LightLoc: Learning Outdoor LiDAR Localization at Light Speed -Supplementary Material-

Wen Li[1,2*] Chen Liu[1,2*] Shangshu Yu[3†] Dunqiang Liu[1,2] Yin Zhou[4]
Siqi Shen[1,2] Chenglu Wen[1,2] Cheng Wang[1,2†]
[1]Fujian Key Laboratory of Sensing and Computing for Smart Cities, Xiamen University
[2]Key Laboratory of Multimedia Trusted Perception and Efficient Computing,
Ministry of Education of China, Xiamen University
[3]Nanyang Technological University [4]GAC R&D Center

In this supplementary, we first describe the architecture and training process of the scene-agnostic feature backbone (Sec. 1). Then, we describe the architecture and training process of the scene-specific head (Sec. 2). We further provide additional results (Sec. 3). Finally, we show more visualizations on the QEOxford [1, 8], Oxford [1], and NCLT [11] datasets (Sec. 4).

1. Scene-agnostic Feature Backbone

1.1. Backbone Architecture

Inspired by the DSAC* [2], we adopt the backbone of SGLoc [8], modifying it by reducing the feature dimension and the number of residual layers. As a result, the parameter count decreases significantly from 55M to 16M. The architecture of our backbone is illustrated in Fig. 1. The backbone processes a point cloud as input, progressively reducing the spatial resolution to $\frac{1}{8}$ while increasing the channel dimension to 512.

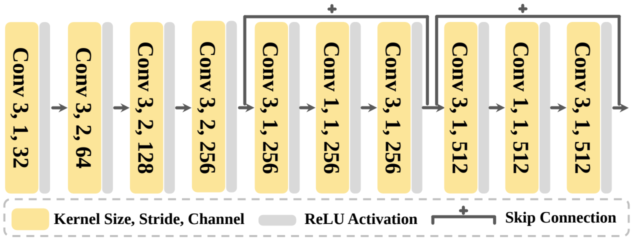
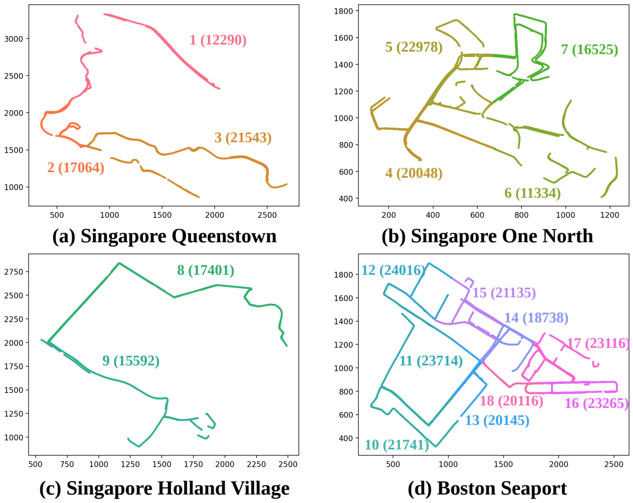


Figure 1. **Architecture of the scene-agnostic feature backbone**. The parameter count is about 16M.



(a) Singapore Queenstown

(b) Singapore One North

(c) Singapore Holland Village

(d) Boston Seaport

Figure 2. **Illustration of the multi-scene division in nuScenes.** The scene indices and the number of samples are reported.

1.2. Backbone Training

We use the nuScenes dataset [4], including both Trainval and Test splits, totaling 350K samples, to train the scene-agnostic feature backbone. The nuScenes dataset is collected using sensors mounted on two autonomous-capable Renault Zoe cars, equipped with identical sensor layouts, operating in Boston and Singapore—two cities known for their dense traffic and challenging driving conditions. The point cloud is captured by a Velodyne HDL-32E LiDAR with a frequency of 20Hz. The ground truth pose is obtained using an Advanced Navigation Spatial GPS&IMU system, offering a position accuracy of 20mm, a heading accuracy of 0.2° with GNSS, and roll&pitch accuracy of 0.1°.

We first apply the K-Means clustering and manually modify the results to divide the dataset into multi-scenes. The results are shown in Fig. 2, where we report the scene indices and the number of samples in each scene. We train our backbone on 18 scenes in parallel, attaching 18 regression heads to it. Each regression head is a multi-layer per-

[*] Equal contribution.
[†] Corresponding author.

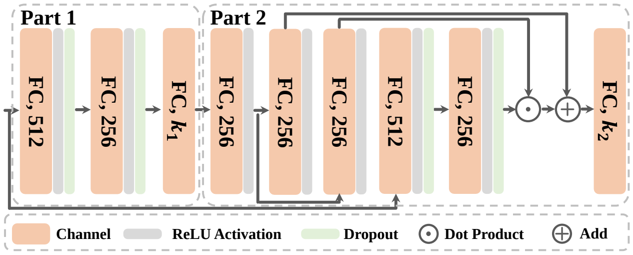


Figure 3. **Architecture of the sample classification head.** Part 1 is used for guiding SCR while Part 2 is used in extended applications requiring finer classification. This figure includes both parts to demonstrate the adaptability to different use cases.

ception with 6 layers and a width of 512. There is a skip connection after the first 3 layers of each head. We train the backbone with half-precision floating point weights. We perform random translation and rotation data augmentation on the input point clouds. Specifically, there is a 50% chance of translating along the x and y axes by -1 to 1 meters, rotating around the roll and pitch axes by -5° to 5°, and rotating around the yaw axis by -10° to 10°. It is important to note that, regardless of how point clouds are transformed, the learned ground truth remain consistent.

We train the backbone using a batch size of 32 samples per regression head. To prevent memory overflow, we process the forward and backward passes for 3 regression heads at a time. Gradients are accumulated across all 18 regression heads before performing a single parameter update.

2. Scene-specific Head

2.1. Head Architecture

In a new scene, we need to train a sample classification network and a SCR. Therefore, we have a classification head and a regression head. We now introduce each of these two heads in detail.

As we described in the main paper, inspired by recent work [5, 6, 10], we implement the hierarchical classification network as the base and hyper network. Fig. 3 illustrates the complete sample classification head, designed to support both SCR and extended applications. The input is the feature after global max pooling. As shown in Part 1, at the first level, the feature map from global max pooling is fed to an MLP to output the classification probability features F_{k_1} with k_1 categories. Starting from the second level, as shown in Part 2, the feature pattern is modulated by a hyper network, according to the classification probability from the previous level. The intuition of the modulation [12] is that similar feature patterns appearing in different regions should be classified under different labels. Then, an MLP is used to output the second level classification probability feature F_{k_2} with k_2 categories.

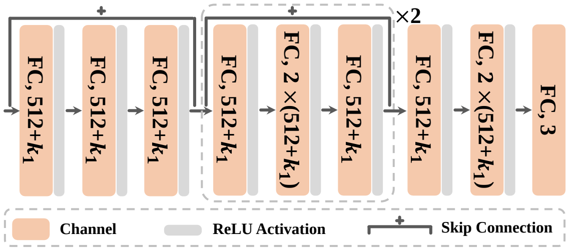


Figure 4. **Architecture of the regression head.** k_1 is the number of clusters.

In the sample classification guidance, only Part 1 is used to help SCR learning. Part 2 is employed only when addressing accumulated errors in SLAM, where finer classification results are required.

Fig. 4 is the regression head, corresponding to the MLP in the SCR of Fig. 2 in the main paper. The input is the concatenated features: scene distribution features perturbed by Gaussian noise (standard deviation of 0.1) and dense descriptors obtained from the scene-agnostic backbone. The features are transformed by a residual block, which is followed by 2 sequential residual blocks. Finally, three FC layers are applied to get the corresponding point cloud in the world coordinates.

2.2. Head Training

In this paper, we train the scene-specific heads on QEOxford [1, 8], Oxford [1] and NCLT [11] datasets, respectively.

The **Oxford** dataset is collected in January 2019 along a central Oxford route, capturing a variety of weather (sunny, overcast) and lighting conditions (dim, glare) that make localization challenging. Following [8, 14], we use data from 11-14-02-26, 14-12-05-52, 14-14-48-55, and 18-15-20-12 for training, and data from 15-13-06-37, 17-13-26-39, 17-14-03-00, and 18-14-14-42 for testing. The same trajectories are also selected from the **QEOxford** dataset, where GPS&IMU errors are corrected using the PQEE [8].

The **NCLT** dataset is collected approximately biweekly from January 8, 2012 to April 5, 2013, on the University of Michigan's North Campus. It includes a variety of environmental changes, such as seasonal variations, lighting conditions, and changes in building structures. The dataset also covers both indoor and outdoor scenes, which adds to the complexity. For our experiments, we use the data from 2012-01-22, 2012-02-02, 2012-02-18, and 2012-05-11 as the training set, and the data from 2012-02-12, 2012-02-19, 2012-03-31, and 2012-05-26 as the test set.

For classification head training, as mentioned in the main paper, we accelerate the process by creating a buffer on the GPU to store global features and their associated classification labels. The buffer is filled by cycling repeatedly through the shuffled training sequence. Each point cloud is

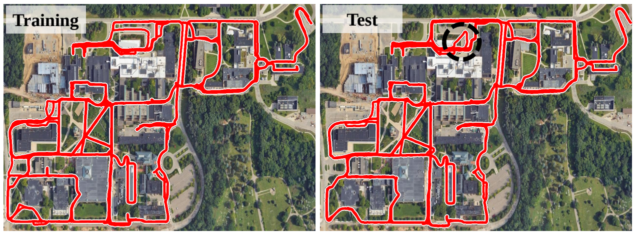


Figure 5. **Illustration of training and test trajectories of NCLT.** We use a black dashed circle to highlight the unknown region in the test trajectory.

augmented using a similar approach to the backbone training. The classification head is then trained by iterating over the shuffled buffer. Specifically, within a 5-minute training period (including the time spent filling the buffer), we complete 50 epochs.

For regression head training, we first apply the same data augmentation used in the backbone training to each frame of the input point cloud. Then, the point cloud is passed through the feature backbone to obtain dense descriptors. The features from the classification head are normalized to the unit sphere, Gaussian noise is added, and the features are normalized back to the unit sphere. Finally, the two feature sets are concatenated, and 256 voxels are randomly selected to learn their corresponding global coordinates through regression. Throughout this process, we also apply the proposed redundant sample downsampling technique to enhance efficiency.

2.3. Implementation Details

For the Oxford and QEOxford datasets, we set the voxel size to 0.25. During classifier training, the number of clusters k_1 and k_2 are configured to 25 and 100, respectively. A 150MB buffer is constructed on the GPU to store features, enabling rapid training over 50 epochs within 5 minutes, following the ACE [3]. In the regressor training stage, the downsampling ratio r_d in RSD, along with the start epoch r_{st}, stop epoch r_{sp}, and total training epochs E, are set to 0.25, 0.25, 0.85, and 25, respectively.

For the NCLT dataset, the voxel size, buffer size, and k_1 are set to 0.3, 120MB, and 100, respectively. In the regressor training phase, the r_d and E are configured to 0.15 and 30, respectively.

3. Additional Results

3.1. Results of 2012-05-26 on NCLT

As described in Tab. 3 of the main paper, we discard areas with localization failure, as regression-based methods cannot generalize to unknown regions. Details follow.

As shown in Fig. 5, we present the training and test trajectories of the NCLT dataset, highlighting the unknown region in 2012-05-26. Previous work [13] demonstrates that regression-based methods are not guaranteed to generalize from the training data in practical scenarios. We also report the results with and without the area, as shown in Tab. 1. It is clear that when the area is excluded, the errors of the different methods are significantly reduced. For a fair comparison of the methods, we report the results after excluding these regions, where the methods fail to provide useful information, and the results are essentially unreliable.

Methods	HypLiLoc	DiffLoc	SGLoc	LiSA	Ours
Area Included	2.90/3.47	1.88/2.43	3.81/4.74	3.30/2.84	3.81/3.48
Area Excluded	2.29/3.34	1.36/2.48	3.48/4.43	3.11/2.72	3.10/3.26

Table 1. **Results on the 2012-05-26.** We report the mean error [m/°] with unknown area included and excluded.

Methods	15-13-06-37	17-13-26-39	17-14-03-00	18-14-14-42	Average
nuScenes	0.82/1.12	0.85/1.07	0.81/1.11	0.82/1.16	0.83/1.12
KITTI	1.56/1.79	1.60/1.73	1.16/1.69	1.69/1.70	1.50/1.73

Table 2. **Results on QEOxford dataset.** We report the mean error [m/°] for scene-agnostic feature backbone training on the nuScenes and KITTI datasets.

Methods	15-13-06-37	17-13-26-39	17-14-03-00	18-14-14-42	Average
nuScenes	2.33/1.21	3.19/1.34	3.11/1.24	2.05/1.20	2.67/1.25
KITTI	3.49/2.71	4.15/2.79	3.70/2.59	3.40/2.43	3.69/2.63

Table 3. **Results on Oxford dataset.** We report the mean error [m/°] for scene-agnostic feature backbone training on the nuScenes and KITTI datasets.

Methods	2012-02-12	2012-02-19	2012-03-31	2012-05-26†	Average
nuScenes	0.98/2.76	0.89/2.51	0.86/2.67	3.10/3.26	1.46/2.80
KITTI	1.51/4.15	1.61/3.70	1.46/3.99	5.77/5.30	2.59/4.29

Table 4. **Results on NCLT dataset.** We report the mean error [m/°] for scene-agnostic feature backbone training on the nuScenes and KITTI datasets.. † indicates that we discard areas with localization failure, as regression-based methods cannot generalize to unknown regions.

3.2. Results of Training Backbone on KITTI

In this section, we train the scene-agnostic feature backbone using the KITTI dataset [7, 15]. Since KITTI provides ground truth poses only for the training set (trajectories 00-10, totaling 23K samples), we use these 11 trajectories to train the backbone. The KITTI dataset, collected in Karlsruhe, Germany, utilizes the autonomous driving platform Annieway. It captures diverse real-world driving scenarios, including urban, rural, and highway environments. Point clouds are recorded using a Velodyne HDL-64E LiDAR operating at 10Hz, while ground truth poses are derived from a GPS&IMU system.

Similar to the backbone training with the nuScenes dataset, as described in Sec. 1, we train the backbone across 11 scenes in parallel, attaching 11 regression heads to it.

Then, we follow Sec. 2 to train the scene-specific prediction heads on the QEOxford, Oxford, and NCLT datasets.

Tab. 2, Tab. 3, and Tab. 4 present the comparison results of training the backbone on the nuScenes dataset, evaluated across three different datasets. The results clearly show a decreasing trend in performance. Specifically, on the QEOxford dataset, position and orientation accuracy decrease by 80.7% and 54.5%, respectively. In the Oxford and NCLT datasets, the corresponding decreases are 38.2%/110.4% and 77.7%/53.2%, respectively. We conclude that this is primarily due to insufficient data and differences in LiDAR types. This motivates us to incorporate data from more scenes, platforms, and LiDAR types to jointly train the backbone in future work, further enhancing its generalization capabilities.

4. Visualization

We show more visualization results of the top 4 methods in the main paper (DiffLoc [9], SGLoc [8], LiSA [14], and the proposed LightLoc) in Fig. 6, Fig. 7, and Fig. 8 on the QEOxford, Oxford, and NCLT datasets, respectively.

Clearly, compared to the existing state-of-the-art method, LiSA, our predicted trajectories yield comparable results in terms of accuracy.

It is important to emphasize that our work primarily focuses on minimizing training time and reducing parameter storage requirements. While our method may not always outperform others on all test trajectories, it consistently achieves results within one hour of training on large-scale datasets, which is significantly faster than current state-of-the-art methods. LightLoc achieves an effective balance between training time and performance, making it a practical solution for time-sensitive applications such as autonomous driving, drones, and robotics.

References

[1] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman, and Ingmar Posner. The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In *ICRA*, pages 6433–6438, 2020. 1, 2, 5, 6

[2] Eric Brachmann and Carsten Rother. Visual camera relocalization from rgb and rgb-d images using dsac. *IEEE TPAMI*, 44(9):5847–5865, 2021. 1

[3] Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *CVPR*, pages 5044–5053, 2023. 3

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631, 2020. 1

[5] Siyan Dong, Qingnan Fan, He Wang, Ji Shi, Li Yi, Thomas Funkhouser, Baoquan Chen, and Leonidas J Guibas. Robust neural routing through space partitions for camera relocalization in dynamic indoor environments. In *CVPR*, pages 8544–8554, 2021. 2

[6] Siyan Dong, Shuzhe Wang, Yixin Zhuang, Juho Kannala, Marc Pollefeys, and Baoquan Chen. Visual localization via few-shot scene region classification. In *3DV*, pages 393–402, 2022. 2

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361, 2012. 3

[8] Wen Li, Shangshu Yu, Cheng Wang, Guosheng Hu, Siqi Shen, and Chenglu Wen. Sgloc: Scene geometry encoding for outdoor lidar localization. In *CVPR*, pages 9286–9295, 2023. 1, 2, 4, 5

[9] Wen Li, Yuyang Yang, Shangshu Yu, Guosheng Hu, Chenglu Wen, Ming Cheng, and Cheng Wang. Diffloc: Diffusion model for outdoor lidar localization. In *CVPR*, pages 15045–15054, 2024. 4

[10] Xiaotian Li, Shuzhe Wang, Yi Zhao, Jakob Verbeek, and Juho Kannala. Hierarchical scene coordinate classification and regression for visual localization. In *CVPR*, pages 11983–11992, 2020. 2

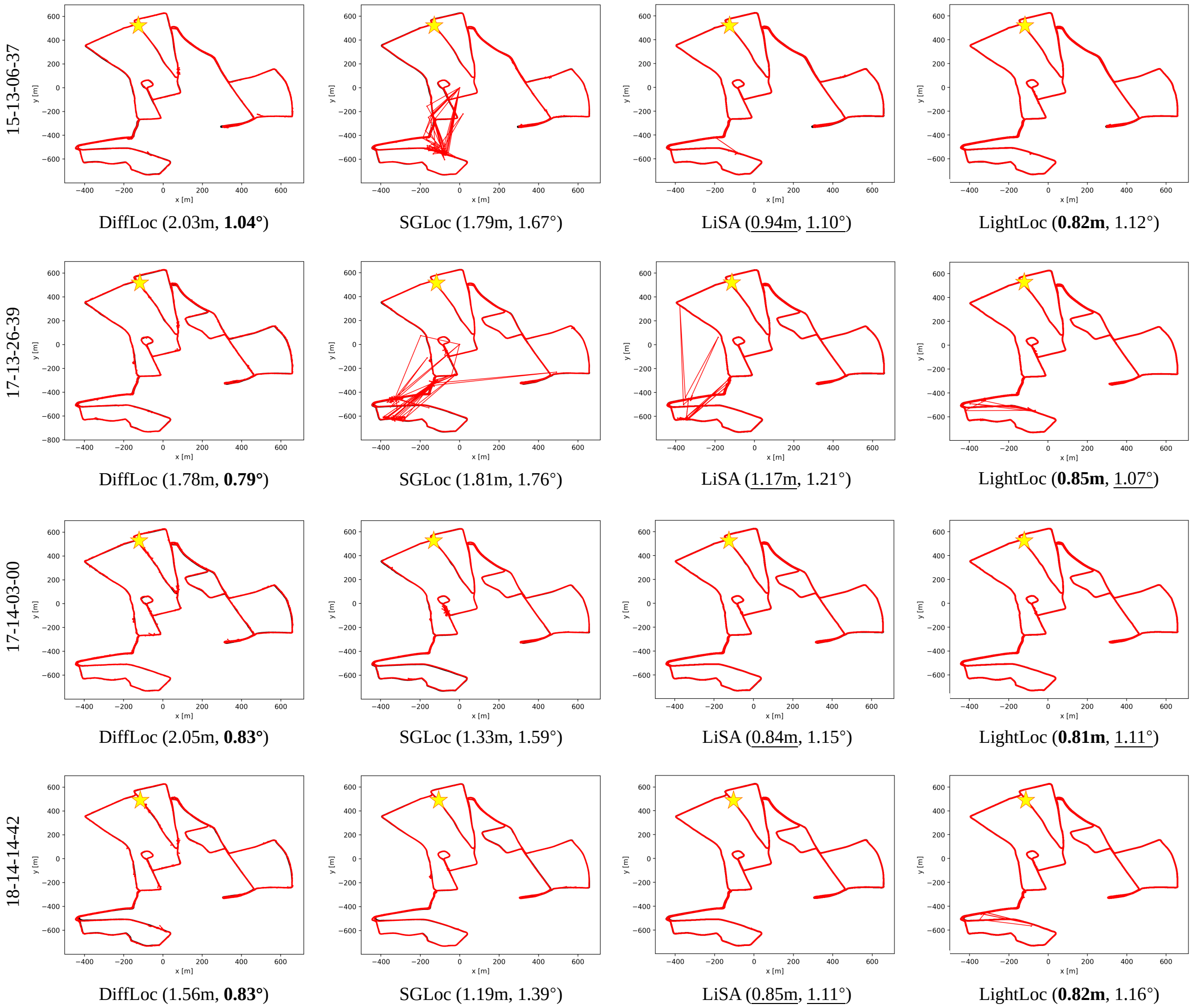
[11] Carlevaris-Bianco Nicholas, K. Ushani Arash, and M. Eustice Ryan. University of michigan north campus long-term vision and lidar dataset. *IJRR*, 35:545–565, 2015. 1, 2, 7

[12] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 2

[13] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regressionr. In *CVPR*, pages 3302–3312, 2019. 3

[14] Bochun Yang, Zijun Li, Wen Li, Zhipeng Cai, Chenglu Wen, Yu Zang, Matthias Muller, and Cheng Wang. Lisa: Lidar localization with semantic awareness. In *CVPR*, pages 15271–15280, 2024. 2, 4

[15] Shangshu Yu, Cheng Wang, Chenglu Wen, Ming Cheng, Minghao Liu, Zhihong Zhang, and Xin Li. Lidar-based localization using universal encoding and memory-aware regression. *PR*, 128:108915, 2022. 3



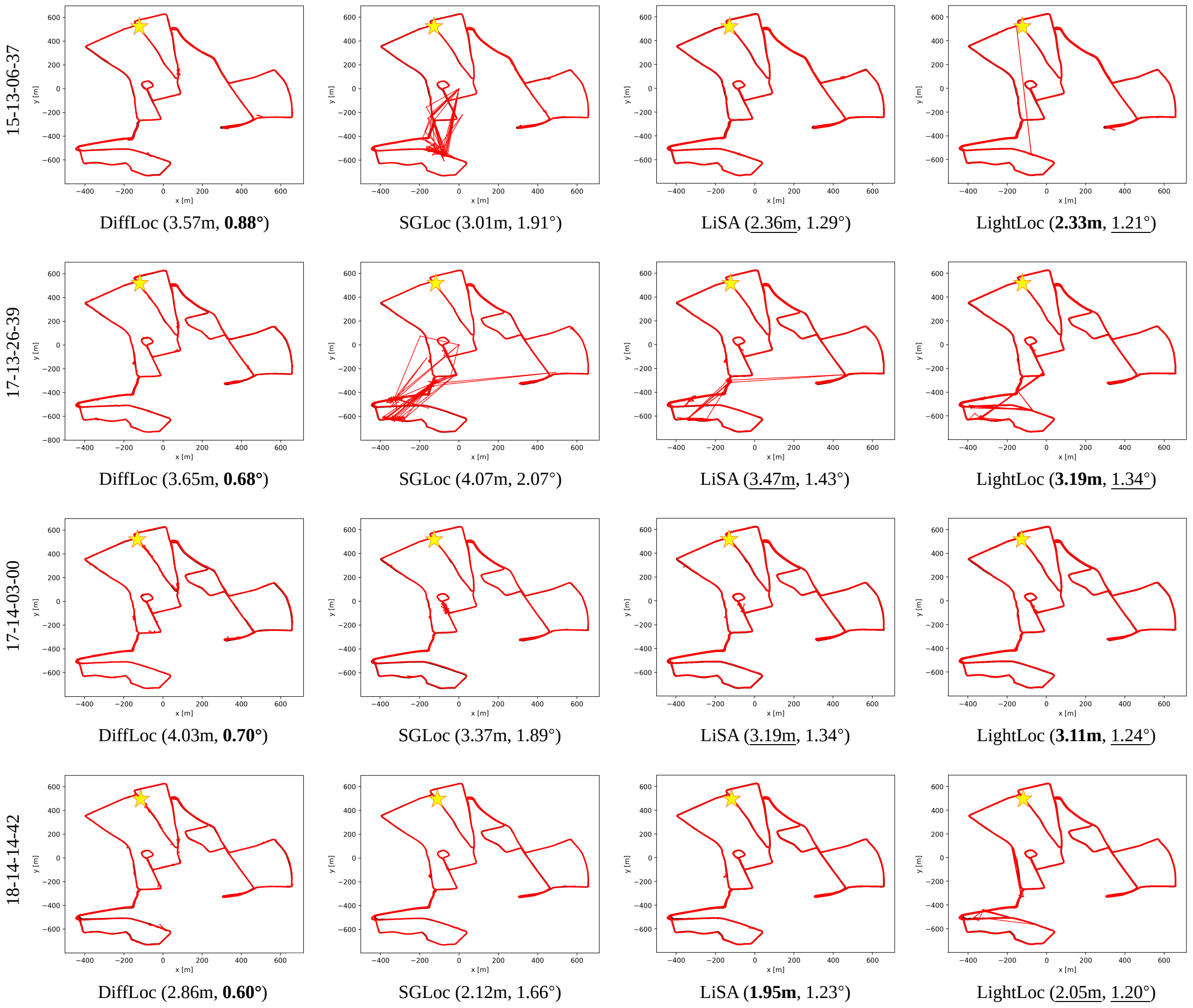
DiffLoc (2.03m, **1.04°**) SGLoc (1.79m, 1.67°) LiSA (0.94m, 1.10°) LightLoc (**0.82m**, 1.12°)

DiffLoc (1.78m, **0.79°**) SGLoc (1.81m, 1.76°) LiSA (1.17m, 1.21°) LightLoc (**0.85m**, 1.07°)

DiffLoc (2.05m, **0.83°**) SGLoc (1.33m, 1.59°) LiSA (0.84m, 1.15°) LightLoc (**0.81m**, 1.11°)

DiffLoc (1.56m, **0.83°**) SGLoc (1.19m, 1.39°) LiSA (0.85m, 1.11°) LightLoc (**0.82m**, 1.16°)

Figure 6. LiDAR localization results on the QEOxford [1, 8] dataset. The ground truth and prediction are black and red lines, respectively. The star denotes the first frame. The caption of each subfigure shows the mean position error (m) and orientation error (°). For each trajectory, we highlight the **best** and second-best results.



DiffLoc (3.57m, **0.88°**)

SGLoc (3.01m, 1.91°)

LiSA (2.36m, 1.29°)

LightLoc (**2.33m**, 1.21°)

DiffLoc (3.65m, **0.68°**)

SGLoc (4.07m, 2.07°)

LiSA (3.47m, 1.43°)

LightLoc (**3.19m**, 1.34°)

DiffLoc (4.03m, **0.70°**)

SGLoc (3.37m, 1.89°)

LiSA (3.19m, 1.34°)

LightLoc (**3.11m**, 1.24°)

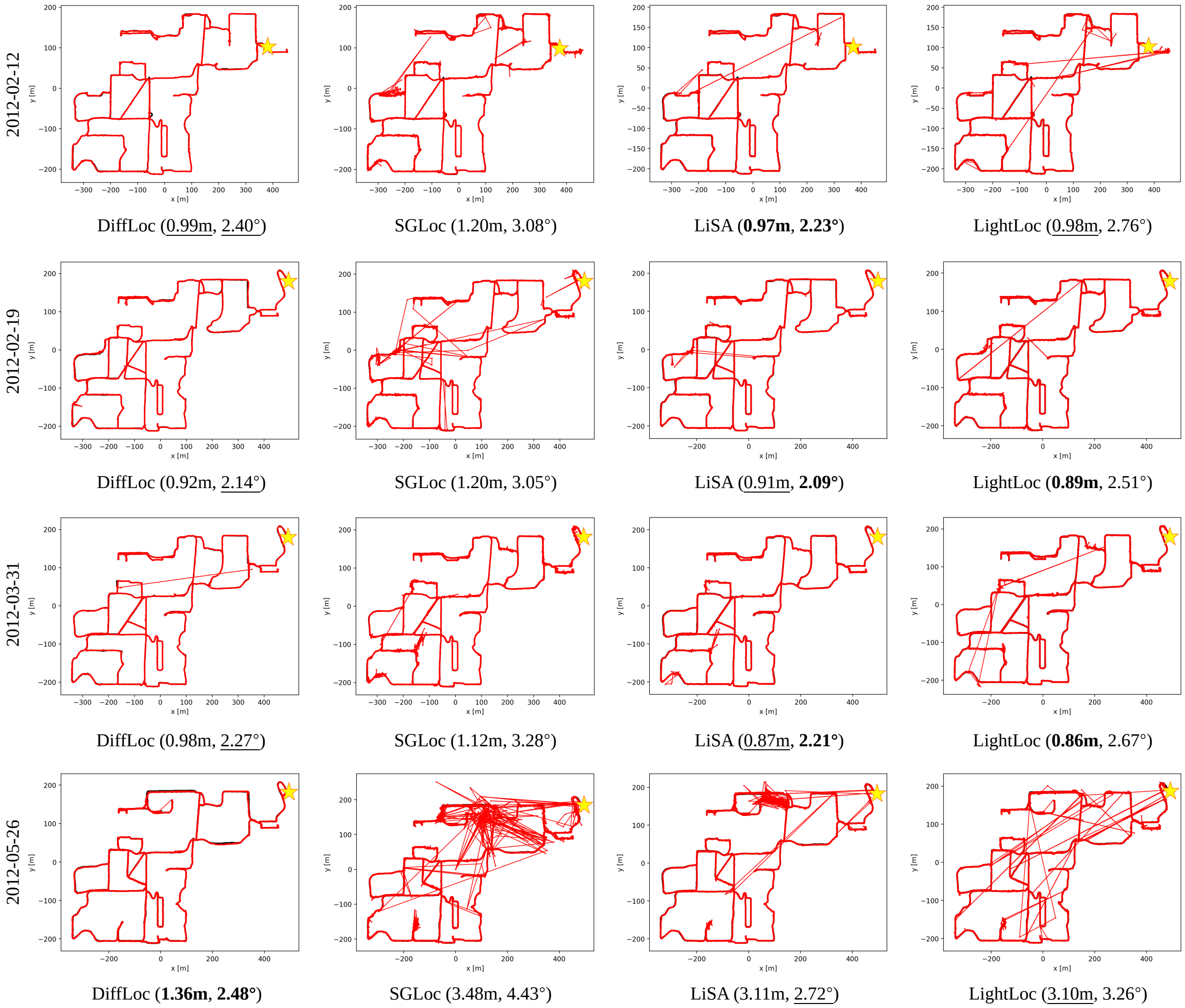
DiffLoc (2.86m, **0.60°**)

SGLoc (2.12m, 1.66°)

LiSA (**1.95m**, 1.23°)

LightLoc (2.05m, 1.20°)

Figure 7. LiDAR localization results on the Oxford [1] dataset. The ground truth and prediction are black and red lines, respectively. The star denotes the first frame. The caption of each subfigure shows the mean position error (m) and orientation error (°). For each trajectory, we highlight the **best** and second-best results.



DiffLoc (0.99m, 2.40°)

SGLoc (1.20m, 3.08°)

LiSA (**0.97m**, **2.23°**)

LightLoc (0.98m, 2.76°)

DiffLoc (0.92m, 2.14°)

SGLoc (1.20m, 3.05°)

LiSA (0.91m, **2.09°**)

LightLoc (**0.89m**, 2.51°)

DiffLoc (0.98m, 2.27°)

SGLoc (1.12m, 3.28°)

LiSA (0.87m, **2.21°**)

LightLoc (**0.86m**, 2.67°)

DiffLoc (**1.36m**, **2.48°**)

SGLoc (3.48m, 4.43°)

LiSA (3.11m, 2.72°)

LightLoc (3.10m, 3.26°)

Figure 8. LiDAR localization results on the NCLT [11] dataset. The ground truth and prediction are black and red lines, respectively. The star denotes the first frame. The caption of each subfigure shows the mean position error (m) and orientation error (°). For each trajectory, we highlight the **best** and second-best results.