

Robust outdoor trajectory mapping using CNN features and loop closure optimization

Kamran Kazi ^{a, b}, Arbab Nighat Kalhoro ^{a, b}, Farida Memon ^b, Tarique Rafique Memon ^c, Azam Rafique Memon ^{b, d*},

^a *Institute of Information & Communication Technologies, Mehran University of Engineering & Technology, Jamshoro, Pakistan*

^b *Department of Electronic Engineering, Mehran University of Engineering & Technology, Jamshoro, Pakistan*

^c *Department of Electronic Engineering, Quaid-e-Awam University of Engineering, Science & Technology, Nawabshah*

^d *College of Engineering, Prince Sultan University, Riyadh, Saudi Arabia*

* Corresponding author: Azam Rafique Memon, Email: azammemon2000@gmail.com

Received: 24 May 2025, Revised: 23 June 2025, Accepted: 30 June 2025, Published: 01 July 2025

KEY WORDS

Visual SLAM
CNN-based Feature Extraction
Outdoor Cluttered Environment
Trajectory Optimization
Odometry Error

ABSTRACT

Autonomous navigation in cluttered outdoor environments remains a significant challenge for robotic systems due to dynamic obstacles, occlusions, and feature degradation. Traditional Simultaneous Localization and Mapping (SLAM) systems, reliant on low-level geometric features, often fail to accurately estimate trajectory in such conditions. This paper introduces a robust framework for outdoor trajectory mapping that synergizes deep learning-based feature extraction with loop closure detection and optimization to address these limitations. To find a suitable layer to obtain features from the CNN model, 313,746 filters were checked to find a filter that has the least odometry error from three pre-trained CNN models, the ConvNeXtXLarge is leveraged to extract high-level semantic features from monocular images, enabling resilient optical flow estimation even in scenes with transient objects and lighting variations. These features are fused with inertial measurement unit (IMU) data to compute initial trajectories, minimizing scale ambiguity. To counteract accumulated drift, a hybrid optimization strategy is proposed: Bag of Words (BoW)-based loop closure detection identifies revisit locations, while an algorithm is developed to optimize the map. Experimental validation on the KITTI dataset demonstrates a significant reduction in offset error compared to state-of-the-art methods like DeepVO and TartanAir.

1. Introduction

Autonomous navigation in dynamic outdoor environments remains a critical challenge for robotic systems, particularly in cluttered settings where features are transient and environmental conditions fluctuate unpredictably. Traditional Simultaneous Localization and Mapping (SLAM) systems, such as ORB-SLAM2 [1] and DeepVO [2], rely heavily on low-level visual features (e.g., corners, edges) for pose estimation and mapping. However, these methods

struggle in outdoor cluttered environments due to frequent occlusions, abrupt scene changes (e.g., moving objects, weather variations), and feature inconsistency. For instance, a robot revisiting a previously mapped area may encounter altered landmarks caused by dynamic obstacles or seasonal changes, leading to tracking failures and map corruption. Furthermore, when the robot revisits the mapped area, it should optimize the map, reducing accumulated odometry errors over time [3], [4]. These

limitations hinder the scalability and reliability of SLAM [5] in real-world applications such as autonomous delivery, disaster response, and long-term environmental monitoring.

To address these challenges, recent advancements have explored integrating semantic segmentation [6], multi-map fusion [3], and deep learning-based feature extraction [7]. Semantic SLAM methods leverage object-level information to improve feature consistency, while map optimization attempts to correct map errors using identified landmarks or features. However, such approaches often require extensive computational resources, struggle with real-time performance, and remain vulnerable to environmental dynamics. For example, semantic segmentation models may fail under extreme lighting or weather conditions, and manual feature matching is prone to errors. Additionally, while deep learning-based odometry (e.g., TartanAir [8]) reduces dependency on geometric features, it introduces challenges in generalizing to unseen environments and managing cumulative drift over long trajectories. These shortcomings underscore the need for a system that balances robustness, efficiency, and adaptability in dynamic outdoor settings.

In this work, we propose a novel visual SLAM framework tailored for cluttered outdoor environments, combining deep learning-based feature extraction with advanced trajectory optimization. Our contributions are threefold: (1) Three pre-trained CNN Models are taken and checked for layers and filters that show features that are less affected by illumination changes, viewpoint changes, and the presence of occlusions. (2) High-level features from the least odometry filter are obtained that are not affected by environmental variations, enabling robust optical flow estimation from monocular images. (3) A hybrid optimization algorithm integrates local loop closure detection (via Bag of Words [9]) with global bundle adjustment to minimize accumulated odometry errors. Experimental validation on the KITTI dataset [10] demonstrates superior accuracy over state-of-the-art methods, with a significant reduction in offset error and efficient handling of occlusions and dynamic obstacles.

The remainder of this paper is organized as follows: Section 2 reviews related work in visual SLAM and trajectory mapping. Section 3 details the proposed methodology, including feature extraction and trajectory optimization. Section 4 presents experimental results and comparisons with baseline methods, and Section 5 concludes the paper.

2. Literature Review

The evolution of Simultaneous Localization and Mapping (SLAM) has been driven by advancements in feature extraction, optimization, and adaptability to dynamic environments. This section synthesizes key contributions across these domains, focusing on their relevance to mapping in cluttered outdoor settings.

2.1 Traditional and Feature-Based SLAM

Early SLAM systems relied on geometric features and probabilistic frameworks. ORB-SLAM2 [1], a seminal monocular SLAM framework, utilized Oriented FAST and Rotated BRIEF (ORB) features for real-time localization and mapping. Its extension, ORB-SLAM3 [11], incorporated multi-map fusion but struggled with dynamic environments [12]. Similarly, Direct Sparse Odometry (DSO) [13] bypassed feature extraction by optimizing photometric error directly, achieving robustness in low-texture environments. However, these methods depend heavily on static scenes and fail in cluttered or dynamic settings [4].

2.2 Semantic and Object-Aware SLAM

To address dynamic environments, recent works have integrated semantic segmentation and object detection. Mahmoud and Atia [6] improved map consistency by combining semantic segmentation with layout estimation, while OA-SLAM [14] leveraged object-level features for relocalization. These approaches enhance feature persistence but require significant computational overhead and struggle with occlusions. RWT-SLAM [15] proposed robust feature matching in weak-textured environments using wavelet transforms, yet its scalability to multisession scenarios remains unproven.

2.3 Deep Learning in SLAM

Deep learning has revolutionized feature extraction and odometry estimation. DeepVO [2] employed recurrent convolutional networks (RCNNs) for end-to-end visual odometry, reducing dependency on geometric constraints. TartanAir [8] extended this by training on diverse synthetic data, improving generalization. Authors in DXSLAM [7] have combined deep features with traditional SLAM pipelines for real-time performance. While these methods excel in single-session scenarios, they accumulate drift over long trajectories and lack mechanisms for multisession map alignment [3].

Table 1

Details of Models used to find filters with least odometry error

Model	Accuracy		Parameters	Layers	Number of Filters		
	Top-1	Top-5			Total	Used	Excluded from Analysis
ConvNeXtXLarge	86.7%	-	350.1 million	297	332,806	36,096	296,710
EfficintNetV2L	85.7%	97.5%	119.0 million	1031	1,253,254	261,216	992,038
NASNetLarge	82.5%	96.0%	88.9 million	1041	502,407	16,434	485,973

2.4 Multisession and Multi-Map Fusion

Multisession mapping requires robust strategies to align disjointed maps. Giubilato et al. [16] proposed sub-maps for planetary rovers, merging sessions via stereo camera data, but their method assumes static environments. MS-Mapping [17] introduced Wasserstein-based keyframe selection for LiDAR multisession mapping, offering computational efficiency but limited applicability to visual SLAM. Menezes et al. [18] adapted RatSLAM for multisession scenarios using bio-inspired place recognition, though their approach lacks precision in large-scale outdoor environments.

2.5 Loop Closure Detection and Map Optimization

Loop closure detection is critical for error correction. Bag of Words (BoW) [9] and FAB-MAP [19] pioneered appearance-based methods, while SeqSLAM [20] matched image sequences under extreme perceptual changes. Recent works like PALoc [21] integrated prior-assisted trajectory generation to reduce uncertainty, and Gaussian Splatting SLAM [22] optimized map consistency using probabilistic surfels. However, these methods often fail in highly dynamic or occluded scenes [23].

2.6 Dynamic Environment Handling

Dynamic SLAM systems aim to filter transient objects. DP-SLAM [24] used probability estimations to distinguish static and dynamic features, while DE-SLAM [25] employed motion consistency checks. These methods improve robustness but require manual parameter tuning and struggle with sudden environmental changes.

3. Methodology

The proposed framework for trajectory mapping and correction in outdoor cluttered environments integrates deep learning-based feature extraction from

a filter that has features that are less affected by different lighting conditions, viewpoint changes, and the presence of occlusions.

3.1 System Overview

The proposed system takes features from the selected intermediate layer of the CNN model, uses these features to find optical flow, and estimates the map by just using the IMU and monocular camera. The overall system is shown in Fig. 1.

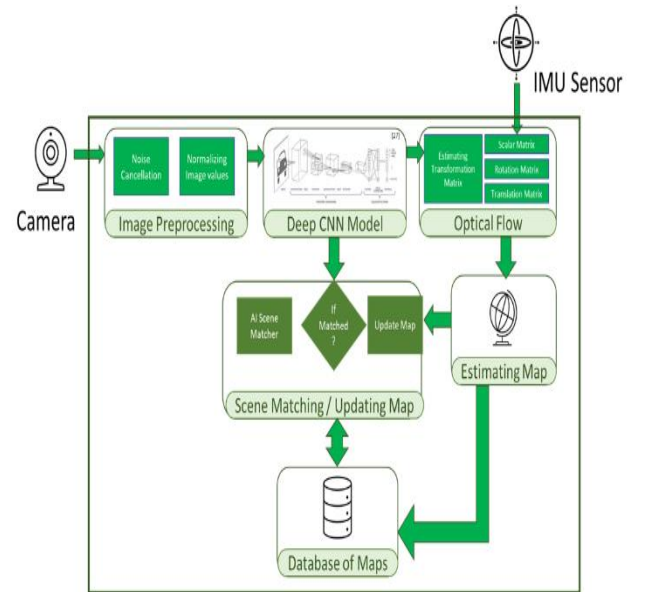


Fig. 1. System overview for creating maps of the environment and optimizing them

The IMU-camera extrinsic calibration (spatial transformation between sensors) was performed using the Kalibr toolbox [26], with time synchronization achieved via hardware triggering at 20 Hz. Calibration parameters were validated using the KITTI dataset's ground truth trajectories. The camera calibration was also performed using chessboard markers to find the camera matrix and distortion coefficients, which are required for map estimation. The system takes images from a monocular camera, applying image pre-processing techniques by considering the camera

matrix (equation 1), focal length (equation 2), principal point (equation 3), and distortion coefficients (equation 4) into account. These images are then fed to the *ConvNeXtXLarge* Model, and features from filter # 10 of the *convnet_xlarge_stem* layer are obtained because it shows the minimum odometry error among all layers and filters of the 3 models having top-1 accuracy and unique architecture. The details about models, their accuracy, total parameters, total number of layers, total filters in each model, and number of filters used to analyze odometry error are detailed in Table 1.

$$\text{Camera Matrix} = \begin{bmatrix} 718.85 & 0 & 607.19 \\ 0 & 718.85 & 185.21 \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

$$\text{Focal Length} = \begin{bmatrix} 7.1885 \times 10^2 \\ 7.1885 \times 10^2 \end{bmatrix} \quad (2)$$

$$\text{Principal Point (PP)} = [6.0719 \times 10^2 \quad 1.8521 \times 10^2] \quad (3)$$

$$\text{Distortion Coefficient} = [-0.0119846 \quad -0.05403985 \quad -0.29219686] \quad (4)$$

In Table 1, we used all filters in the Used Filters column for each filter to find a suitable filter with minimum odometry error. There were around 313,746 filters that were checked, and out of all of them, the minimum error was found in filter # 10 of *convnet_xlarge_stem* layer. This layer is then used to find odometry errors from all datasets because this layer has shown the least odometry error despite having different viewpoints, illumination variations, and the presence of occlusions. The rotation R and translation T matrix from these CNN features [27] is then estimated using the method described in [28], which is then used to plot the estimated map.tra

3.2 Offset Error Estimation

To compare the proposed method with other state-of-the-art methods, the odometry error is considered as the metric to identify the best method, which has the least odometry error. This odometry error is obtained as:

$$E_x = \sum ||P_{xg} - P_{xe}|| \quad (5)$$

$$E_y = \sum ||P_{yg} - P_{ye}|| \quad (6)$$

$$\text{Average offset Error} = \frac{E_x + E_y}{N_f} \quad (7)$$

where,

P = position

x = x-axis

y = y-axis

g = ground truth

e = estimated value

E = offset error

N_f = Number of total frames

3.3 Optimizing the Map

The estimated map is not accurate and it still has an error, which is shown in Table 2. To optimize it and reduce the accumulated error, consider the estimated trajectory as:

$$\text{traj} = [(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)] \quad (8)$$

To find the accumulated error after finding the loop closure location as l and the current location as c . The error can be estimated as:

$$E_{lc} = \text{traj}[l] - \text{traj}[c] \quad (9)$$

This error between current location c and loop closure location l , needs to be reduced, as both points l and c , resemble the same location, so they should be at the same point in the map. But this change should be reflected throughout the loop, from l to c . Hence, we need to estimate transition vector tv , which makes a smooth transition within a loop, so c and l become the same point in the map.

$$tv = [(E_{lcx}^z, E_{lcy}^z) \dots (0,0)] \quad (10)$$

The transition vector starts from E_{lc} up to 0, the vector is of size z , where z is the number of map points between l and c .

So now the trajectory will be updated as:

$$\text{traj}_{\text{updated}} = \text{traj}[0, l] + (\text{traj}[l, c] - tv) \quad (11)$$

The $\text{traj}_{\text{updated}}$ is the corrected trajectory, which has the same previous trajectory from the first map point till loop closure detected point l , and then the remaining map points from l to c are corrected by subtracting the transition vector from the remaining map points and re-creating the complete map.

4. Results and Discussion

To evaluate the performance of the proposed method, we performed various tests on 6 benchmarks of *Kitti* Sequence Datasets [10], which include occlusions, dynamic objects, illumination variations, and

viewpoint changes. The benchmarks that are used include sequences 00, 02, 05, 06, 08 and 09.

4.1 Map Estimation from Image Sequences

To create the estimated map, the rotation R and translation T matrices are used and these two matrices are updated with every new frame obtained as described in the equations below:

$$R_c = R \cdot R_c \quad (12)$$

$$T_c = T_c + Scale \times R_c \cdot T \quad (13)$$

Where:

$Scale$ = distance travelled between two consecutive image frames obtained from the IMU sensor

R_c and T_c are the combined rotation and translation matrix

The obtained maps from various sequences are:

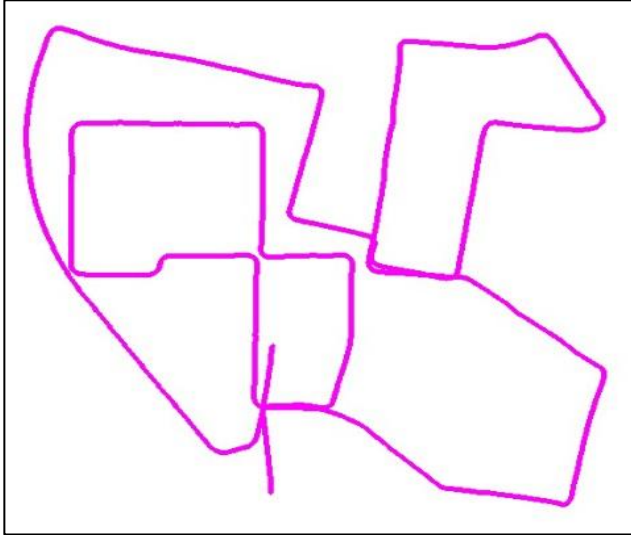


Fig. 2. Estimated map from the Kitti Sequence 00 dataset

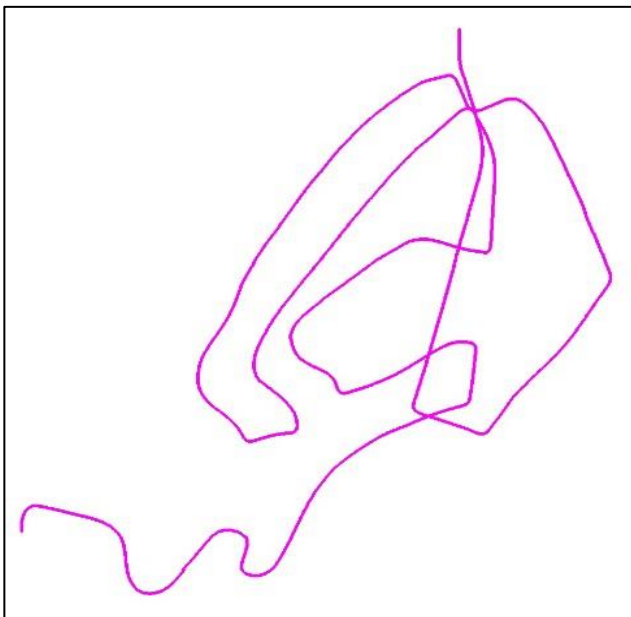


Fig. 3. Estimated map from the Kitti Sequence 02 dataset

These results shown in Fig. 2-7 show that maps are almost close to the original shape without even optimizing the maps, just from rotation and translation matrices and when compared with state-of-the-art methods like DeepVO [2] and TartanAir [8], which are further discussed in section 4.2, shows that error in our method even without optimization is much lower than their methods.

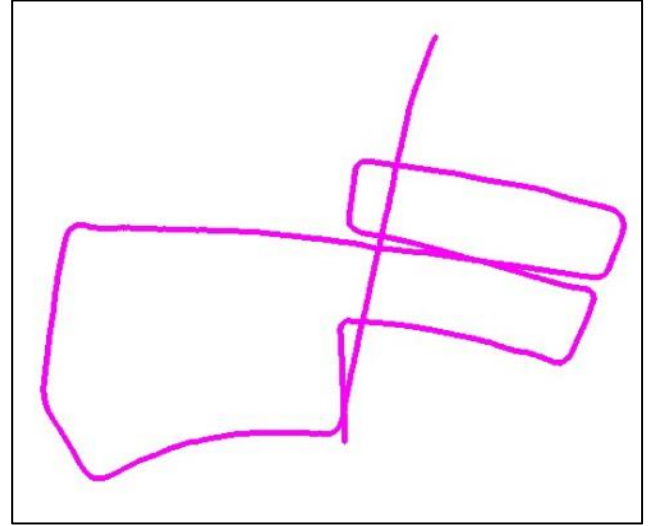


Fig. 4. Estimated map from the Kitti Sequence 05 dataset

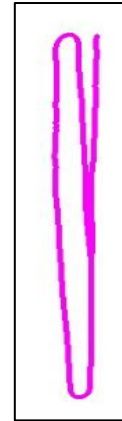


Fig. 5. Estimated map from the Kitti Sequence 06 dataset

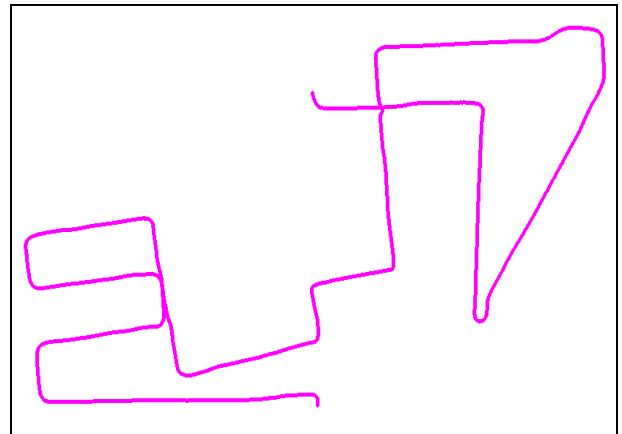


Fig. 6. Estimated map from the Kitti Sequence 08 dataset

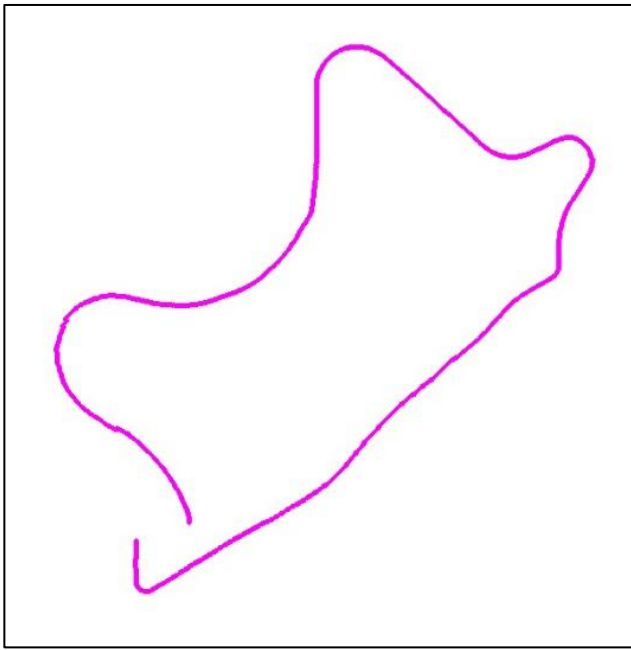
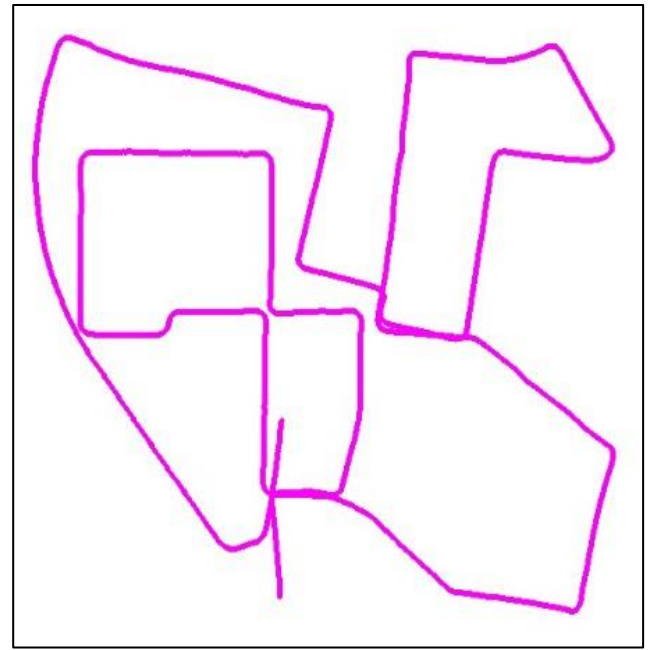


Fig. 7. Estimated map from the Kitti Sequence 09 dataset

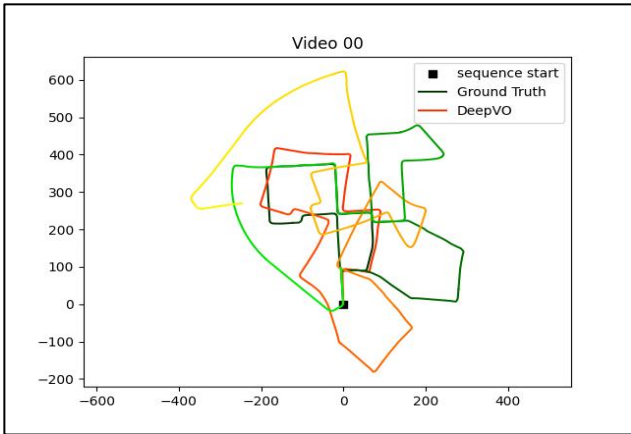
4.2 Comparison of Estimated maps with State-of-the-art methods

These estimated maps are compared with DeepVO and TartanAir and the results are shown in the Fig. 8-18.

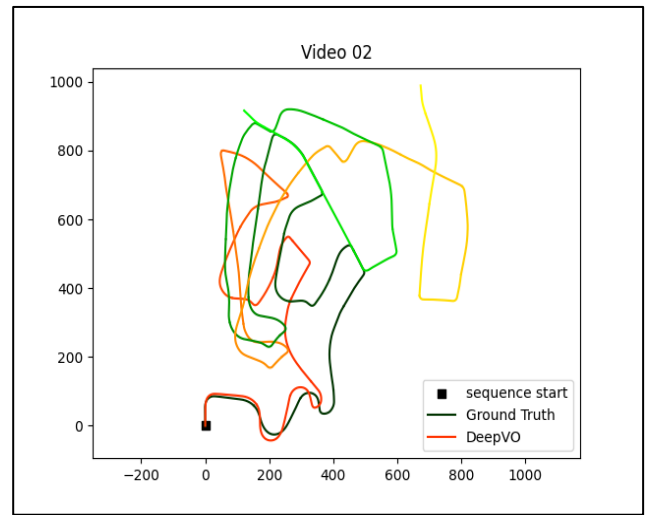


(c) Results of the proposed method

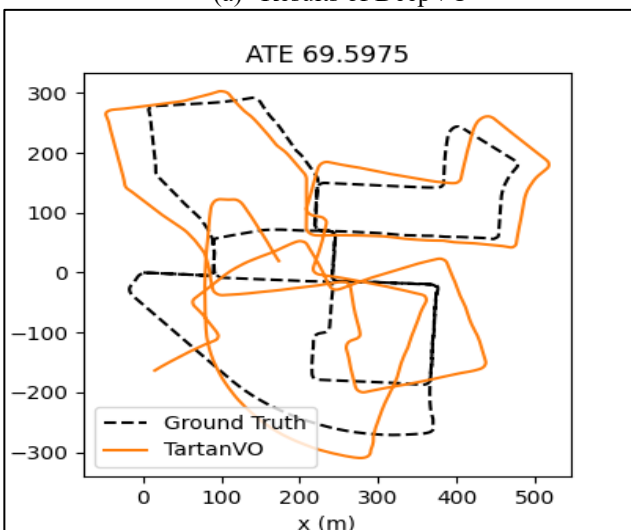
Fig. 8. Comparison Results on Kitti Sequence 00 Dataset



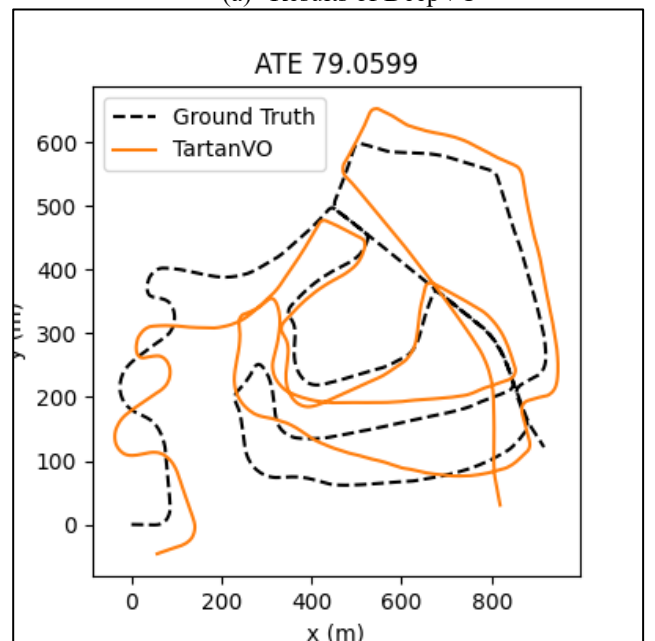
(a) Results of DeepVO



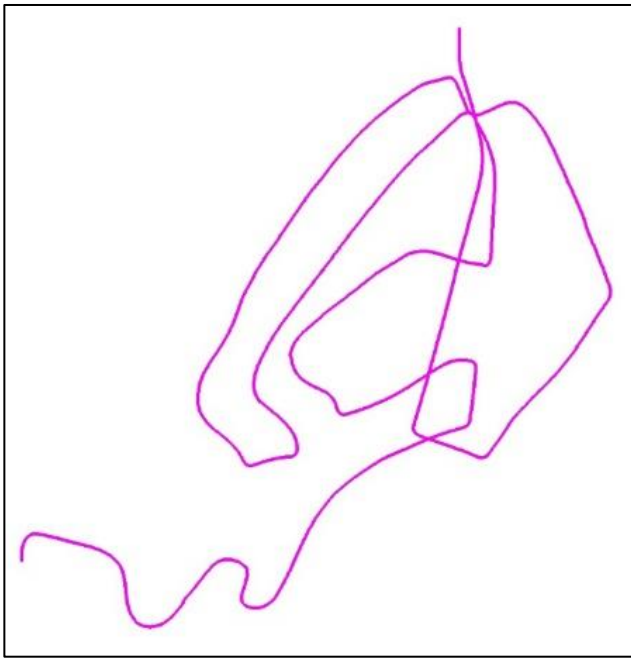
(a) Results of DeepVO



(b) Results of TartanAir

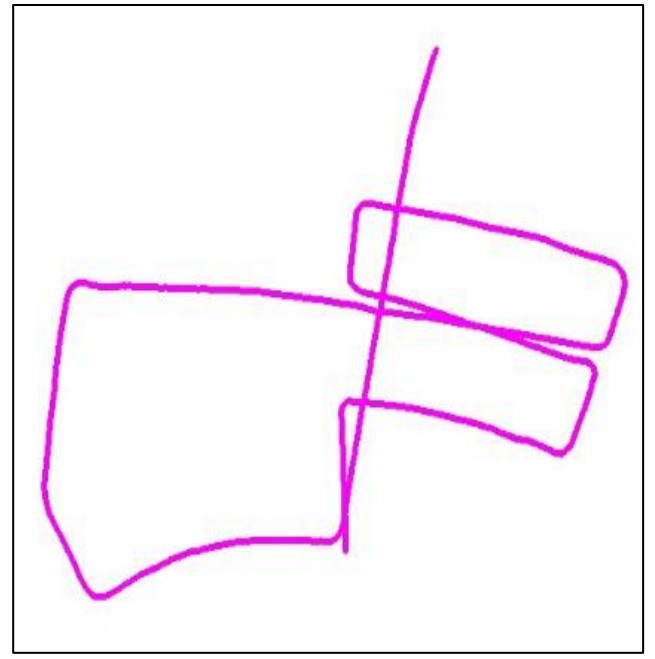


(b) Results of TartanAir



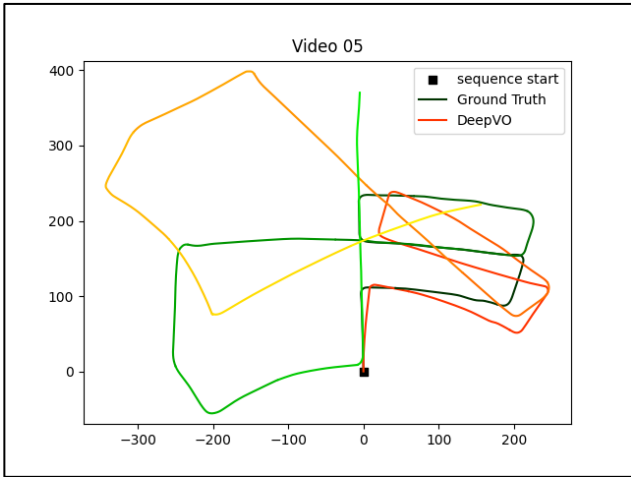
(c) Results of the proposed method

Fig. 9. Comparison Results on Kitti Sequence 02 Dataset

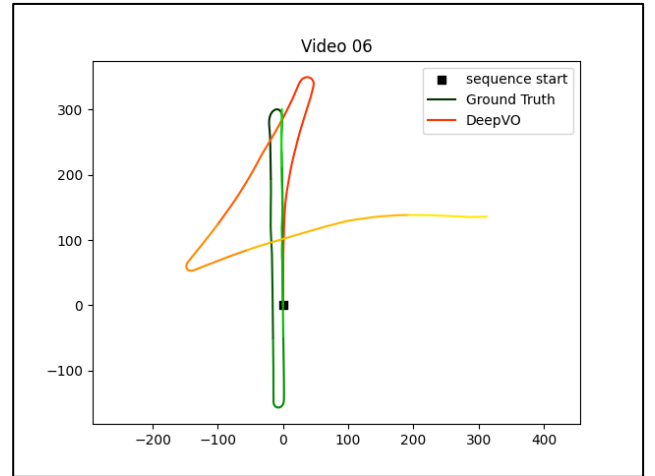


(c) Results of the proposed method

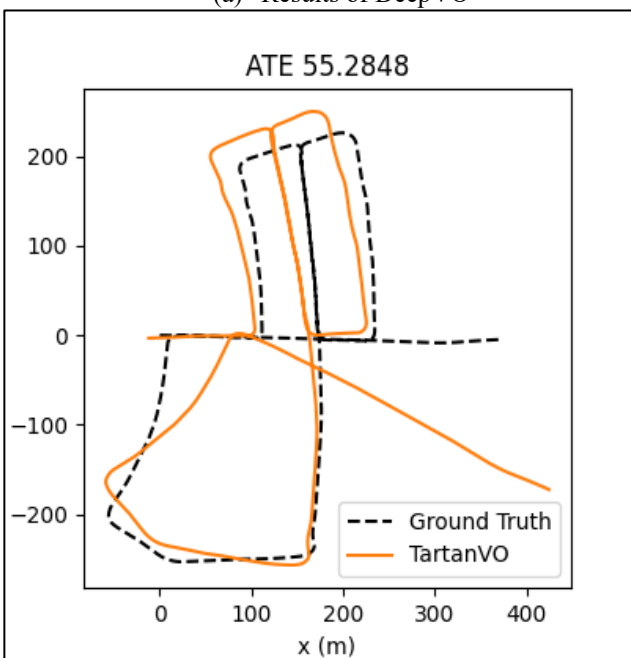
Fig. 10. Comparison Results on Kitti Sequence 05 Dataset



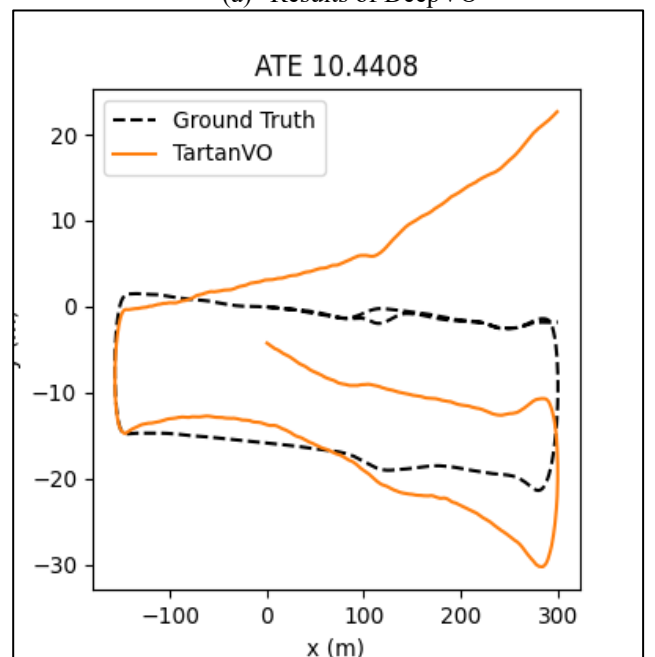
(a) Results of DeepVO



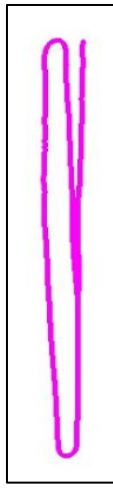
(a) Results of DeepVO



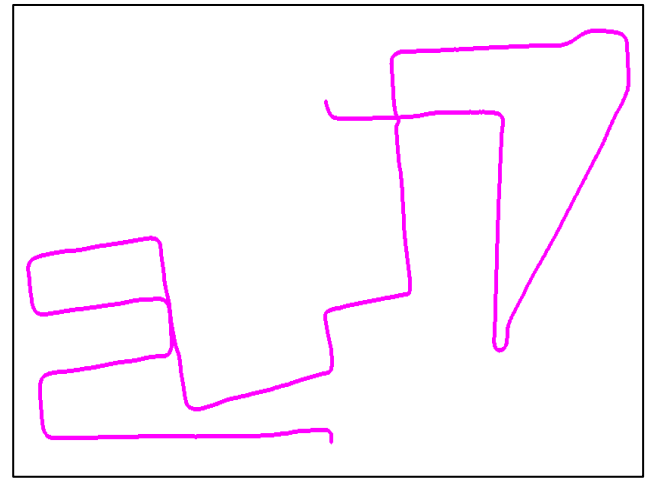
(b) Results of TartanAir



(b) Results of TartanAir



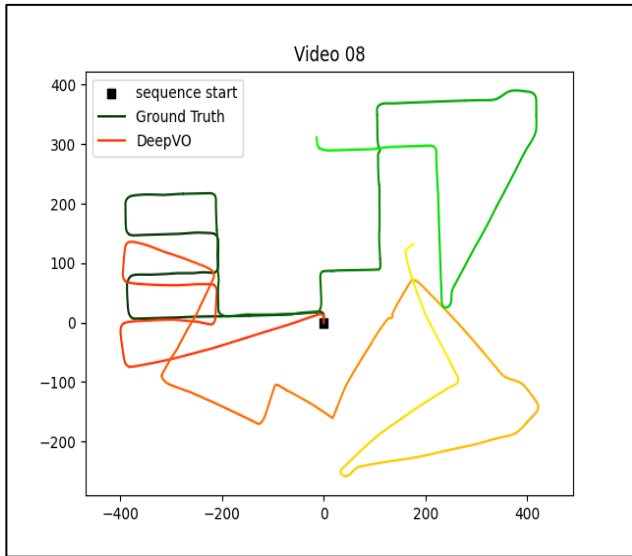
(c) Results of the proposed method



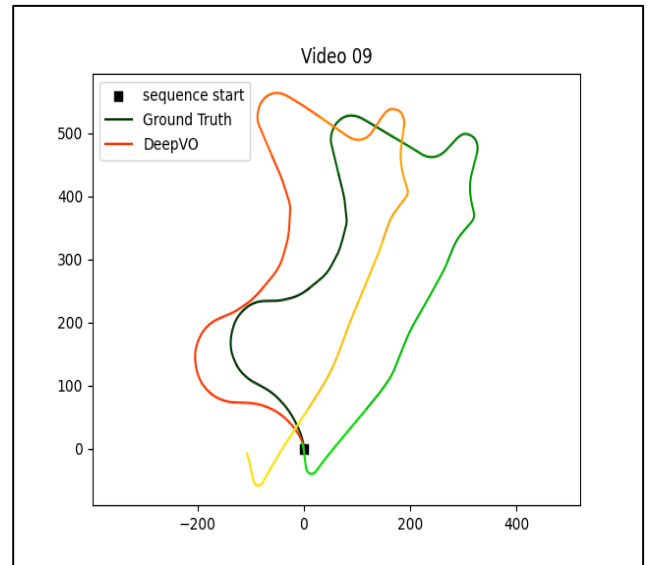
(c) Results of the proposed method

Fig. 11. Comparison Results on Kitti Sequence 06 Dataset

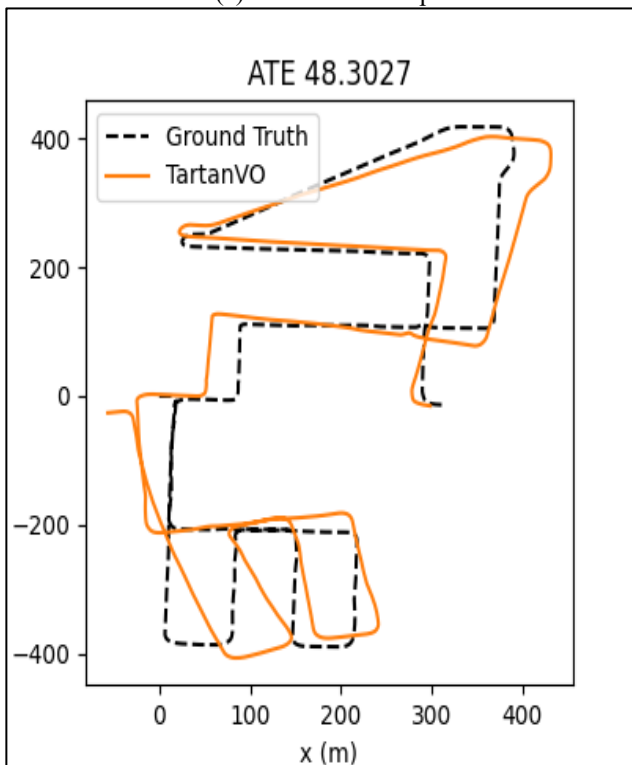
Fig. 12. Comparison Results on Kitti Sequence 08 Dataset



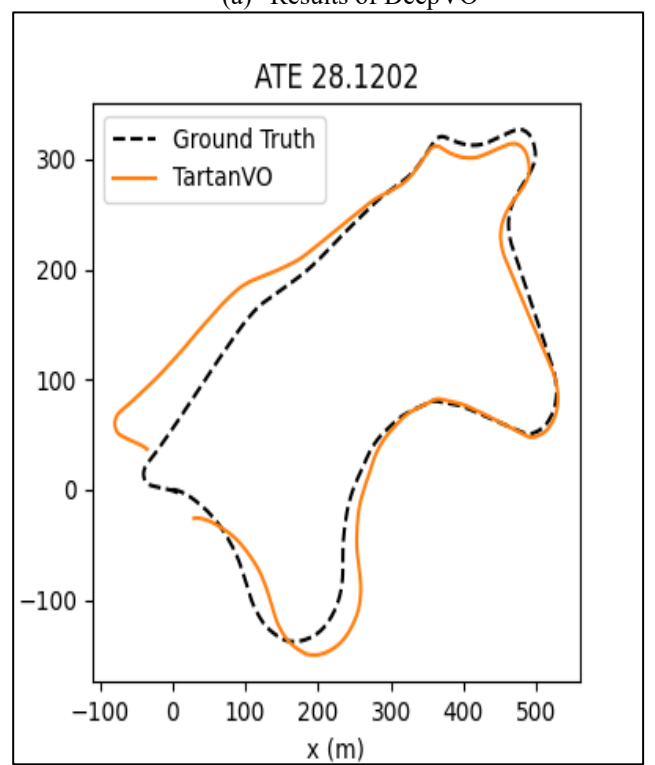
(a) Results of DeepVO



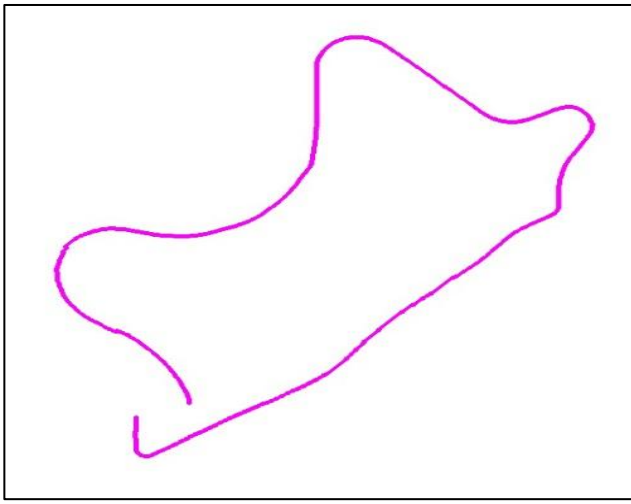
(a) Results of DeepVO



(b) Results of TartanAir



(b) Results of TartanAir



(c) Results of the proposed method

Fig. 13. Comparison Results on Kitti Sequence 09 Dataset
The offset error is shown in Table 2, which shows the proposed method has the least odometry error even without optimization.

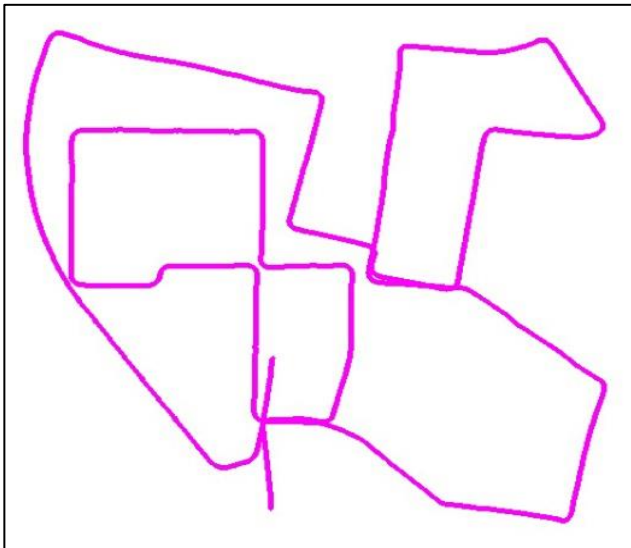
Table 2

Comparison Results with State-of-the-Art methods

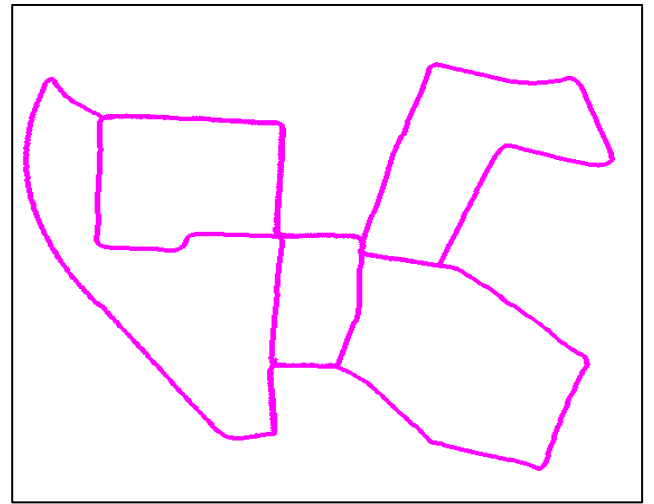
Kitti Sequence Number	Frames	Average offset Error (m)		
		DeepVO	TartanAir	Ours
00	4541	13,227.92	124,361.71	476.40
02	4661	13,005.68	317,924.60	1,451.84
05	2761	7,250.48	133,977.30	286.65
06	1101	7,774.23	45,526.16	310.44
08	4071	23,942.91	203,107.88	374.23
09	1591	4,379.55	110,614.15	536.42

4.3 Corrected Trajectories after Loop Closure Detection

Once the loop closure is obtained, which is obtained using the Bag of Words [29], [30], [31] dictionary, the estimated maps are corrected and the results are shown below.

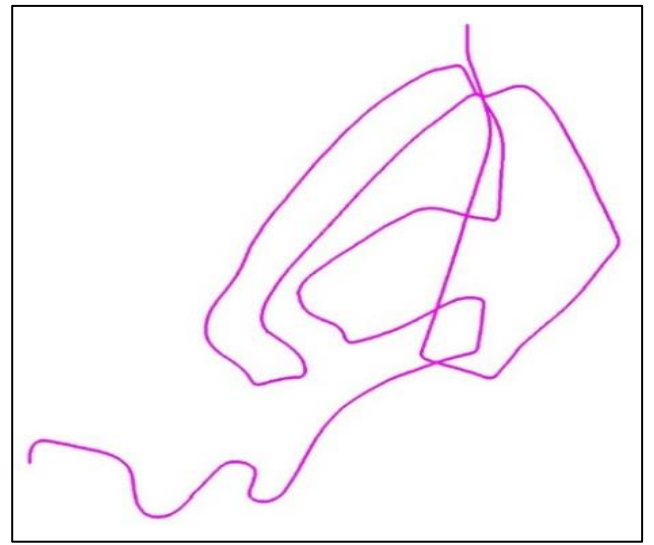


(a) Unoptimized Map

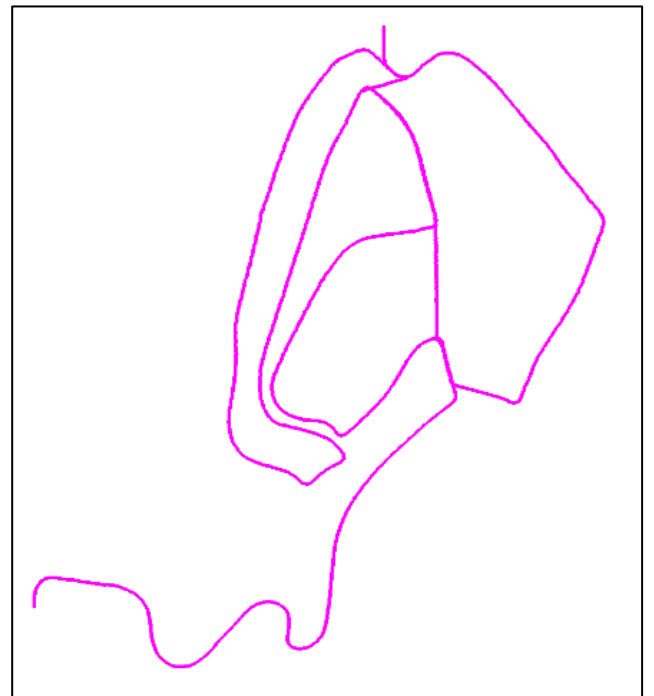


(b) Corrected Map

Fig. 14. Kitti Sequence 00 optimization

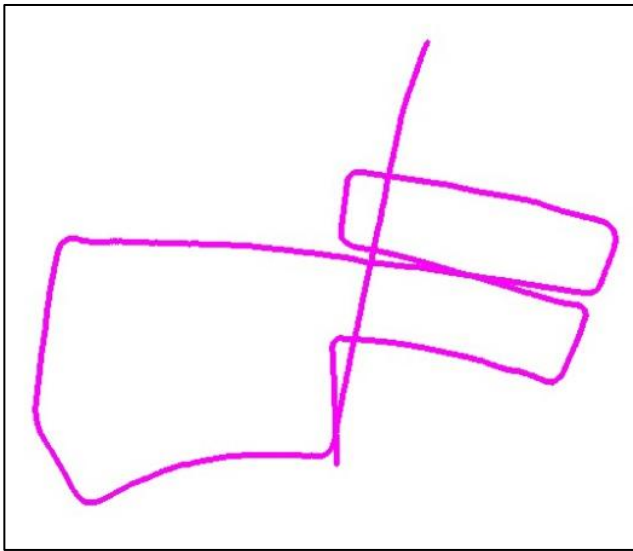


(a) Unoptimized Map

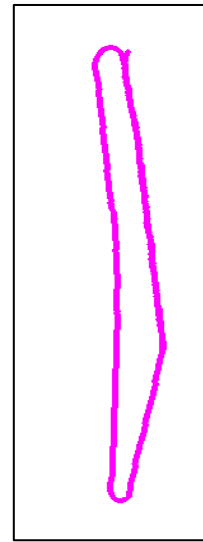


(a) Corrected Map

Fig. 15. Kitti Sequence 02 optimization

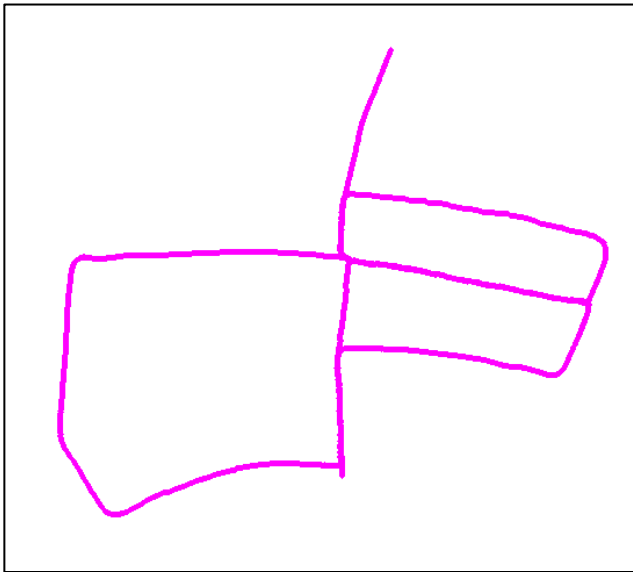


(a) Unoptimized Map



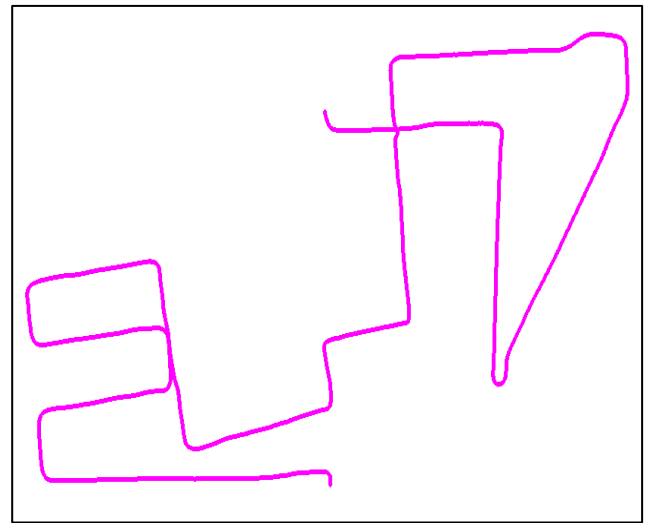
(b) Corrected Map

Fig. 17. Kitti Sequence 06 optimization

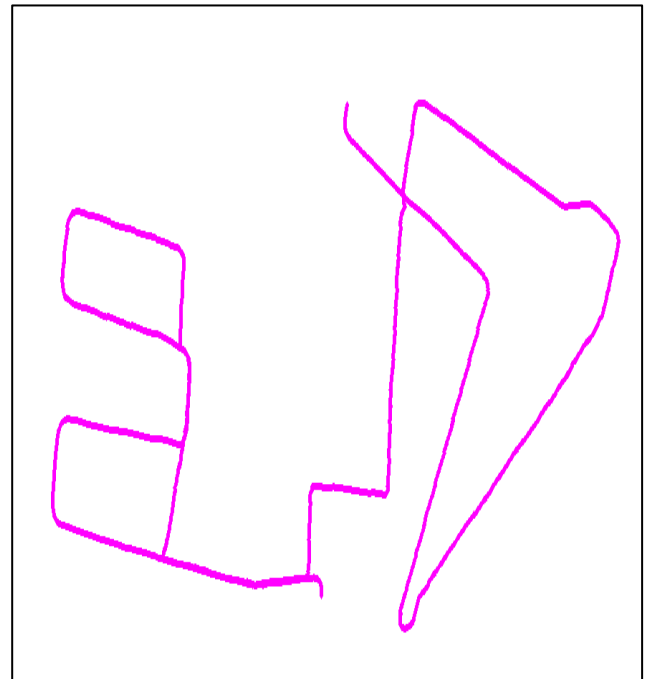


(b) Corrected Map

Fig. 16. Kitti Sequence 05 optimization

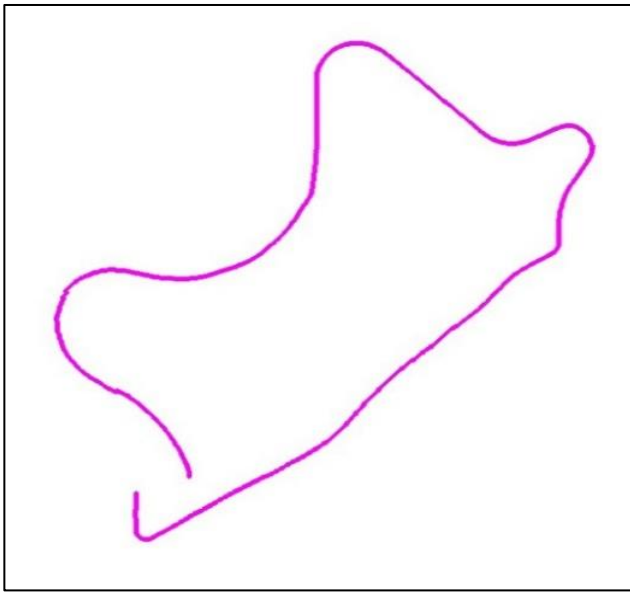


(a) Unoptimized Map

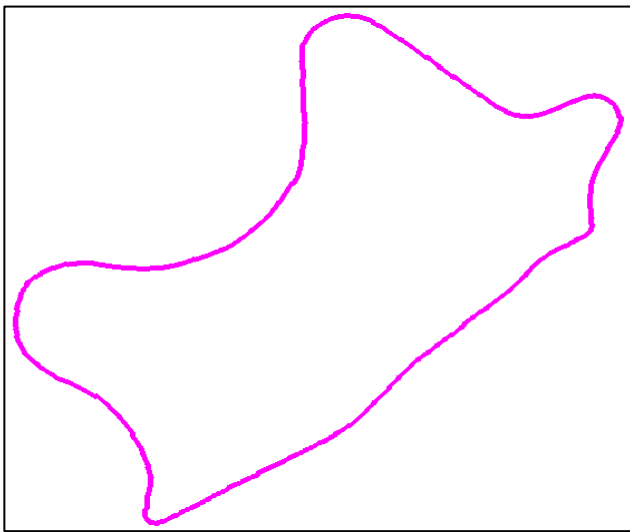


(b) Corrected Map

Fig. 18. Kitti Sequence 08 optimization



(a) Unoptimized Map



(b) Corrected Map

Fig. 19. Kitti Sequence 09 optimization

5. Conclusion

Accurate trajectory estimation in cluttered outdoor environments hinges on robust feature extraction and error correction. This paper introduced a systematic framework that addresses these challenges through a data-driven approach. After evaluating 313,746 filters across three pre-trained CNN models (ConvNeXtXLarge, EfficientNetV2L, NASNetLarge), filter #10 from the convnet_xlarge_stem layer of ConvNeXtXLarge was identified as the optimal choice, exhibiting the lowest odometry error due to its ability to capture high-level semantic features resilient to environmental clutter. These features, fused with IMU data, enabled precise optical flow estimation, while a novel optimization algorithm—combining Bag of Words (BoW)-based loop closure detection and bundle adjustment—corrected accumulated drift by redistributing errors across trajectory segments.

Validation on the KITTI dataset and a custom MUET outdoor dataset demonstrated the framework's superiority, achieving a significant reduction in odometry error compared to DeepVO and TartanAir. The targeted use of filter #10 not only improved feature consistency in dynamic scenes but also reduced computational overhead by eliminating redundant filters.

6. References

- [1] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017, doi: 10.1109/TRO.2017.2705103.
- [2] S. Wang, R. Clark, H. Wen, and N. Trigoni, "DeepVO: Towards end-to-end visual odometry with deep Recurrent Convolutional Neural Networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2017, pp. 2043–2050. doi: 10.1109/ICRA.2017.7989236.
- [3] D. Ming and X. Wu, "Research on Monocular Vision SLAM Algorithm for Multi-map Fusion and Loop Detection," in *2022 6th International Conference on Automation, Control and Robots, ICACR 2022*, 2022. doi: 10.1109/ICACR55854.2022.9935516.
- [4] Z. Xu, Z. Rong, and Y. Wu, "A survey: which features are required for dynamic visual simultaneous localization and mapping?," *Vis Comput Ind Biomed Art*, vol. 4, no. 1, p. 20, Dec. 2021, doi: 10.1186/s42492-021-00086-w.
- [5] A. Hussain, A. R. Memon, H. Wang, Y. Wang, Y. Miao, and X. Zhang, "S-VIT: Stereo Visual-Inertial Tracking of Lower Limb for Physiotherapy Rehabilitation in Context of Comprehensive Evaluation of SLAM Systems," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 4, pp. 1550–1562, Oct. 2021, doi: 10.1109/TASE.2020.3010415.
- [6] A. Mahmoud and M. Atia, "Improved Visual SLAM Using Semantic Segmentation and Layout Estimation," *Robotics*, vol. 11, no. 5, p. 91, Sep. 2022, doi: 10.3390/robotics11050091.
- [7] D. Li *et al.*, "DXSLAM: A Robust and Efficient Visual SLAM System with Deep Features," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2020, pp. 4958–4965. doi: 10.1109/IROS45743.2020.9340907.

- [8] W. Wang *et al.*, “TartanAir: A Dataset to Push the Limits of Visual SLAM,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2020, pp. 4909–4916. doi: 10.1109/IROS45743.2020.9341801.
- [9] D. Galvez-López and J. D. Tardos, “Bags of Binary Words for Fast Place Recognition in Image Sequences,” *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012, doi: 10.1109/TRO.2012.2197158.
- [10] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2012, pp. 3354–3361. doi: 10.1109/CVPR.2012.6248074.
- [11] R. Elvira, J. D. Tardos, and J. M. M. Montiel, “ORB-SLAM-Atlas: a robust and accurate multi-map system,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Nov. 2019, pp. 6253–6259. doi: 10.1109/IROS40897.2019.8967572.
- [12] A. Memon, F. K. Shaikh, and E. Felemban, “Experimental Evaluation of Vehicle-to-Vehicle based Data Transfer,” in *International Conference on Information and Communication Technology Research (ICTRC2015)*, 2015, pp. 274–277. doi: 10.1109/ICTRC.2015.7156475.
- [13] J. Engel, V. Koltun, and D. Cremers, “Direct Sparse Odometry,” *IEEE Trans Pattern Anal Mach Intell*, vol. 40, no. 3, pp. 611–625, Mar. 2018, doi: 10.1109/TPAMI.2017.2658577.
- [14] M. Zins, G. Simon, and M.-O. Berger, “OA-SLAM: Leveraging Objects for Camera Relocalization in Visual SLAM,” in *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, IEEE, Oct. 2022, pp. 720–728. doi: 10.1109/ISMAR55827.2022.00090.
- [15] Q. Peng, Z. Xiang, Y. Fan, T. Zhao, and X. Zhao, “RWT-SLAM: Robust Visual SLAM for Highly Weak-textured Environments,” Jul. 2022, doi: <https://doi.org/10.48550/arXiv.2207.03539>.
- [16] R. Giubilato *et al.*, “Relocalization With Submaps: Multi-Session Mapping for Planetary Rovers Equipped With Stereo Cameras,” *IEEE Robot Autom Lett*, vol. 5, no. 2, pp. 580–587, Apr. 2020, doi: 10.1109/LRA.2020.2964157.
- [17] X. Hu, J. Wu, J. Jiao, W. Zhang, and P. Tan, “MS-Mapping: Multi-session LiDAR Mapping with Wasserstein-based Keyframe Selection,” 2024, doi: 10.48550/arXiv.2406.02096.
- [18] M. Menezes *et al.*, “A Multisession SLAM Approach for RatSLAM,” *J Intell Robot Syst*, vol. 108, no. 4, p. 61, Aug. 2023, doi: 10.1007/s10846-023-01816-3.
- [19] M. Cummins and P. Newman, “FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance,” *Int J Rob Res*, vol. 27, no. 6, pp. 647–665, Jun. 2008, doi: 10.1177/0278364908090961.
- [20] M. J. Milford and G. F. Wyeth, “SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights,” *Proc IEEE Int Conf Robot Autom*, pp. 1643–1649, 2012, doi: 10.1109/ICRA.2012.6224623.
- [21] X. Hu *et al.*, “PALoc: Advancing SLAM Benchmarking With Prior-Assisted 6-DoF Trajectory Generation and Uncertainty Estimation,” *IEEE/ASME Transactions on Mechatronics*, vol. 29, no. 6, pp. 4297–4308, Dec. 2024, doi: 10.1109/TMECH.2024.3362902.
- [22] H. Matsuki, R. Murai, P. H. J. Kelly, and A. J. Davison, “Gaussian Splatting SLAM,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2024, pp. 18039–18048. doi: 10.48550/arXiv.2312.06741.
- [23] Z. Xing, X. Zhu, and D. Dong, “DE-SLAM: SLAM for highly dynamic environment,” *J Field Robot*, no. May 2021, pp. 1–15, Feb. 2022, doi: 10.1002/rob.22062.
- [24] A. Li, J. Wang, M. Xu, and Z. Chen, “DP-SLAM: A visual SLAM with moving probability towards dynamic environments,” *Inf Sci (N Y)*, vol. 556, pp. 128–142, May 2021, doi: 10.1016/j.ins.2020.12.019.
- [25] Z. Xing, X. Zhu, and D. Dong, “DE-SLAM: SLAM for highly dynamic environment,” *J Field Robot*, vol. 39, no. 5, pp. 528–542, Aug. 2022, doi: 10.1002/rob.22062.
- [26] “ethz-asl/kalibr: The Kalibr visual-inertial calibration toolbox.” Accessed: Jun. 21, 2025. [Online]. Available: <https://github.com/ethz-asl/kalibr>

- [27] K. Kazi, A. N. Kalhor, F. Memon, A. R. Memon, and M. Iqbal, "Beyond Handcrafted Features: A Deep Learning Framework for Optical Flow and SLAM," *J Imaging*, vol. 11, no. 5, p. 155, May 2025, doi: 10.3390/jimaging11050155.
- [28] M. Iqbal, A. R. Memon, and D. J. Almakhlles, "Accelerating Resource-Constrained Swarm Robotics With Cone-Based Loop Closure and 6G Communication," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2025, doi: 10.1109/TITS.2025.3526819.
- [29] H. Sun, P. Wang, C. Ni, and J. ming Li, "Loop closure detection based on image semantic feature and bag-of-words," *Multimed Tools Appl*, 2022, doi: 10.1007/s11042-022-13353-6.
- [30] A. R. Memon, H. Wang, and A. Hussain, "Loop closure detection using supervised and unsupervised deep neural networks for monocular SLAM systems," *Rob Auton Syst*, vol. 126, p. 103470, Apr. 2020, doi: 10.1016/j.robot.2020.103470.
- [31] A. R. Memon, Z. Liu, and H. Wang, "Viewpoint-Invariant Loop Closure Detection Using Step-Wise Learning With Controlling Embeddings of Landmarks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, pp. 20148–20159, Nov. 2022, doi: 10.1109/TITS.2022.3173681.