2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

GR-Fusion: Multi-sensor Fusion SLAM for Ground Robots with High Robustness and Low Drift

Ting Wang†, Yun Su†,*, Shiliang Shao, Chen Yao, Zhidong Wang

***Abstract*—This paper presents a tightly coupled pipeline, which efficiently fuses measurements of LiDAR, camera, IMU, encoder, and GNSS to estimate the robot state and build a map even in challenging situations. The depth of visual features is extracted by projecting the LiDAR point cloud and ground plane into image. We select the tracked high-quality visual features and LiDAR features and tightly coupled the pre-integrated values of the IMU and the encoder to optimize the state increment of a robot. We use the estimated relative pose to re-evaluate the matching distance between features in the local window and remove dynamic objects and outliers. In the mapping node, we use refined features and tightly coupled the GNSS measurements, increment factors, and local ground constraints to further refine the robot's global state by aligning LiDAR features with the global map. Furthermore, the method can detect sensor degradation and automatically reconfigure the optimization process. Based on a six-wheeled ground robot, we perform extensive experiments in both indoor and outdoor environments and demonstrated that the proposed GR-Fusion outperforms state-of-the-art SLAM methods in terms of accuracy and robustness.**

I. Introduction

Localization and mapping are fundamental and essential for ground robots to perform navigation in unknown environments. The LiDAR, the camera, IMU, the encoder and GNSS are among the most popular sensor choices for simultaneous localization and mapping (SLAM), and numerous excellent methods have achieved satisfactory performance [1]-[5]. The camera can obtain extensive appearance information regarding the environment but without depth. Although the IMU can be integrated to improve its performance, the depth achieved through triangulation calculation is not accurate enough in large-scale outdoor environments because of the small baseline. Additionally, the camera is susceptible to light, and it can fail in dark and weak texture environments. LiDAR can obtain accurate 3D point cloud of the environment and is resistant to light. Therefore, SLAM based on LiDAR is superior to vision-based methods in terms of both accuracy and robustness. However, in large-scale outdoor environments, the point cloud obtained using LiDAR is sparse. It can also degrade in corridors, tunnels, or open environments. As for the IMU, the noise and bias results in the IMU integral drift, especially for low-cost IMUs. The robot encoder can measure the movement distance of a robot. However, it is susceptible to uneven terrain [6] [7], and large errors occur when the robot wheels slip. The global navigation satellite system (GNSS) can output the absolute position. However, it depends heavily on the strength of satellite signals. In urban environments, it can be blocked by buildings, causing significant errors or even failures. Each modal sensor has its advantages and disadvantages and can only play an active role in the appropriate scenario. Therefore, many multi-sensor fusion methods, such as camera and IMU fusion (VIO) [8], LiDAR, IMU, and GNSS fusion [9], have been proposed to improve the performance of SLAM. Therefore, in this paper, we propose ground robot (GR)-Fusion to handle various challenging scenarios through the multi-modal fusion of multi-sensors. Even in the event of sensor degradation or failure, the robot can continue to perform state estimation.

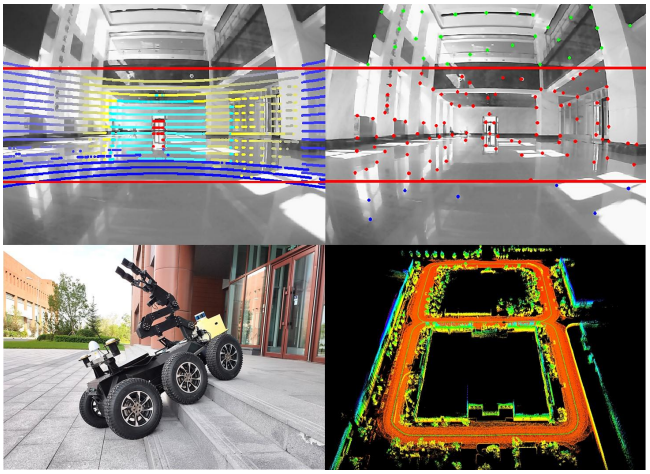


Figure 1. Ground robot with GR-Fusion. In the upper right corner of the figure, the red points represent tracked features and can extract depth from LiDAR point cloud. The blue dots indicate features located on the ground.

Based on our previous work regarding GR-SLAM [10], a method fusing stereo camera, IMU, and encoder to estimate the robot state, GR-Fusion further tightly couples LiDAR and GNSS to improve the performance of SLAM and cope with various challenging scenarios. To our best knowledge, this is also the first method that tightly couples LiDAR, camera, encoder, IMU, and GNSS to estimate robot state. We use LiDAR point cloud and ground plane parameters to extract the depth of visual features and to enhance these features. We select the tracked high-quality visual features and LiDAR features and tightly coupled local sensors and ground constraints to optimize the local state of robot. In the mapping

† T. Wang and Y. Su contributed equally to the paper.

*Corresponding author, e-mail: robosu12@gmail.com.

T. Wang, Y. Su, C. Yao, and S. L. Shao are with the State Key Laboratory of Robotics, Shenyang Institute of Automation, Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang, China. (e-mail: {wangting, cyao, shaoshiliang}@sia.cn).

Y. Su is also with the University of Chinese Academy of Sciences, Beijing, China and Guangzhou Shiyuan Electronic Technology Company Limited, Guangzhou, china.

Z. D. Wang is with the Department of Advanced Robotics, Chiba Institute of Technology, Chiba, Japan. (e-mail: zhidong.wang@it-chiba.ac.jp).

This work was supported by National Natural Science Foundation of China (U20A20201).

node, align the LiDAR features with the global map, and integrate local constraints and GNSS constraints to further optimize global state. Additionally, this method can detect sensor degradation and reconfigure the optimization process. The main contributions of this study are summarized as follows:

- We propose a framework that can fuse multi-modal measurements from LiDAR, the camera, the IMU, the encoder, GNSS and local ground constraints for robot state estimation.
- The LiDAR point cloud and ground plane is projected into the image to extract the depth of visual features. The method can detect the degradation of each sensor and reconfigure the optimization process to cope with various challenging scenarios.
- Extensive experiments were performed on real ground robots, showing that GR-Fusion has high robustness and low drift.

II. Related Work

Many existing methods have improved the accuracy and robustness of visual SLAM by fusing the camera and the IMU [2] [11] [12]. However, LiDAR can obtain accurate 3D point cloud of the environment and is resistant to light. Therefore, SLAM based on LiDAR is superior to that based on vision. The well-known LiDAR odometry and mapping (LOAM) [3] algorithm can extract corner and plane features and estimate the odometer and mapping in different threads. The LeGO-LOAM [13], which is lighter and more efficient than LOAM, can segment and filter the point cloud while ensuring the same performance. [14] proposed LIOM, which further improves the performance by tightly coupling LiDAR and IMU pre-integration, however, the efficiency was low. LIO_SAM [9] applies a sliding window-based method to match the new keyframe with the local map instead of the global map, which further improves efficiency. It can fuse GPS for global position optimization and has loop detection and optimization.

Zhang and Singh [4] [5] proposed a processing method for multi-modal measurements, which can use LiDAR, the camera, and the IMU for motion estimation and mapping through multi-layer optimization. Loosely coupled VIO is used for high-frequency motion estimation, after which the LiDAR point cloud is matched with the global map to optimize the estimation result at low frequency. Visual lidar odometry and mapping (V-LOAM) is a flexible system that can filter out failed modules to deal with sensor degradation. Shin, Park, and Kim [15] proposed a direct visual SLAM method that combines a monocular camera with sparse depth measurement from LiDAR. Shao et al. [16] proposed method that can incorporate tightly-coupled stereo visual inertial odometry with LiDAR mapping and LiDAR enhanced visual loop closure. This method is similar to V-LOAM. Zuo et al. [17] proposed LIC-Fusion, which effectively combines IMU measurement, sparse visual features, and two different LiDAR features within the multi-state constraint Kalman filter (MSCKF) framework, which can achieve efficient motion estimation and has online space and time calibration. LIC-Fusion 2.0 [18] further improves the algorithm performance by tracking the planar features in the 3D environment in a sliding window. However, the degradation of the sensor has not been considered.

GR-Fusion can use point cloud and ground plane parameters to extract the depth of visual features and improve the quality of visual information. Simultaneously, the measurements of the encoder and the IMU are tightly coupled to estimate more precise motion increments. GNSS measurements, the global map and local ground constraints are fused to optimize the global state of the robot. Additionally, GR-Fusion actively detects the quality of each sensor to deal with its failure in challenging environments.

III. The Proposed GR-Fusion

A. System overview

The pipeline of the proposed method is shown in Figure 2 and contains four nodes. The measurement frequency for each sensor is 100 Hz for the encoder, 100 Hz for the IMU, 30 Hz for the camera, 10 Hz for LiDAR, and 5 Hz for the GNSS. The measurement preprocessing node first performs data caching and synchronization. The measurements of the encoder and the IMU are then pre-integrated according to the time stamp to remove the motion distortion of the LiDAR point cloud. According to the ground measurement model proposed in our previous study, ground points are extracted, and ground plane parameters are fitted. The calculated local ground and LiDAR point cloud is then projected into the image to calculate the depth of visual features. The odometer node estimates the motion increment of the robot in a sliding window by tightly coupling multi-modal local constraint factors. The mapping node aligns the LiDAR features with the global map and tightly couples local constraints and GNSS constraints to optimize the global state of a robot. The loop-closing node receives pose and features from GR-mapping and performs loop detection and optimization to eliminate drift.

B. Measurement pre-processing

1) *Odometer and IMU pre-integration*

We used the odometer increment model on manifold proposed in [10], which can fuse measurements of the encoder and the IMU to calculate 3D position increments of the robot on complex terrain (e.g., slopes, stairs, and lawns), as shown in Figure 3.

Next, the measurements of the encoder and the IMU were pre-integrated according to the timestamp of LiDAR and the image to calculate the pose increment between frames. Under normal circumstances, the position increment is provided by the encoder and the posture increment by the IMU. If the wheels of the robot are detected to be slipping, the full pose increment is provided by the IMU.

2) LiDAR and camera measurements pre-processing

First, the motion distortion of the point cloud is removed based on the motion increment between frames. To remove noisy and unstable points, we performed point cloud segmentation following the approach presented by [13]. The corner and plane features were extracted according to the roughness. Because the robot is always running on the ground, we propose using local ground constraints to optimize the robot's pose. We used the ground measurement model proposed in the previous study to extract ground points, after which we fit ground plane parameters to calculate currently

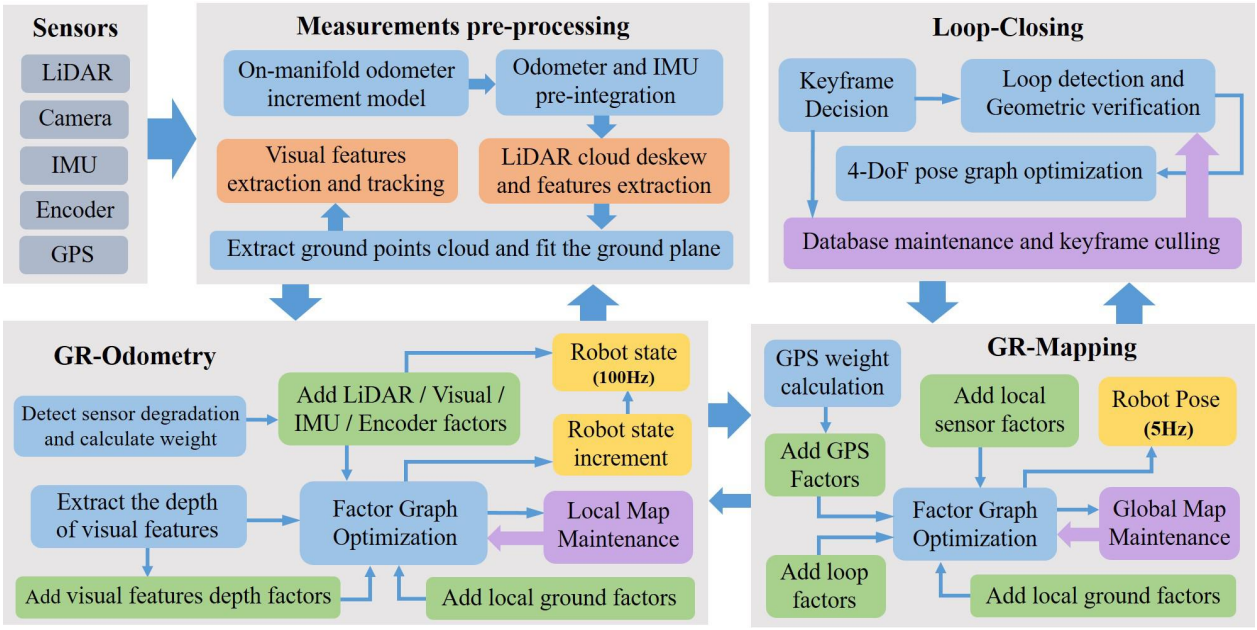


Figure 2. Pipeline of proposed GR-Fusion.

visible ground area.

For each image from the stereo camera, corner features were detected. A uniform feature distribution was achieved by setting a minimum separation of pixels between two neighboring features. These features were tracked using the KLT optical flow algorithm. We also used the KLT tracker to match features between the right and left images.

In a large-scale environment, because the baseline is small, the depth obtained using the triangulation of visual features has a significant error. Therefore, we project LiDAR point cloud into the image to extract a more robust depth for visual features, through the method introduced by [20]. The local ground plane was projected into the image to segment the visual features on the ground. The camera projection model was used to calculate the depth of ground features. The experiments revealed that the visual features located on the ground move fast, and the tracking time is short, and it is often impossible to obtain accurate depth measurements through triangulation. Therefore, the ground parameters fitted using LiDAR can be used to directly calculate the depth of ground features. Additionally, it implicitly provides co-planar constraints for ground visual features.

C. GR-odometry

The odometer node was designed to select a small number of high-quality features to quickly estimate the robot's motion increment in a local window. Meanwhile, the optimization process was reconfigured according to the degradation of the sensor. Additionally, before the current LiDAR frame was sent to the mapping thread, the features of the current frame were refined using the results of the odometer, and the features located on dynamic targets and unstable feature points are eliminated.

1) State vector

The proposed method takes the LiDAR frame as the optimization node and selects the nearest image as the visual constraints. The state vector is expressed as follows:

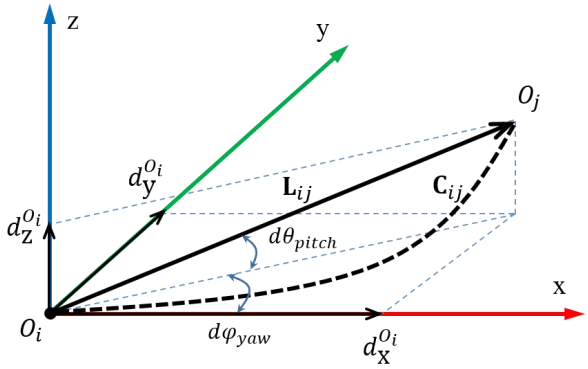


Figure 3. Illustration of odometer increment model on manifold ($\mathbf{C}_{ij}$ is the robot trajectory from timestep i to j and $\mathbf{L}_{ij}$ is the motion vector)[10].

$$\boldsymbol{\chi}_k = \left[\mathbf{X}_I^{\mathrm{T}}, \mathbf{X}_L^{\mathrm{T}}, \mathbf{X}_C^{\mathrm{T}}, \mathbf{T}_{OI}, \mathbf{T}_{IL}\right]^T. \tag{1}$$

$\mathbf{X}_I, \mathbf{X}_L, \mathbf{X}_C$ represent the status of the IMU, LiDAR, and the camera at the current node. $\mathbf{T}_{OI}$ represents the external parameter between the robot and the IMU, and $\mathbf{T}_{IL}$ represents the external parameter between the IMU and LiDAR. The intrinsic parameters of camera and the extrinsic parameters between the IMU and the camera were calibrated offline. The robot odometer's intrinsic parameters of wheel radius and distance between the two wheels were calibrated using the method presented in our previous study [10]. Notably, the method still updates the wheel radius based on the original encoder measurement and the estimated robot motion to cope with the change in the robot's wheel radius.

Additionally, we estimated the time offset t_{IL} between IMU and LiDAR as well as t_{IC} between the IMU and camera.

$$\begin{aligned} t_I &= t_L + t_{IL} \\ t_I &= t_C + t_{IC} \end{aligned} \tag{2}$$

We used the IMU time as the benchmark for converting the measurement time of LiDAR and the camera to the IMU time. It was then optimized together with the IMU measurements to estimate the time offset between sensors.

2) Sensor factors

We maintained a sliding window in the odometer phase, as shown in Figure 4. There are n + 1 nodes, and the optimization window contains the three latest consecutive nodes. Other nodes are selected as keyframes according to the movement distance and keep them in the sliding window to provide constraints for the optimization window. LiDAR features in the sliding window are used to maintain a local point cloud map and the newest frame is projected into the image. For each visual feature, the adjacent block of point cloud is extracted for depth estimation [20]. For the feature points whose depth has been obtained from ground parameters, the depth value remains unchanged. Additionally, when extracting features from a new image, in the area covered by the point cloud and the lower area, the visual feature spacing was set as half of the other areas. This strategy enables the extraction of more features from the image area covered by the point cloud, and the depth can be directly estimated and is more robust.

The more times a feature point is tracked, the higher quality it has. Therefore, we added the points that are tracked more times than the threshold to the optimization. Next, the following re-projection error was constructed according to the tracking result.

$$\begin{aligned}\mathbf{r}_C(\mathbf{z}_l^{C_j},\boldsymbol{\chi}) &= \mathbf{z}_l^{C_j} - h_l^{C_j}(\mathbf{p}_i,\mathbf{R}_i,\mathbf{p}_j,\mathbf{R}_j,\lambda_l) \\ &= \begin{bmatrix} u_l^{C_j} \\ v_l^{C_j} \end{bmatrix} - \pi_c(T_{IC}^{-1}T_j^{-1}T_iT_{IC}\pi_c^{-1}(\lambda_l,\begin{bmatrix} u_l^{C_i} \\ v_l^{C_i} \end{bmatrix}))\end{aligned} \tag{3}$$

where π_c and π_c^{-1} represent the projection and back projection functions of the camera's model, respectively. λ_l represents the inverse depth of the feature. When estimating the depth of each visual feature, we used the extracted depth as the initial value and construct the value as prior factor to add to the optimization.

We matched the corner points and plane features extracted from the LiDAR point cloud with the current local map and used the distances from point to line and point to surface to construct LiDAR constraint factors. Further, the measurements of the encoder and the IMU were pre-integrated to construct an incremental constraint factor.

3) Local ground constraints

We considered the local ground to be unchanged between two consecutive frames when the robot moves on the ground and the z-axis displacement in the local ground to be close to zero if the flat ground is fitted. To ensure the robustness of our formulation, we considered factors such as the normal vector, ground roughness, and ground continuity to determine whether to add the ground constraints.

- The angle between the normal vector of the ground plane and the z-axis of the robot must be less than the set threshold.

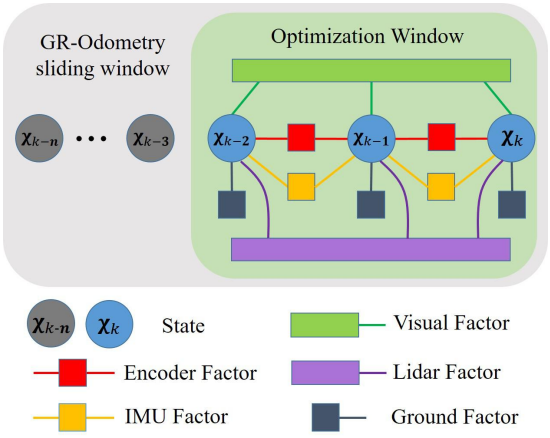


Figure 4. Illustration of local window for GR-odometry. The size of the optimization window size is 3, and the sliding window is $n + 1$.

- The sum of the squares of the distance between the segmented ground point and the fitted ground plane must be less than the set threshold.
- The distance between the farthest point and the closest point of the ground must be less than the set threshold.

4) Sensor degradation detection

The camera cannot extract stable features for tracking in a weak texture environment or a dark environment. When there are large numbers of dynamic targets in the scene, a large tracking error also occurs. First, we counted the tracking times of each feature in the current frame and reduce the weight of visual features if it is less than the set threshold. If the sum of all feature tracking times was less than the set threshold, it was considered to be unstable. At this time, visual constraints are not added to the optimization.

LiDAR degrades in highly repetitive scenes, such as corridors or open outdoor environments. Therefore, we distinguished between the depth of the latest frame point cloud and that of the first frame of the point cloud outside the optimization window and calculated the average depth difference. Next, histogram statistics on the depth difference of all points were obtained. If the number of points where the depth difference is greater than the threshold is small, LiDAR is considered degraded. At this time, the LiDAR constraints are not added to the optimization, and the size of the optimization window is adjusted. The system degenerates to GR-SLAM and tightly couples the camera, the encoder, and the IMU for state estimation. However, the local point cloud map is still maintained, and the mapping node still runs to maintain the global map.

Because the IMU is a built-in sensor and is immune to the external environment, the estimation result within a short time is relatively accurate. The slipping of robot wheels inevitably increases the pre-integration value of the encoder. Therefore, we calculated the ratio of the pre-integration of the encoder to that of the IMU. If it is greater than the set threshold, the wheel is considered to be slipping, and the encoder constraint is not added to the optimization. Additionally, the experimental results revealed that the robot easily slips when turning and that the encoder has higher accuracy when it moves in a straight line. Therefore, we also used the rotation of the robot to adjust the optimization weight of the encoder.

5) Local factor graph optimization

Bundle adjustment was used to minimize the residuals of all the factors and obtain the maximum a posteriori estimation of the robot state as follows:

$$\chi^* = \arg\min\left\{\sum_{(l,i)\in L}\rho\left(\left\|\mathbf{r}_L(\mathbf{z}_l^{L_i},\chi)\right\|^2_{\Sigma_l^{L_i}}\right)\right.$$
$$\sum_{(l,j)\in C}\rho\left(\left\|\mathbf{r}_C(\mathbf{z}_l^{C_j},\chi)\right\|^2_{\Sigma_l^{C_j}}\right)+\left\|\mathbf{r}_G(\mathbf{z}^G,\chi)\right\|^2_{\Sigma_G} \quad (4)$$
$$\left.+\left\|\mathbf{r}_I(\mathbf{z}^I_{I_k I_{k+1}},\chi)\right\|^2_{\Sigma^{I_k}_{I_{k+1}}}+\left\|\mathbf{r}_O(\mathbf{z}^O_{I_k I_{k+1}},\chi)\right\|^2_{\Sigma^{O_k}_{O_{k+1}}}\right\}$$

where $\mathbf{r}_G(\mathbf{z}^G,\chi)$ represents local ground constraints. Notably, what is optimized here is the state increment of the node in the optimization window shown in Figure 4. Only after the node slides out of the optimization window will it be sent to the mapping node for global optimization.

Notably, we used the results of the GR-odometry to remove dynamic objects and unstable features. First, the current LiDAR features were projected to the start frame of the local window to calculate the matching distance. For a static environment, the projected LiDAR points are coincident, or the matching distance is small. For moving objects, the matching distance of projected points must be large. Therefore, the LiDAR features were ranked according to the matching distance, and the points whose matching distance exceeds the set threshold are judged as dynamic objects and be eliminated. The maximum removal ratio is set to 10%. This strategy uses the motion characteristics of the dynamic objects, and uses the time span of the sliding window, so that the projection error of the dynamic objects can be highlighted. Finally, the refined feature points are sent to the GR-Mapping node.

D. GR-Mapping and Loop-Closing

1) GNSS weight calculation

Although we can fuse multi-modal measurements for robot state estimation, the long-term operation still produces cumulative errors. The GNSS can provide absolute position. However, the experimental results revealed that the GNSS measurement jumps when GNSS is obstructed by buildings or trees, resulting in large errors or even failure. Therefore, we evaluated GNSS quality based on the following data:

- Number of satellites in use
- Dilution of precision (DOP)
- Variance of local increment

The more satellites searched, the more accurate the GNSS measurements are. However, it is also related to the spatial geometric distribution of satellites. Therefore, we also used the DOP value output by the GNSS to evaluate the measurements error. Additionally, we calculated the variance of the local GNSS increment and the odometer increment to evaluate the GNSS measurement quality. Although odometry can produce cumulative errors, its local accuracy is relatively high. Therefore, the odometry can be used to determine whether the GNSS measurement has jumped and whether the quality of the local measurement has decreased to avoid large errors.

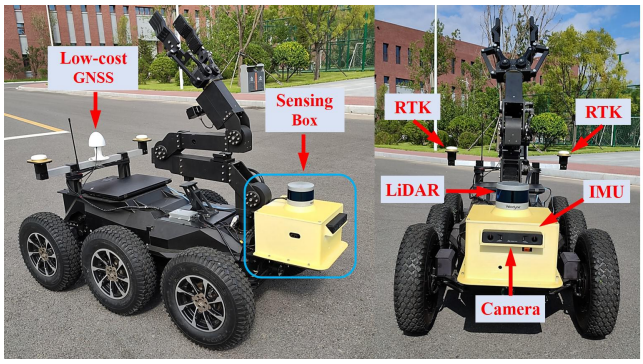


Figure 5. Experimental setup using a six-wheeled ground robot.

2) Global factor graph optimization

The constraint factors we added to the global optimization are as follows:

- Odometry increment factor
- Local ground factor
- Pitch and roll factors estimated from the IMU
- GNSS factors
- Global point cloud map factors
- Loop-Closing factors

Using the estimated results of the odometry as the initial value, we aligned the LiDAR features with the global map while coupling other local constraints and global constraints to optimize the global state of the robot. The closed-loop factor comes from the closed-loop detection node, and we used the point cloud descriptor proposed by [21] for position recognition. It can also save and reuse maps and quickly relocate to existing maps.

Notably, our method supports the low-cost configuration which only contains encoder, IMU, and GNSS and perform efficient state estimation for ground robots.

IV. EXPERIMENTS

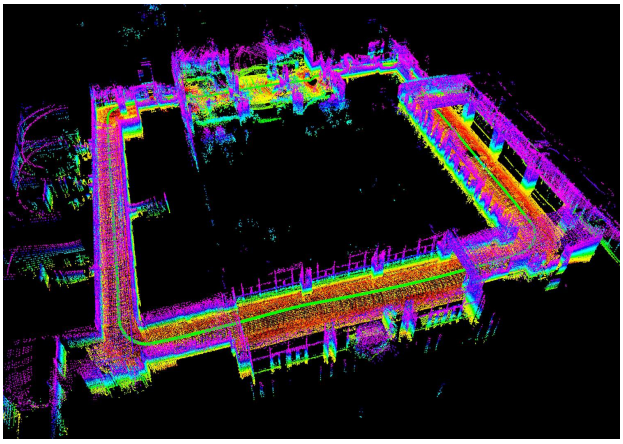
We evaluated the performance of the proposed method in various environments based on a six-wheeled ground robot, as shown in Figure 5. We built a sensing box， including a VLP-16 LiDAR, a stereo camera, a NUC8I7HVK onboard computer, a wireless communication system, and a power supply system. We use a low-cost IMU built into the stereo camera. Low-cost GNSS is installed behind the robot. The onboard computer communicates with the robot controller through a serial port to acquire the raw encoder angle data and control data from the robot controller and send velocity commands to the robot controller at a frequency of 100 Hz.

We collected multiple datasets in different environments. These datasets are referred to as *Indoor*, *underground*, *Open Lawn*, and *Campus*, respectively. And for the robotics community, we open source the GroundRobotDataset [1], and we will continue to update the dataset to make the content as complete as possible. It should be noted that for fairness, we used the open-loop mode of the proposed algorithm for evaluation. The details of used dataset are shown in Table I.

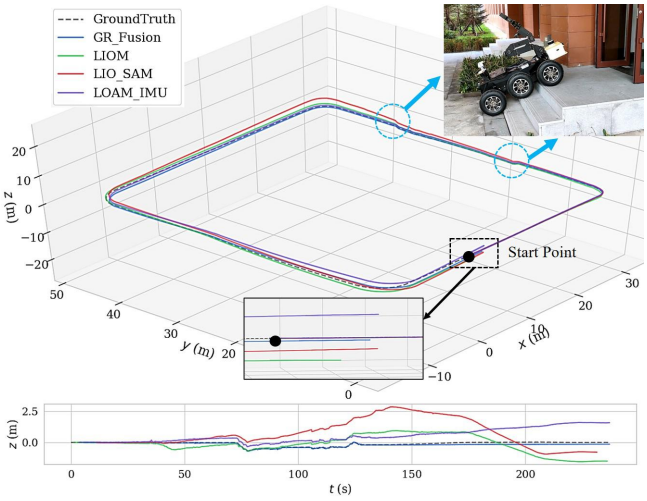
[1]GroundRobotDataset: *https://drive.google.com/drive/folders/110Hko3zPcDmY0_bnZdXxJXJKe6wr3t10?usp=sharing.*

TABLE I. DATASET DETAILS

Dataset	Trajectory length (m)	Characteristics
Indoor	196	Flat ground, indoors, stairs
Underground	408	Uneven road, long slope
Open Lawn	337	Lawns, open environment, steps
Campus	863	Uneven road, dynamic objects



(a) Mapping result by GR_Fusion



(b) Trajectory comparison

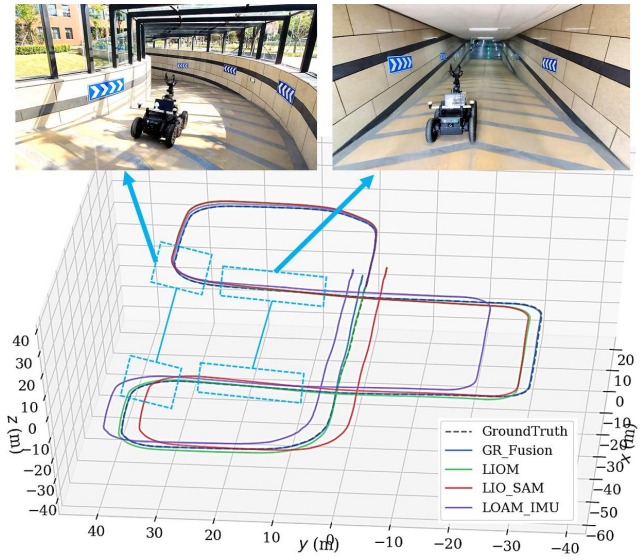
Figure 6. Results of various methods on the *Indoor* dataset.

A. Indoor Dataset

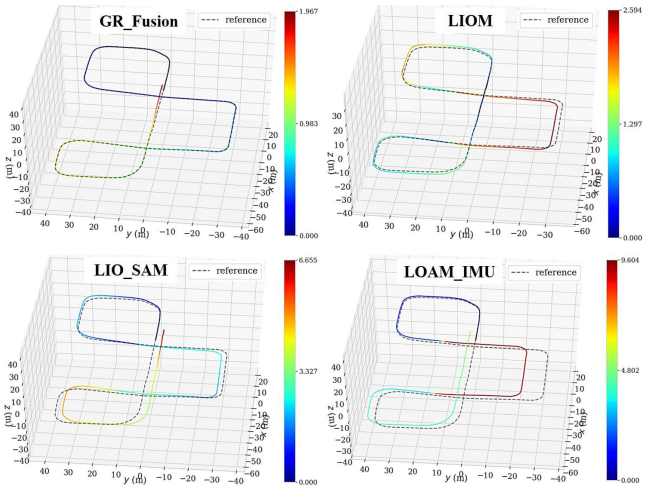
This dataset contains indoor corridor environments, as well as stairs. The start and end points of the trajectory coincide, and we use the results of offline batch optimization and closed-loop optimization as the ground truth. The results are shown in Figure 6. The robot goes up and down the stairs at the place marked by the circle in Figure 6(b). Due to the drastic change of the LiDAR viewing, both LIO_SAM and LOAM_IMU produced large errors. LIOM [14] tightly couples with IMU, and the error is small. GR_Fusion tightly couples the camera, IMU, encoder and LiDAR on the front end to quickly estimate state increment of the robot, which can cope with the situation of drastic changes in the viewing angle. The curve at the bottom of Figure 6(b) is the displacement of algorithms on the z-axis. Because GR_Fusion considers local ground constraints in optimization, it has higher accuracy.

TABLE II. ROOT MEAN SQUARE ERROR OF TRAJECTORY (METERS)

Dataset	GR_Fusion	LIOM	LIO_SAM	LOAM_IMU
Indoor	**0.56**	0.77	1.04	0.89
Underground	**0.87**	1.39	3.17	5.76
Open Lawn	**1.54**	9.36	2.94	12.11
Campus	**1.62**	3.45	2.28	4.71



(a) Trajectory comparison



(b) Trajectory error comparison

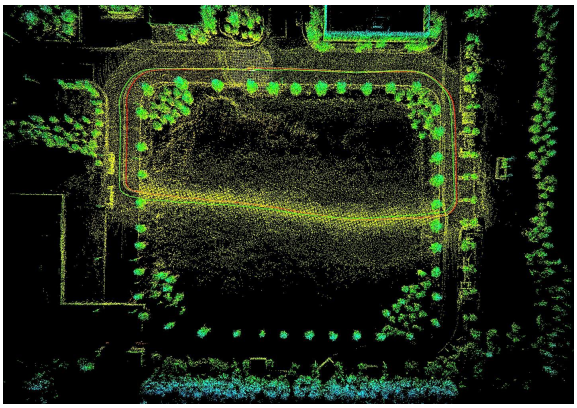
Figure 7. Results of various methods on the *Underground* dataset.

The absolute translational error of all methods are shown in Table II. LOAM_IMU is the optimized LOAM. We added local attitude increment and global gravity constraint of the IMU to improve the accuracy of its estimation.

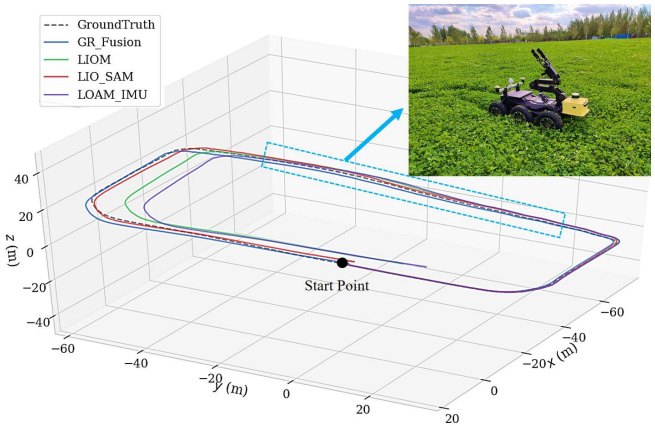
B. Underground Dataset

In this dataset, the robot starts from the campus, passes through the underground garage, and then returns to the ground campus. The access of the underground garage has a rotating channel and a straight channel, as shown in Figure 7(a). The environment of these channel is highly repetitive, causing degradation of LiDAR.

The results of each algorithm are shown in Figure 7(b). It

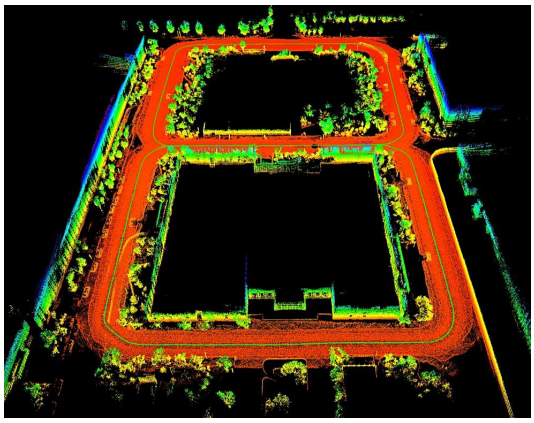


(a) Environment of *Open Lawn* dataset

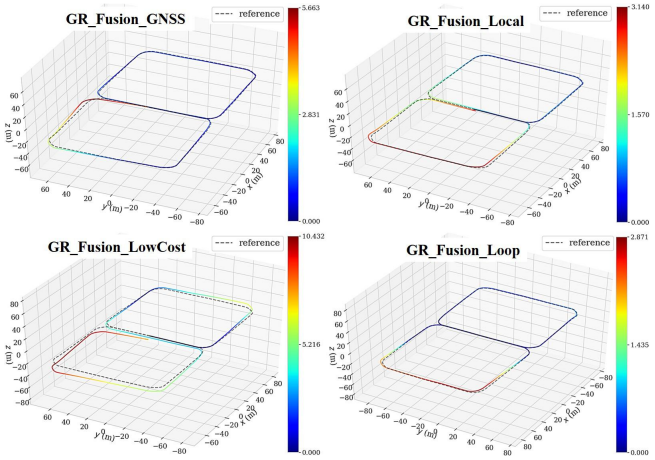


(b) Trajectory comparison

Figure 8. Results of various methods on the *Open Lawn* dataset.



(a) Mapping result by GR_Fusion after closed loop fusion



(b) Trajectory error comparison

Figure 9. Results of various methods on the *Campus* dataset.

can be seen that LIOM, LIO_SAM and LOAM_IMU all have obvious errors in the underground channel because of the degradation of LiDAR. GR_Fusion can still rely on vision, IMU and encoder incremental models for continuous state estimation. When GR_Fusion detects LiDAR degradation, it will reduce the optimization weight of LiDAR. And on the basis of the existing state, the size of the local sliding window will be gradually increased, and more data will be included for optimization. While ensuring that the pose does not jump, the accuracy of the state estimation is guaranteed.

C. *Open Lawn Dataset*

In this test, we evaluate the performance of the proposed method in an outdoor large-scale open environment. In the *Open Lawn* data set, the robot traversed a very open lawn and could only see some sparse trees in the distance, as shown in Figure 8. When crossing the lawn, most of the features extracted from the LiDAR are distributed on the ground and degenerate the motion estimation. It can be seen in Figure 8(b) that LIOM and LOAM_IMU produced significant errors due to LiDAR degradation. Because LIO_SAM fuses GNSS measurement, the estimated global pose can guarantee a certain accuracy. But when we remove the GNSS measurement, the state estimated by LIO_SAM has a jump, and the resulting error is larger than LOAM_IMU. This is because LIO_SAM uses fewer features than LOAM_IMU in its estimation. In addition, we found that in the outdoor environment, the features extracted from LiDAR are relatively sparse, so the performance of other algorithms evaluated has decreased, compared to the indoor environment. GR_Fusion combines LiDAR, camera, IMU and encoder measurements to ensure the same accuracy regardless of indoor or outdoor environments. And by fusing GNSS measurements, GR_Fusion can output global drift-free estimation results.

D. *Campus Dataset*

In this test, we evaluated the performance of GR_Fusion in different configurations. GR_Fusion_Local uses local sensors, including LiDAR, camera, IMU and encoder. GR_Fusion_GNSS further fuses GNSS measurements. GR_Fusion_LowCost only uses low-cost sensor configuration, including GNSS, IMU and encoder. GR_Fusion_Loop has closed loop optimization. It should be noted that closed loop optimization was not used in the previous experiments.

The results are shown in Figure 9. GR_Fusion_Loop has the highest accuracy and can eliminate accumulated errors. The accuracy of GR_Fusion_Local is higher than that of GR_Fusion_GNSS. This is because the quality of the GNSS measurement in a section of the trajectory is reduced, which leads to errors in the global optimization. However, GR_Fusion_GNSS has higher accuracy when running for a long time. GR_Fusion_LowCost only uses IMU and encoder incremental model for local estimation, and uses GNSS for global state optimization. Its accuracy is easily affected by GNSS measurements, but it can still provide acceptable results and is very useful in low-cost robots.

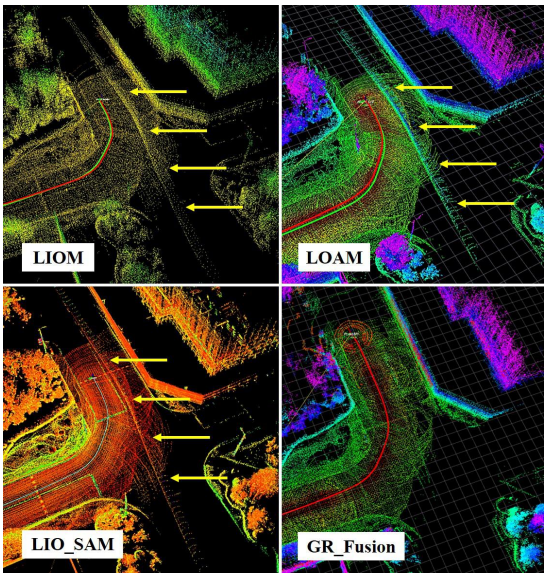


Figure 10. Mapping results when a vehicle passes by.

E. Dynamic objects removal

We also tested the mapping performance of GR_Fusion when dynamic objects appear in the environment. Figure 10 shows the mapping results when a vehicle passes by. It can be seen that other algorithms include all the point clouds of moving vehicles in the map. GR_Fusion can effectively eliminate dynamic objects and build a more reasonable map of static environments.

In addition, we have made statistics on the real-time performance of algorithms, as shown in Table Ⅲ. It can be seen that although GR_Fusion fuses more sensors, it can provide efficient and accurate estimation results through careful processing and selection of information, and efficient strategies in engineering implementation.

TABLE III. MEAN COMPUTATION TIME

Algorithm	Computation time (ms)	
	Odometry	Mapping
GR_Fusion	86.1	205.3
LIOM	325.2	537.1
LIO_SAM	---	97.8
LOAM_IMU	23.9	356.2

V. CONCLUSIONS AND FUTURE WORK

We propose a multi-modal sensor fusion method that can robustly and accurately estimate the state of the robot. Through careful selection and refinement of the data of each sensor, while improving the accuracy, it can also eliminate dynamic targets and unstable features. It can detect sensor degradation in real time and can be flexibly configured for multiple working modes. At present, visual information is only used in the front end to estimate the local state of the robot. In the future, we will combine visual and LiDAR to create landmarks with global descriptors to further optimize the global state of the robot.

REFERENCES

[1] R. Mur-Artal and J. D. Tard´os, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” IEEE Trans. Robot., 2017.

[2] T. Qin, P. Li, and S. Shen, “Vins-mono: A robust and versatile monocular visual-inertial state estimator,” IEEE Trans. Robot., vol. 34, no. 4, pp. 1004–1020, Aug 2018.

[3] Zhang J, Singh S, “Low-drift and real-time lidar odometry and mapping,” Autonomous Robots, 2017, 41(2): 401-416.

[4] J. Zhang and S. Singh, “Visual-lidar odometry and mapping: Low-drift, robust, and fast,” in Robotics and Automation (ICRA), 2015 IEEE International Conference on. IEEE, 2015, pp. 2174–2181.

[5] J. Zhang and S. Singh, “Laser–visual–inertial odometry and mapping with high robustness and low drift,” Journal of Field Robotics, vol. 35, no. 8, pp. 1242–1264, 2018.

[6] Zheng, Fan, and Yun-Hui Liu, “Visual-Odometric Localization and Mapping for Ground Vehicles Using SE(2)-XYZ Constraints,” International Conference on Robotics and Automation (ICRA), 2019.

[7] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis, “Vins on wheels,” International Conference on Robotics and Automation (ICRA), IEEE, 2017, pp. 5155– 5162.

[8] Qin T, Pan J, Cao S, et al., “A general optimization-based framework for local odometry estimation with multiple sensors,” arXiv preprint arXiv:1901.03638 (2019).

[9] T. Shan, B. Englot, et al., “LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping,” 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[10] Yun Su, Ting Wang, Chen Yao, Shiliang Shao and Zhidong Wang, “GR-SLAM: Vision-Based Sensor Fusion SLAM for Ground Robots on complex terrain,” IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[11] Carlos Campos, Richard Elvira, et al., “ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM,” arXiv:2007.11898.

[12] S. Leutenegger, et al., “Keyframe-based visual–inertial odometry using nonlinear optimization,” Int. J. Robot. Res., vol. 34, no. 3, pp. 314– 334, 2015.

[13] Shan T, Englot B, “Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain,” 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018: 4758-4765.

[14] Ye H, Chen Y, Liu M, “Tightly coupled 3d lidar inertial odometry and mapping,” 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019: 3144-3150.

[15] Y.-S. Shin, Y. S. Park, and A. Kim, “Direct visual slam using sparse depth for camera-lidar system,” in 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–8, IEEE, 2018.

[16] W. Shao, S. Vijayarangan, C. Li, et al., “Stereo visual inertial lidar simultaneous localization and mapping,” 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019.

[17] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, “Lic-fusion: Lidarinertial-camera odometry,” in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, Nov. 2019, pp. 5848–5854.

[18] X. Zuo, Y. Liu, P. Geneva et al., “LIC-Fusion 2.0: LiDAR Inertial Camera Odometry with Sliding-Window Plane-Feature Tracking,” 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[19] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “On-manifold preintegration for real-time visual–inertial odometry,” IEEE Trans. Robot., vol. 33, no. 1, pp. 1–21, 2017.

[20] J. Graeter, A. Wilczynski, and M. Lauer, “Limo: Lidar-monocular visual odometry,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 7872–7879.

[21] H. Wang, C. Wang and LH. Xie, “Intensity Scan Context: Coding Intensity and Geometry Relations for Loop Closure Detection,” arXiv preprint arXiv:2003.05656 (2020).