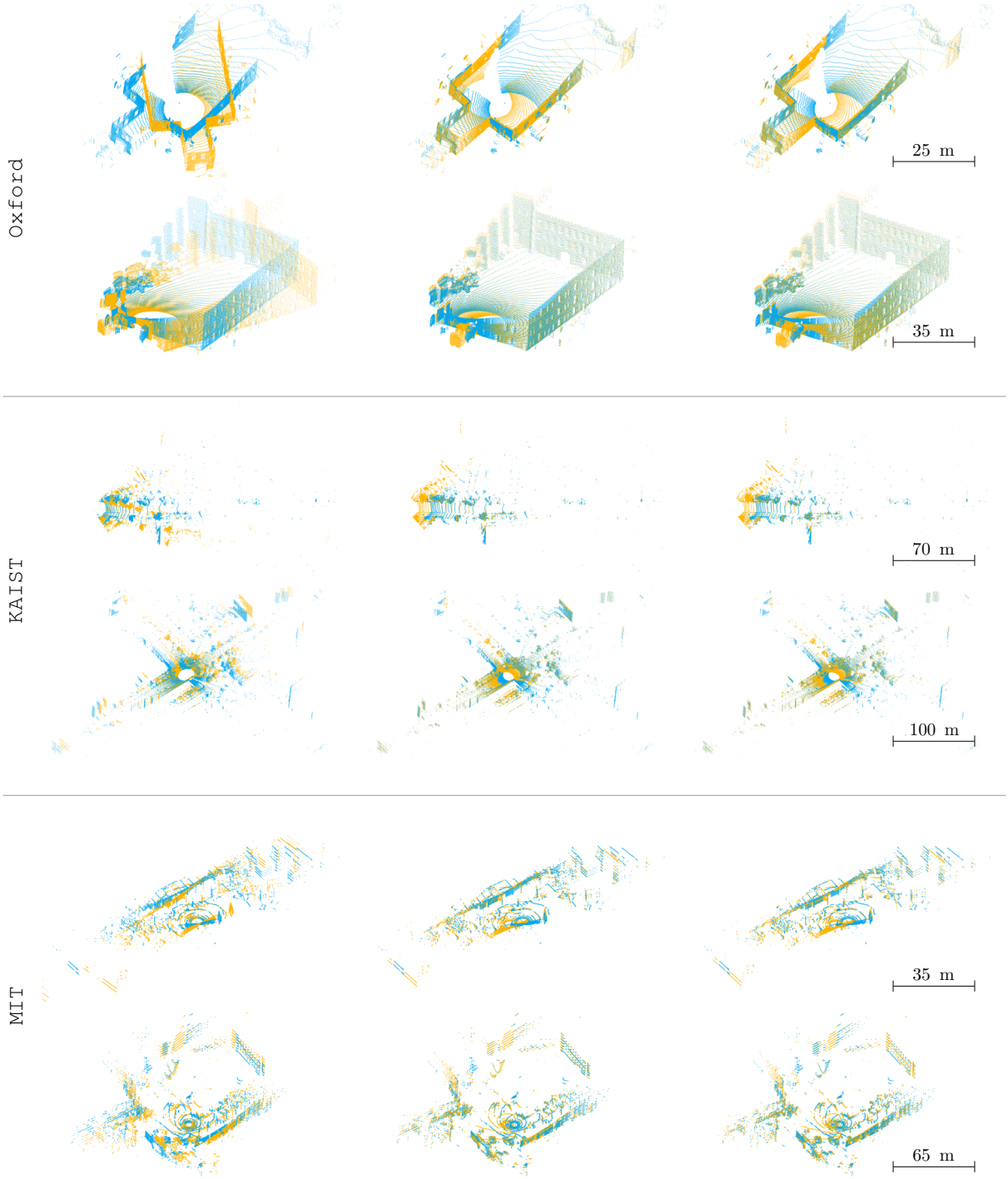
We present the qualitative results in Fig. A5 and Fig. A6. Remarkably, even when trained on 3DMatch, which consists solely of RGB-D point clouds with an approximate maximum range of 3.5 m, the model performs robustly on sparse LiDAR point clouds in both indoor and outdoor environments. In particular, our approach successfully performed registration regardless of the sensor type or acquisition setup, whether the sparse 3D point clouds were acquired by solid-state LiDAR sensors (the first row among the KAIST rows in Fig. A6) or captured from a robot platform (MIT rows in Fig. A6).

Therefore, these results further support the zero-shot registration capability of our BUFFER-X, regardless of the environment, sensor type, acquisition setup, or range.



(a) Source and target (b) BUFFER-X (Ours) (c) Ground truth

Fig. A5: Qualitative results on indoor point cloud registration (T-B): `ScanNet++i`, `ScanNet++F`, and `TIERS` sequences. (a) Input source (yellow) and target (cyan) point clouds before registration. (b) Registration results obtained using our BUFFER-X, trained only on `3DMatch`. (c) Ground truth alignment. Visualization demonstrates that BUFFER-X achieves accurate alignment, closely matching the ground truth.



(a) Source and target (b) BUFFER-X (Ours) (c) Ground truth

Fig. A6: Qualitative results on outdoor point cloud registration: (T-B): `Oxford`, `KAIST`, and `MIT` sequences. (a) Input source (yellow) and target (blue) point clouds before registration. (b) Registration results obtained using our BUFFER-X, trained only on `3DMatch`. (c) Ground truth alignment. Visualization demonstrates that BUFFER-X achieves accurate alignment, closely matching the ground truth.