IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

LLIO: Lidar-Kinematic-Inertial Odometry with Ground Contact Constraints for Legged Robots

Chengjie Gu, Zhongqu Xie, Beichen Xiang, Shichao Zhou, Lingkun Chen, Binbin Ci, and Yulin Wang*

***Abstract*—This letter presents a robust multi-sensor fusion framework for state estimation in legged robots (LLIO) based on an iterated extended Kalman filter. To address the limitations of IMU priori estimation, which often leads to legged robot localization errors or failures, our method integrates the contact constraints of the robot's leg kinematics with the ground. By introducing a sliding window-based ground contact constraint module, we effectively combine the contact state of the legged robot's foot with ground features, enhancing the constraints in complex environments and reduce localization drift. Additionally, factor graph optimization minimizes global cumulative drift. The proposed method has been extensively evaluated through numerous experiments and relevant public datasets. The results demonstrate that our approach significantly reduces local drift and better computational efficiency.**

***Index Terms*—SLAM, Legged Robots, Localization.**

I. Introduction

STATE estimation and localization are essential prerequisites for motion, navigation, and planning in legged robots. In environments with limited GPS signal availability or poor lighting conditions, Lidar-Inertial Odometry (LIO), which integrates high-frequency IMU measurements with LiDAR depth information, can significantly enhance the localization accuracy of the system. However, the substantial impacts and high-frequency vibrations experienced during legged robot locomotion can accelerate the degradation of both the accelerometer and LiDAR point cloud quality [1]. Such degradation can severely compromise LIO system accuracy, leading to rapid drift accumulation, particularly in systems relying on IMU propagation for prior estimation.

Many previous studies [2], [3], [4] have utilized leg kinematics to provide additional constraints for the state estimation of robots. Although the kinematic-inertial approach is cost-effective, it lacks the ability to offer external distance measurements, which can lead to significant distortion and drift in state estimation. The incorporation of exteroceptive sensors effectively combines environmental information constraints with robot kinematics, thereby reducing position drift. Many contemporary methods [5], [6], [7], [8] successfully enhance the accuracy of robot state estimation by integrating external sensors, such as LiDARs and cameras. However, the introduction of these sensors often impose substantial computational overhead that compromises real-time performance, particularly in challenging terrain conditions.

In this paper, we aim to improve the localization accuracy and computational efficiency of robots on different terrain by integrating leg kinematics and ground contact constraints into the back-end of LIO system. We propose a tightly coupled Lidar-Kinematic-Inertial Odometry framework, LLIO, which is based on ground contact at the robot's foot. To this end, we summarize the contributions of our work as follows:

- We propose a filter-based Lidar-Kinematic-Inertial odometry approach and utilize factor graphs for back-end optimization to improve localization accuracy for legged robots in complex environments.
- We formulate a ground contact constraint that operates on the converged filter state, demonstrating effective attitude drift compensation across varied terrain conditions (flat, rough and slope).
- The experimental result demonstrate that our method has lower local drift and better computational efficiency compared with other Lidar-Kinematic-Inertial methods. The dataset[1] are released for the benefit of the community.

II. Related Work

State estimation for legged robots typically relies on information from multiple sensors. These sensors primarily include the robot's proprioceptors, such as joint encoders, IMU, and contact sensors. The methods of [2], [3], [4], [9] perform state estimation using filtering techniques that incorporate data from the robot's joint encoders and IMU. Hartley et al. [10] reduced localization drift by using a factor graph strategy and considering the robot's kinematic model and its interactions with the environment. However, relying solely on proprioceptive sensors can lead to distortions in the robot's local position, resulting in global localization drift. Many methods [7], [8], [11], [12], [13], [14]that further integrate cameras are used to improve the accuracy of state estimation.

LiDAR-based state estimation has emerged as a prominent topic in the field of robotics. Zhang et al. [15] developed the

Received 20 March 2025; accepted 9 July 2025. Date of publication 23 July 2025; date of current version 31 July 2025. This article was recommended for publication by Associate Editor X. Xiong and Editor A. Kheddar upon evaluation of the reviewers' comments. This work was supported partly by the National Key Research and Development Program of China under grant 2024YFB4711100, the National Natural Science Foundation of China under grant 52305024, the Natural Science Foundation of Jiangsu Province under grant BK20230928, the China Postdoctoral Science Foundation under grant 2023M731690 and 2025T181128, and the Fundamental Research Funds for the Central Universities under grant 30923011029 and 2024301001. *(Corresponding author: Yulin Wang.)*

Chengjie Gu, Zhongqu Xie, Beichen Xiang, Shichao Zhou, Lingkun Chen, Binbin Ci, and Yulin Wang are with Department of Mechanical Engineering, Nanjing University of Science and Technology, 210094 Nanjing, China {gucj, xiezq, blacklioneer, njustsmezsc, chenlingkun, bennychan1028} @njust.edu.cn, wyl_sjtu@126.com

Digital Object Identifier (DOI): 10.1109/LRA.2025.3592136

[1]https://github.com/GuCJ-Acc/LLIO-Dataset

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

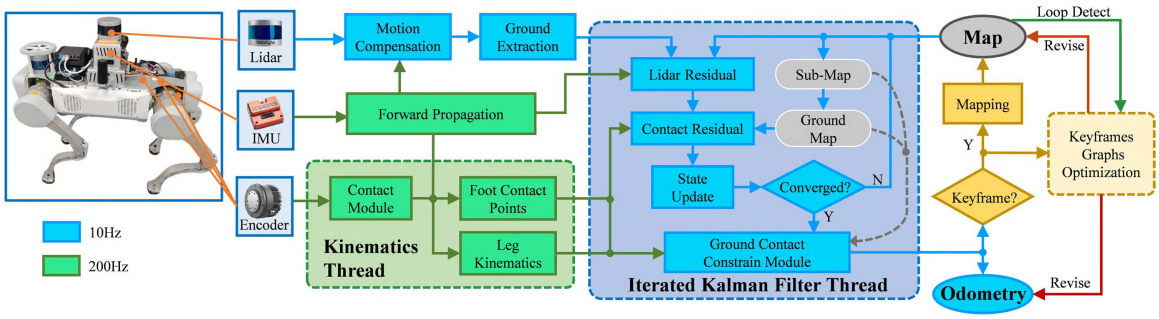

Fig. 1. System Overview of LLIO: The LLIO system, which receives input from a 3D LiDAR, IMU, and joint encoders, can be divided into three main components: a kinematics thread, an iterated Kalman filter thread, and a backend keyframes graph optimization module.

foundational LiDAR localization framework known as LOAM, which employs the Iterative Closest Point (ICP) method along with feature point matching. Building on this, Shan et al. proposed the LeGO-LOAM [16] framework, focusing on ground features, and the Lidar-Inertial Odometry framework LIO-SAM [17], which leverages factor graph optimization to improve localization accuracy. Xu et al. [18] introduced Fast-LIO2, a fast and lightweight Lidar-Inertial odometry solution based on a Kalman filter. Point-LIO [19] enhances robustness during aggressive movements by integrating IMU measurements as the posterior of the filter.

Multi-sensor fusion of proprioceptive and exteroceptive data enhance localization precision and robustness. STEP [8] introduce a preintegrated foot velocity factor combined with visual factor effectively improve localization accuracy. BIG-STEP [11] improves initialization and optimization by refining 3D features onto the estimated ground plane based on STEP. VILENS [5] achieve more accurate state estimation by coordinating the processing of leg kinematics, LiDAR, IMU, and camera data based on factor graphs. LIKO [20] and KLILO [21] integrate legged robot kinematics with LiDAR measurements to further improve state estimation accuracy. Leg-KILO [6] projects foot contact points onto the environment map to calculate terrain-relative height differences. These measurements are processed through [2] to update the foot odometry system, generating accurate pose priors for the Lidar-Inertial Odometry (LIO) pipeline. However, its computational intensity on rough and deformable terrain limits real-time performance. Unlike Leg-KILO, our approach instead computes foot-ground contact residuals and apply it to the constraints optimization in LIO back-end, significantly improving computational efficiency in challenging environments. Additionally, We have conducted comparative experiments based on the publicly available Leg-KILO dataset.

III. METHODLOGY

In this paper, we denote scalars and frame abbreviations in lowercase italics, matrices in uppercase Roman bold, and vectors in lowercase Roman bold. $\boxplus$ and $\boxminus$ are two encapsulation operators that represent a bijective mapping from a local neighborhood on $\mathcal{M}$ to its tangent space $\mathbb{R}^n$ [18]. The operator $[\mathbf{a}]^\wedge$ is used to convert a vector $\mathbf{a}$ into its corresponding skew-symmetric matrix. Besides, we also use $\mathbf{x}$, $\hat{\mathbf{x}}$, and $\bar{\mathbf{x}}$ to represent the groundtruth, prior, and updated posterior values of state respectively.

A. System Overview

The system architecture, illustrated in Fig. 1, integrates multi-sensor data from a 3D LiDAR, an IMU, and joint encoders. The proposed system comprises three core components: (1) a kinematics thread for foot-end state estimation from joint encoders, (2) an iterated Kalman filter (IKF) thread for LiDAR/contact residual minimization with ground contact constraint refinement, and (3) a keyframe-based graph optimization backend. The pipeline first compensates LiDAR motion distortion via IMU propagation while extracting ground features, then performs parallel kinematic estimation and sensor fusion, and finally optimizes the global state and mapping. And, the data from each sensor is synchronized in time through linear interpolation.

B. State Representation

We denote the world frame as G, and the IMU frame as I. We assume that the robot frame coincides with the IMU frame. The robot state $\mathbf{x}$ is defined as follows:

$$\mathbf{x} = [{}^G\mathbf{R}_I^T \;\; {}^G\mathbf{p}_I^T \;\; {}^G\mathbf{v}_I^T \;\; \mathbf{b}_\mathbf{g}^T \;\; \mathbf{b}_\mathbf{a}^T \;\; {}^G\mathbf{g}^T]^T \tag{1}$$

where ${}^G\mathbf{R}_I \in SO(3)$ is the rotation matrix, ${}^G\mathbf{p}_I \in \mathbb{R}^3$ is the position vector, ${}^G\mathbf{v}_I \in \mathbb{R}^3$ is the velocity of the robot. $\mathbf{b_g}, \mathbf{b_a} \in \mathbb{R}^3$ are the bias of the IMU gyroscope and accelerometer, and ${}^G\mathbf{g}$ is the unknown gravity vector in the world frame.

C. Forward Propagation

Taking the IMU frame (denoted as I) as the robot's frame of reference leads to the kinematics of the system. The continuous-time model of this system can be expressed as follows:

$$\begin{aligned} {}^G\dot{\mathbf{R}}_I &= {}^G\mathbf{R}_I(\boldsymbol{\omega}_m - \mathbf{b_g} - \mathbf{n_g})^\wedge, \\ {}^G\dot{\mathbf{p}}_I &= {}^G\mathbf{v}_I, \\ {}^G\dot{\mathbf{v}}_I &= {}^G\mathbf{R}_I(\mathbf{a}_m - \mathbf{b_a} - \mathbf{n_a}) + \mathbf{g}, \\ \dot{\mathbf{b}}_\mathbf{g} &= \mathbf{n_{bg}}, \;\; \dot{\mathbf{b}}_\mathbf{a} = \mathbf{n_{ba}}, \;\; {}^G\dot{\mathbf{g}} = 0 \end{aligned} \tag{2}$$

where $\mathbf{a}_m$, $\boldsymbol{\omega}_m$ are IMU measurements, $\mathbf{n_a}$ and $\mathbf{n_g}$ are the white noise of IMU measurements, $\mathbf{n_{ba}}$ and $\mathbf{n_{bg}}$ are the random walk process with Gaussian noises of IMU bais.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

The continuous kinematic model in (2) can be discretized at the IMU sampling time t using a zero-order holde method, with period Δt.

$$\mathbf{x}_{t+1} = \mathbf{x}_t \boxplus (\mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t)\Delta t) \tag{3}$$

the function $\mathbf{f}$, input $\mathbf{u}$, and noise $\mathbf{w}$ are defined as below:

$$\mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t) = \begin{bmatrix} \boldsymbol{\omega}_{m_t} - \mathbf{b}_{\mathbf{g}_t} - \mathbf{n}_{\mathbf{g}_t} \\ {}^{G}\mathbf{v}_t \\ {}^{G}\mathbf{R}_{I_t}(\mathbf{a}_{m_t} - \mathbf{b}_{\mathbf{a}_t} - \mathbf{n}_{\mathbf{a}_t}) + {}^{G}\mathbf{g}_t \\ \mathbf{n}_{\mathbf{bg}_t} \\ \mathbf{n}_{\mathbf{ba}_t} \\ \mathbf{0}_{3\times 1} \end{bmatrix} \tag{4}$$

$$\mathbf{u}_t = \begin{bmatrix} \boldsymbol{\omega}_m^T & \mathbf{a}_m^T \end{bmatrix}^T, \quad \mathbf{w}_t = \begin{bmatrix} \mathbf{n}_{\mathbf{g}}^T & \mathbf{n}_{\mathbf{a}}^T & \mathbf{n}_{\mathbf{bg}}^T & \mathbf{n}_{\mathbf{ba}}^T \end{bmatrix}^T$$

The forward propagation is performed upon receiving an IMU input, and the prior state $\hat{\mathbf{x}}$ can be propagated through the formula (3) by setting $\mathbf{w}_t$ to zero:

$$\hat{\mathbf{x}}_{t+1} = \bar{\mathbf{x}}_t \boxplus (\mathbf{f}(\bar{\mathbf{x}}_t, \mathbf{u}_t, 0)\Delta t) \tag{5}$$

To propagate the covariance, we utilize the error state dynamic model, which can be linearized as follows:

$$\begin{aligned} \delta\mathbf{x}_{t+1} &= \mathbf{x}_{t+1} \boxminus \hat{\mathbf{x}}_{t+1} \\ &\simeq \mathbf{F}_{\delta\mathbf{x}}\delta\mathbf{x}_t + \mathbf{F}_{\mathbf{w}}\mathbf{w}_t \end{aligned} \tag{6}$$

where $\delta\mathbf{x}$ is the error state between the ground-truth state and the prior state, $\mathbf{F}_{\delta\mathbf{x}}$ and $\mathbf{F}_{\mathbf{w}}$ are Jacobian matrices of the error state equation with respect to $\delta\mathbf{x}$ and $\mathbf{w}$, respectively.

Denoting the covariance of white noises $\mathbf{w}$ as $\mathbf{Q}$, and the prior covariance $\hat{\mathbf{P}}_{t+1}$ can be linearized as follows:

$$\hat{\mathbf{P}}_{t+1} = \mathbf{F}_{\delta\mathbf{x}}\bar{\mathbf{P}}_{t+1}\mathbf{F}_{\delta\mathbf{x}}^T + \mathbf{F}_{\mathbf{w}}\mathbf{Q}_t\mathbf{F}_{\mathbf{w}}^T \tag{7}$$

where $\bar{\mathbf{P}}$ is the posterior covariance matrix after iterative optimization.

D. Kinematic Thread

1) Contact Module: The introduction of probabilistic contact models can enhance the robot's accuracy in detecting contact [22]. We employ a straightforward foot force estimation method combined with threshold filtering to determine the foot contact status of the robot. The contact force $\mathbf{F}_i$ is defined as:

$$\mathbf{F}_i = \mathbf{J}_i^{-T}\boldsymbol{\tau}_i \tag{8}$$

where $\mathbf{J}_i \in \mathbb{R}^{3\times n_d}$ and $\boldsymbol{\tau}_i \in \mathbb{R}^{n_d}$ represent the Jacobian matrix and the joint torque measurement of the robot's i-th leg, respectively. n_d is the number of degrees of freedom for one leg. Using (8), the contact force $\mathbf{F}_i$ will get by after joint data input is received. The robot's i-th leg has achieved contact status if the value of $\mathbf{F}_i$ exceeds the set threshold (90N).

2) Foot Contact Points: Given the robot rotation matrix ${}^{G}\mathbf{R}_I$ and position ${}^{G}\mathbf{p}_I$ in the world frame, the foot contact position in the world frame can be calculated as follows:

$${}^{G}\hat{\mathbf{c}}_i = {}^{G}\mathbf{R}_I \cdot \mathcal{F}(\mathbf{s}_{i_m} - \mathbf{n}_c) + {}^{G}\mathbf{p}_I \tag{9}$$

where $\mathbf{s}_{i_m} \in \mathbb{R}^{n_d}$ is the joint angle measurement for the i-th leg of the robot, $\mathbf{n}_c$ represents the Gaussian noise included in the measurement of the joint data, and $\mathcal{F}(\mathbf{s}_{i_m} - \mathbf{n}_c)$ denotes the forward kinematics function with respect to the foot contact points in the IMU frame.

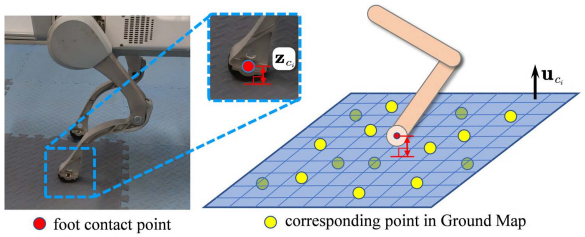

Fig. 2. Contact residual module. $\mathbf{u}_{c_i}$ and $\mathbf{z}_{c_i}$ are the normal vectors of the ground at the contact points of the i-th leg and the residual, respectively.

E. State Update

1) Ground Feature Extraction And Keyframe: The LiDAR system acquires measurements through point-by-point sampling, where k denotes the LiDAR index and ${}^{L}\mathbf{p}_{f_j}$ represents the j-th point in the scan, expressed in the LiDAR coordinate frame. First, we apply motion compensation to correct distortions, projecting all points in the k-th scan to its end-time t_k. Next, following the method of [16], we leverage LiDAR scan lines and incidence angle differences between adjacent points to extract ground feature points, denoted as $\mathbf{p}^{lg}$.

We follow the method in [17], employing approach that combines Euclidean distance-based criteria with time thresholds to select the keyframe. Let $\mathcal{K}_k$ be the keyframe at time k. Based on n keyframes near the current state, construct the *Sub-Map*. Note that the LiDAR scan frame of current state may not be keyframe. Extract the points in the *Sub-Map* that contain ground labels $\mathbf{p}^{lg}$ to construct the *Ground Map*. Additionally, we sequentially transform all keyframes into the initial frame coordinate system based on the converged pose state to construct the *Map*. To process point clouds quickly and efficiently,we utilize the ikd-Tree method [23] to manage all local maps.

2) Lidar Residual: The Lidar residual is modeled the same way as in [18], and denote the residual at point ${}^{L}\mathbf{p}_{f_j}$ in k-th scan as $\mathbf{z}_{L_j}^k$, is computed as follows:

$$\mathbf{z}_{L_j}^k = \mathbf{u}_{L_j}^T\left(\left({}^{G}\mathbf{T}_I\,{}^{I}\mathbf{T}_L\left({}^{L}\mathbf{p}_{f_j} - {}^{L}\mathbf{n}_{f_j}\right)\right) - {}^{G}\mathbf{q}_j\right) \tag{10}$$

where $\mathbf{u}_{L_j}$ and ${}^{G}\mathbf{q}_j$ are the normal vector and plane point corresponding to the *Map*, respectively. ${}^{G}\mathbf{T}_I$ and ${}^{I}\mathbf{T}_L$ are the relative pose between the global frame and the robot frame, and the external parameter relationship of LiDAR and IMU, respectively. ${}^{L}\mathbf{n}_{f_j}$ is the LiDAR noise at the j-th point. Let k denote the current iteration number of the iterative filter and let $\hat{\mathbf{x}}_t^k$ denote the corresponding state estimate. According to state propagation equation (5), the approximation of equation (10) at $\hat{\mathbf{x}}_t^k$ can be calculated as follows:

$$\begin{aligned} \mathbf{0} &= \mathbf{h}_j\left(\mathbf{x}_t, {}^{L}\mathbf{n}_{f_j}\right) \simeq \mathbf{h}_j\left(\hat{\mathbf{x}}_t^k, \mathbf{0}\right) + \mathbf{H}_{L_j}^k\delta\mathbf{x}_t^k + \mathbf{v}_j \\ &= \mathbf{z}_{L_j}^k + \mathbf{H}_{L_j}^k\delta\mathbf{x}_t^k + \mathbf{v}_j \end{aligned} \tag{11}$$

where $\mathbf{H}_L$ is the Jacobian matrix of LiDAR measurement residual $\mathbf{z}_{L_j}^k$ with respect to $\delta\mathbf{x}$, and $\mathbf{v}_j \in \mathcal{N}(\mathbf{0}, \mathbf{R}_j)$ represents the raw LiDAR measurement noise ${}^{L}\mathbf{n}_{f_j}$.

3) Contact Residual: Search for the n points corresponding to the foot contact point ${}^{G}\hat{\mathbf{c}}_i$ in the *Ground Map*, denoted as the set of $\mathcal{P}_c$, and fit a plane based on the principle of Principal

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

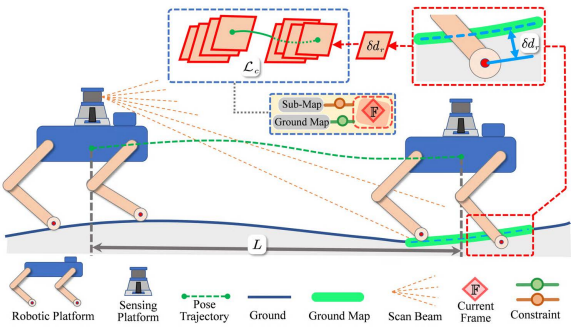

Fig. 3. Overview of the Ground Contact Constraint Module: An adaptive sliding window is constructed based on the LiDAR blind spot L. $\mathcal{L}_c$ is the contact residual in all gait cycles within the window. When $\mathcal{L}_c$ exceeds the set threshold, the corresponding constraint will be constructed for optimization of localization.

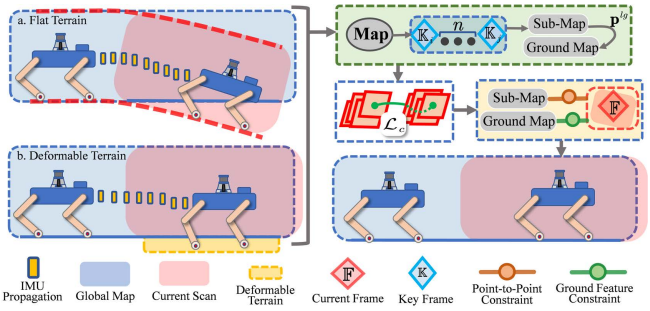

Fig. 4. The ground contact constrain modules are primarily triggered by two types of ground contact conditions: one is caused by flat terrain drift due to IMU prior error, and the other is caused by deformable terrain conditions. When the ground contact constraint module $\mathcal{L}_c$ detects current state drift, the constraints based on *Sub-Map* and *Ground Map* (see Section III-E1) are constructed to strengthen the optimization of the current state estimation.

Component Analysis (PCA), as shown in Fig. 2. And then, we using the point-to-plane residual to determine whether the plane fits successfully:

$$\begin{aligned} d_k &= \mathbf{u}_{c_i}^T\left(\mathbf{q}_k - \bar{\mathbf{q}}\right), \\ \bar{\mathbf{q}} &= \frac{1}{n}\sum_{k=1}^{n}\mathbf{q}_k, \quad \mathbf{q}_k \in \mathcal{P}_c. \end{aligned} \tag{12}$$

where $\mathbf{u}_{c_i}^T$ is the normal vector of the corresponding plane at foot contact point of the i-th leg, and $\bar{\mathbf{q}}$ is the average value of the search points. Unlike method [6] project foot point onto Map and then calculates the residuals, we directly calculating the point-to-plane residual more efficient and the the roughness of the plane fit is based on all relevant points rather than the average value. When all d_k are less than the set threshold $\boldsymbol{\sigma}$ (0.05 m) and the i-th leg is in contact state by (8), the contact residual can be calculated as follow:

$$\mathbf{z}_{c_i}^k = \begin{cases} \mathbf{u}_{c_i}^T\left({}^G\hat{\mathbf{c}}_i - \bar{\mathbf{q}}_i\right), & \forall d_{i_k} < \|\boldsymbol{\sigma}\| \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

Subsequently, we can approximation of equation (13) by its first order at $\hat{\mathbf{x}}_t^k$ as follows:

$$\begin{aligned} \mathbf{0} &= \mathbf{h}_i\left(\mathbf{x}_t, {}^c\mathbf{n}_i\right) \simeq \mathbf{h}_{c_i}\left(\hat{\mathbf{x}}_t^k, \mathbf{0}\right) + \mathbf{H}_{c_i}^k \delta\mathbf{x}_t^k + \mathbf{w}_c \\ &= \mathbf{z}_{c_i}^k + \mathbf{H}_{c_i}^k \delta\mathbf{x}_t^k + \mathbf{w}_c \end{aligned} \tag{14}$$

where $\mathbf{H}_{c_i}^k$ is the Jacobian matrix of residual $\mathbf{z}_{c_i}^k$ with respect to $\delta\mathbf{x}$, and $\mathbf{w}_c \in \mathcal{N}(\mathbf{0}, \mathbf{n}_c)$ represents the raw encoder measurement noise $\mathbf{n}_c$.

4) State Update: We utilize the Fast-LIO2 [18] method for state updates, where the maximum a posteriori (MAP) estimate is obtained by combining the prior from equation (5) with the measurement models from equations (11) and (14):

$$\begin{aligned} \min_{\delta\mathbf{x}_t}\Big(&\|\mathbf{x}_t \boxminus \hat{\mathbf{x}}_t\|_{\hat{\mathbf{P}}_t}^2 + \sum_{j=1}^{m}\|\mathbf{h}_j\left(\hat{\mathbf{x}}_t^k, \mathbf{0}\right) + \mathbf{H}_{L_j}^k \delta\mathbf{x}_t^k\|_{\mathbf{R}_j^{-1}}^2 \\ &+ \sum_{i=1}^{n}\|\mathbf{h}_{c_i}\left(\hat{\mathbf{x}}_t^k, \mathbf{0}\right) + \mathbf{H}_{c_i}^k \delta\mathbf{x}_t^k\|_{\Sigma_{\mathbf{n}_c}^{-1}}^2\Big) \end{aligned} \tag{15}$$

Following [18], the standard iterated Kalman filter can be computed as below, denote:

$$\begin{aligned} \mathbf{H} &= [\mathbf{H}_{L_1}^{k\,T}, \ldots, \mathbf{H}_{L_m}^{k\,T}, \mathbf{H}_{c_1}^{k\,T}, \ldots, \mathbf{H}_{c_i}^{k\,T}]^T \\ \mathbf{z}_t &= [\mathbf{h}_1(\hat{\mathbf{x}}_t, \mathbf{0}), \ldots, \mathbf{h}_j(\hat{\mathbf{x}}_t, \mathbf{0}), \mathbf{h}_{c_1}(\hat{\mathbf{x}}_t, \mathbf{0}), \ldots, \mathbf{h}_{c_i}(\hat{\mathbf{x}}_t, \mathbf{0})] \\ \mathbf{R} &= \mathrm{diag}(\mathbf{R}_1, \ldots, \mathbf{R}_m, \Sigma_{\mathbf{n}_1}, \ldots, \Sigma_{\mathbf{n}_i}) \end{aligned} \tag{16}$$

then, the Kalman gain and the state estimate can be computed as:

$$\begin{aligned} \mathbf{K} &= (\mathbf{P}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1} \\ \hat{\mathbf{x}}_t^{k+1} &= \hat{\mathbf{x}}_t^k \boxplus \left(-\mathbf{K}\mathbf{z}_t^k - (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{J}_k^{-1}(\hat{\mathbf{x}}_t^k \boxminus \hat{\mathbf{x}}_t)\right) \end{aligned} \tag{17}$$

The updated estimate $\hat{\mathbf{x}}_t^{k+1}$ is repeat the process until convergence (i.e., $\|\hat{\mathbf{x}}_t^{k+1} \boxminus \hat{\mathbf{x}}_t^k\| < \epsilon$). After convergence, the optimal estimation and covariance is:

$$\bar{\mathbf{x}}_t = \hat{\mathbf{x}}_t^{k+1}, \quad \bar{\mathbf{P}}_k = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P} \tag{18}$$

5) Ground Contact Constraint Module: The contact point of the legged robot is close to the ground. Leg-KILO [6] projects the contact points onto Map to calculate height variations for leg kinematic odometry updates, then uses this odometry's orientation as prior for LIO state estimation. Unlike Leg-KILO, our constraint module mainly monitors kinematic residuals and performs constraint module corrections after the state updated by LIO. Our ground contact constraint module effectively monitors the phenomenon of subsequent attitude drift caused by prior errors generated by violent vibrations during the IMU's forward propagation due to legged robots walking. As shown in Fig. 3, we construct an adaptive sliding window based on the LiDAR blind area L and build a residual accumulation function based on the contact residual (13) within the window to evaluate whether its status has drifted. The function is calculated as follows:

$$\begin{cases} \delta d_r = \|\mathbf{z}_{c_i}\|, \\ \mathcal{L}_c = \sum_{i=1}^{h}\|\delta d_{r_i}\|. \end{cases} \tag{19}$$

where δd_r is the contact residual calculated according to the state of the corresponding frame in the sliding window, get by equation (13). h represents the number of legged robot gait cycles within the sliding window.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

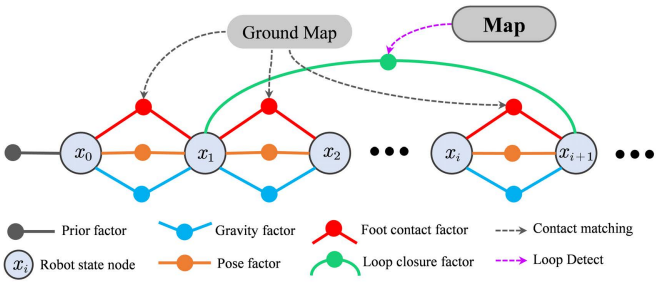

Fig. 5. Overview of the factor graph structure. The factor are: prior (grey), foot contact (from *Ground Map* contact matching, red), pose (orange), gravity (blue), and loop closure (from loop detect, green).

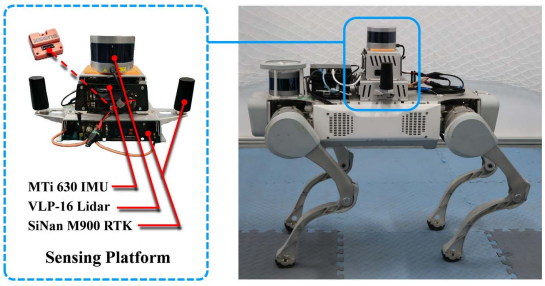

Fig. 6. The X20 robot with sensing platform. The external parameters between the sensors are calibrated by *lidar_align* [25].

Our modules are primarily triggered by two types of ground contact conditions: one is caused by flat terrain drift due to IMU prior error, and the other is caused by deformable terrain conditions. As shown in Fig. 4, when $\mathcal{L}_c$ in the sliding window exceeds the set threshold, the state is considered to have drifted. We then constructed two scan-to-map constraints based on the LIO-SAM method [17]. One constraint is between the current frame and the *Sub-Map*, which corrects errors in the rotation direction. The other constraint is between the point with ground feature labels in the current frame and the *Ground Map*, which mainly corrects height errors. Although this module offers fewer benefits on deformable terrain, but it provides a reliable constraint and monitoring solution for legged robots traversing multiple terrains.

F. Keyframes Graphs Optimization

We utilize factor graphs to tightly couple the gravity factor, foot contact, pose odometry, and loop closure factors. And using the keyframes strategy outlined in [17] for factor node selection and loop detection. For any two time-steps i and j ($j > i$) corresponding to keyframes, we denote the set of all gravity, foot contact, and pose odometry measurements as g_{ij}, $\mathcal{C}_{ij}$ and $\mathcal{O}_{ij}$, respectively. Additionally, denoting the loop constraints between keyframes m and n by $\mathcal{L}_{mn}$.

1) Gravity Factor g_{ij}: We introduce a gravity factor g_{ij} that uses the gravity g estimated from the states at keyframes i and j to construct a constraint aligned with the direction of the true gravity. We utilized the IMU_utils tool [24] to calibrate the true gravity values ${}^{G}g$. The residual terms of cost function are as follows:

$$\|\mathbf{r}_{g_{ij}}\|^2_{\Sigma_{g_{ij}}} \triangleq \|g_j - g_i\|^2_{\Sigma_{g_{ij}}} + \sum_{k\in(i,j)} \|g_k - {}^{G}g\|^2_{\Sigma_{g_{ij}}} \tag{20}$$

where the covariance matrix $\Sigma_{g_{ij}}$ is determined by the IMU calibration parameters.

2) Foot Contact Factor $\mathcal{C}_{ij}$: For the foot contact factor, we use the distance between the foot contact points ${}^{G}\mathbf{c}_k$ in the contact state and the *Ground Map* as the residual term, and ${}^{G}\mathbf{c}_k$ are get by (9). The number of foot contact points associated with the current factor will range from 2 to 4. The foot contact residual $\mathbf{r}_{\mathcal{C}_{ij}}$ can be defined:

$$\|\mathbf{r}_{\mathcal{C}_{ij}}\|^2_{\Sigma_{\mathcal{C}_{ij}}} \triangleq \sum_{k\in(2,3,4)} \|{}^{G}\mathbf{u}^T_{c_k}({}^{G}\mathbf{c}_k - {}^{G}\mathbf{q}_k)\|^2_{\Sigma_{\mathcal{C}_{ij}}} \tag{21}$$

where the covariance matrix $\Sigma_{\mathcal{C}_{ij}}$ is influenced by the accuracy of the joint motor encoder, ${}^{G}\mathbf{u}^T_c$ represents the normal vector of the ground near the foot contact point, while ${}^{G}\mathbf{q}$ denotes the point of closest with foot contact point ${}^{G}\mathbf{c}_k$ in the *Ground Map*.

Given the measurements from the keyframe, we can untilize the MAP method to estimate the set of states $\mathcal{X}^*_k$:

$$\begin{aligned}\mathcal{X}^*_k = \arg\min_{\mathcal{X}_k} \|\mathbf{r}_0\|^2_{\Sigma_0} + \sum_{(i,j)\in\mathcal{K}_k} \Big(\|\mathbf{r}_{g_{ij}}\|^2_{\Sigma_{g_{ij}}} + \|\mathbf{r}_{\mathcal{C}_{ij}}\|^2_{\Sigma_{\mathcal{C}_{ij}}} \\ + \|\mathbf{r}_{\mathcal{O}_{ij}}\|^2_{\Sigma_{\mathcal{O}_{ij}}} \Big) + \sum_{(m,n)\in\mathcal{K}_k} \|\mathbf{r}_{\mathcal{L}_{mn}}\|^2_{\Sigma_{\mathcal{L}_{mn}}}\end{aligned} \tag{22}$$

where $\mathbf{r}_0$ and Σ_0 represent the prior over the initial state, which is used to anchor the graph. The residual terms $\mathbf{r}_{\mathbf{g}_{ij}}$, $\mathbf{r}_{\mathcal{C}_{ij}}$, $\mathbf{r}_{\mathcal{O}_{ij}}$ and $\mathbf{r}_{\mathcal{L}_{mn}}$ correspond to the errors associated with gravity, foot contact, pose odometry, and loop closure measurements, respectively. And, the corresponding covariance matrices are denoted as $\Sigma_{g_{ij}}$, $\Sigma_{\mathcal{C}_{ij}}$, $\Sigma_{\mathcal{O}_{ij}}$ and $\Sigma_{\mathcal{L}_{mn}}$. The factor graph structure is shown in Fig. 5.

IV. EXPERIMENTS

The platforms used for our experiments are the Deep-robotics X20 robots. The robot features 4 identical legs, providing a total of 12 active degrees of freedom (DoF). Each leg is equipped with joint encoders and torque sensors. We have developed a sensor suite, as shown in Fig. 6, which includes a Velodyne VLP-16 LiDAR, an MTi 630 IMU, and a SiNan M900 RTK system. All experiments were conducted on an Intel NUC with i7-1165G7 processor and 16 GB of memory.

A. Dataset

To evaluate our proposed algorithm, we have performed comprehensive experiments using our datasets and the Leg-KILO public datasets[2]. Our dataset includes a variety of terrains: open grasslands, complex forests, large parks, rough terrain, slopes, deformable terrain, and gravel surfaces, as the Fig. 7 show. Sequences *wild-01* and *wild-02* are datasets of legged robots traversing flat and rough terrain, used to validate its localization capabilities during traversal in complex wild environments. Additionally, the *wild-02* includes all terrain features present in our other dataset.

[2]https://github.com/ouguangjun/legkilo-dataset/tree/main

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I
COMPARISON OF RELATIVE POSE ERROR (RPE) FOR OUR PROPOSED METHOD AND OTHER METHODS

Sequences (distance [m])	**Mean Euler Angle Error [deg] / Translation Error [m]**							
	Leg-KILO	LIO-SAM	Point-LIO	A-LOAM	Fast-LIO2	LLIO (w/o K)	LLIO (w/o LC)	LLIO (Full)
Park[*1] *(360m)*	0.415/0.435	0.984/0.779	0.948/0.335	0.969/0.518	**0.069/0.006**	0.116/0.014	0.086/0.009	0.070/0.006
*Corridor** *(288m)*	0.421/0.279	×[2]	0.695/0.274	0.859/0.040	×	×	0.577/0.019	**0.401/0.018**
Grass (195m)	0.324/0.319	0.268/0.685	0.332/0.325	0.342/0.667	0.258/0.321	0.255/0.320	0.253/0.318	**0.252/0.316**
Forest (261m)	0.212/0.405	×	0.213/0.404	0.250/0.545	0.122/0.402	0.126/0.412	0.124/0.403	**0.117/0.401**
Parks (1048m)	0.355/0.740	0.298/1.677	0.292/0.722	0.353/1.146	**0.240**/0.722	0.255/0.721	0.253/0.771	0.248/**0.720**
roughTerrain (328m)	0.416/0.055	×	0.305/0.020	0.329/0.025	0.277/0.022	0.300/0.021	0.279/0.017	**0.271/0.016**
wild-01 (290m)	0.439/0.064	×	0.331/0.022	0.301/0.021	0.320/0.021	0.302/0.021	0.294/0.019	**0.279/0.019**
wild-02 (1008m)	0.396/0.064	×	0.303/0.021	0.366/0.023	0.302/0.021	0.299/0.022	0.288/0.020	**0.254/0.019**

[1]∗ denotes that the sequence of the Leg-KILO public datasets; [2]× denotes that the method failed; The bold values represent the best results.

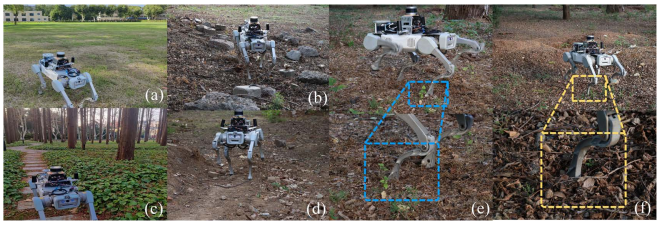

Fig. 7. (a) and (c) show the environments captured by the sequence *Grass* and *Forest*. (b), (d), (e) and (f) are rough terrains captured in sequence *roughTerrain*, *wild-01* and *wild-02*, including rubble, gravel surface, slopes, and deformable terrain. (e) and (f) are two main types of deformable terrain: soft sandy ground and forest leaf litter.

TABLE II
END-TO-END TRANSLATION ERROR [M]

Dataset	Leg KILO	LIO SAM	FAST LIO2	LLIO (w/o K)	LLIO (w/o LC)	LLIO (Full)
*Corridor**[1]	**0.04**	Fail	Fail	Fail	0.07	0.05
*slope**	0.83	1.63	1.46	0.86	0.14	**0.12**
Grass	1.08	4.06	3.68	1.80	0.63	**0.26**
Parks	1.06	0.82	1.68	1.26	0.35	**0.29**

[1]∗ denotes that the sequence of the Leg-KILO public datasets.
The bold values represent the best results.

B. Contact Module

To verify the accuracy of the proposed contact module, we have utilized an airbag sensor in our test experiments to determine the contact status between the foot-end point and the ground. The results of the experiment conducted during the transition from rest to movement of the legged robot's single leg are shown in Fig. 8(c). Throughout the walking process of the legged robot, the contact module can fully track and detect the state of foot contact. We have employed two thresholds to distinguish between standing and moving contact states of foot.

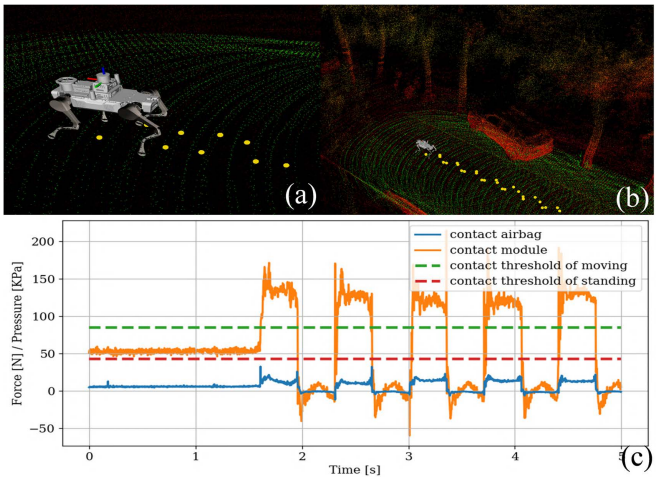

Fig. 8. (a) and (b) show the foot contact point during localization process, the green area represent the *Ground Map*, while the yellow point indicate the contact points of legged robots. (c) show the tracking of foot contact status during the transition from rest to movement of the legged robot's single leg, and we employ two thresholds to accurately differentiate between the foot being in motion and standing still.

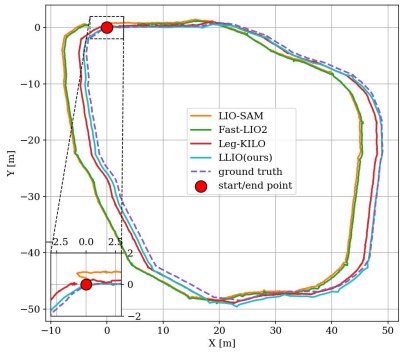

Fig. 9. Comparison of trajectory for various methods on *Grass*. Leg-KILO and LLIO (ours) are utilizing the Lidar-Kinematic-Inertial approach, demonstrate a closer alignment with the true trajectory.

C. Comparison Results

To highlight our method's performance, we compared it with the LiDAR-Kinematic-Inertial method (Leg-KILO [6]) and other state-of-the-art LiDAR methods, including LIO-SAM [17], Point-LIO [19], A-LOAM [15], and Fast-LIO2 [18]. Unless specified, "LLIO (w/o K)" and "LLIO (w/o LC)" denote versions without the ground contact constraint module

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

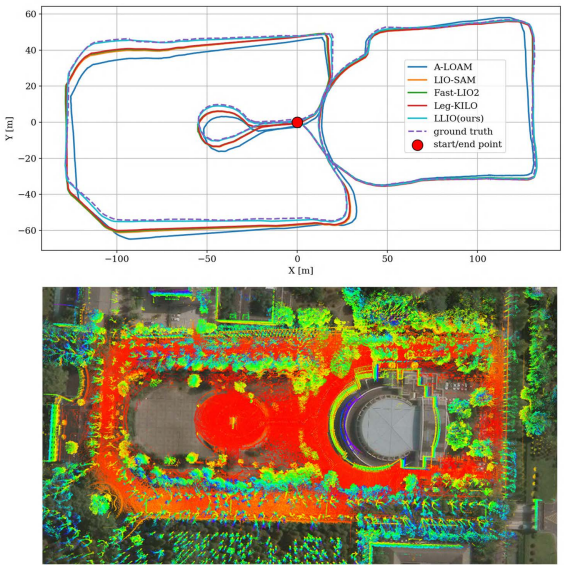

Fig. 10. *Top:* Comparison of trajectory for various methods on *Parks*. *Bottom:* LLIO map aligned with satellite map on *Parks*.

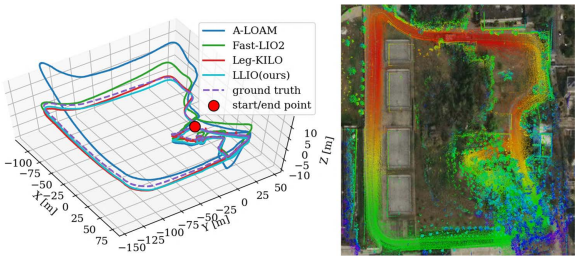

Fig. 11. *Left:* Comparison of trajectory for various methods on *wild-02*. *Right:* LLIO map aligned with satellite map on *wild-02*.

and loop detect closure, respectively. "LLIO (Full)" and "LLIO (ours)" includes all components. We utilized the translational relative pose error (RPE) to evaluate drift, and all experiments were assessed using the evo toolkit [26].

1) Accuracy Comparison: The comparison of mean Relative Pose Error (RPE) is presented in Table I. Our method outperforms others in relative pose accuracy. We find that LIO-SAM fails in wild datasets due to unstable IMU measurements and overreliance on feature point matching. As shown in Fig. 9, Leg-KILO and our LLIO leverage the constraints of Lidar-Kinematic-Inertial, resulting in estimated trajectories that are closer to true values. Furthermore, both methods were validated in large-scale dataset *Parks* and *wild-02*, demonstrating that the robot kinematic constraints can make the localization trajectory closer to the true value, as shown in Fig. 10 and Fig. 11. However, the kinematic constraints applied by Leg-KILO to the front end of the LIO system tend to cause local fluctuations in localization on rough terrain. As shown in Fig. 12, in *roughTerrain*, Leg-KILO exhibits more significant height fluctuations when the robot travels across different terrains, especially on deformable terrain and rubble terrain. In contrast, the kinematic constraints of our method act on the back end of LIO system, resulting in smaller fluctuations in the localization trajectory.

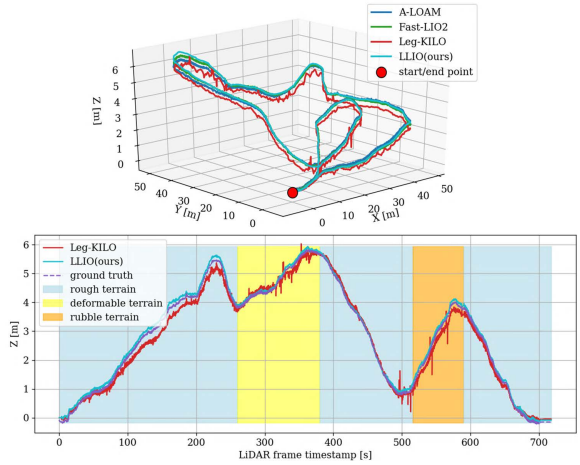

Fig. 12. *Top:* Comparison of trajectory for various methods on *roughTerrain*. *Bottom:* Comparison of body height's variation for Leg-KILO and LLIO(our) methods in the sequence *roughTerrain*. The light blue area is the legged robot traveling on gravel surface, as shown in Fig. 7(d). The yellow area is traveling on deformable terrain, including soft sand and leaf litter, as shown in Fig. 7(e-f). The orange area is traveling on rubble and slope, as shown in Fig. 7(b).

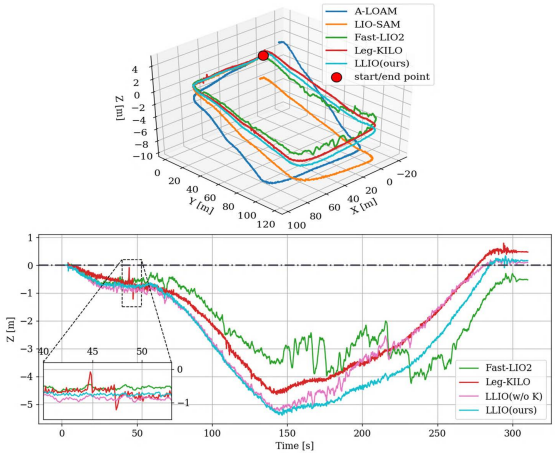

Fig. 13. *Top:* Comparison of trajectory for various methods on *slope*. *Bottom:* The height measurements for various methods on *slope*.

2) Ablation Study: From the Table II, it can be observed that the end-to-end error of methods with Kinematic constraints is smaller. The introduction of loop detect closure can further reduce the errors. As the Fig. 13 show, in *slope*, after LLIO(Full) introduced the ground contact constraint module, the local drift of height was significantly reduced, especially abrupt variations in elevation. The public sequence *Corridor* features a single-scene environments with flat terrain, which is prone to LiDAR SLAM degradation. Table I show that methods without kinematic constraints (e.g., LIO-SAM and Fast-LIO2) are prone to localization drift and instability, potentially leading to system failure. As illustrated in Fig. 14, at Time = 280 s, state estimation failed due to inaccurate a priori observations from IMU forward propagation in FastLIO2 and LLIO (w/o K). LLIO(Full) integrates a ground contact constraint module within its backend filter framework, enabling real-time error detection and correction of error states induced by IMU forward propagation.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

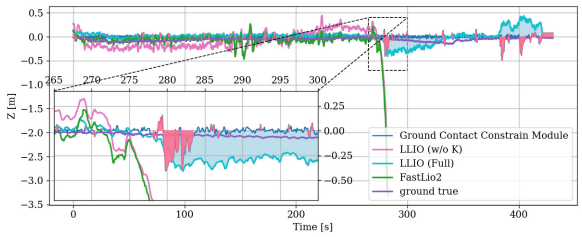

Fig. 14. Comparison of height's variation in *Corridor*. The light red area is triggering the ground contact constraint module. The light blue area is where the state is vulnerable to degradation and drift.

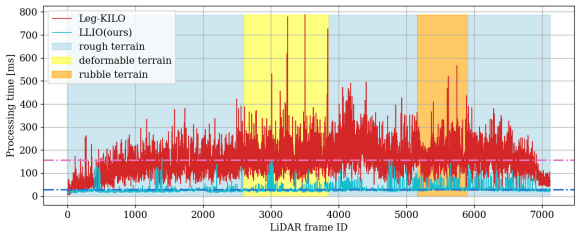

Fig. 15. Runtime comparison on sequence *roughTerrain*. The background areas of different colors are consistent with the description in Fig.12.

3) Efficiency: In the *roughTerrain* sequence, the robot's maximum speed and angular velocity remained below Leg-KILO's default LiDAR scan slicing threshold. Consequently, its state estimation output remained synchronization with LiDAR frame timestamps. Time consumption per LiDAR frame is shown in Fig. 15. Our method achieves real-time performance with an average processing time of 25 ms per frame, while Leg-KILO requires over 150 ms. This significant difference arises from Leg-KILO's computational pipeline: it projects foot contact points onto the Map upon receiving joint encoder data and computes foot height variations to update leg kinematics odometry. This method employs leg kinematics orientation as prior for LIO state estimation, but contact point projection incurs substantial computational overhead, particularly on deformable terrain (yellow region in Fig. 15). In contrast, our ground contact constraint module operates at the LiDAR frame rate, synchronizing with the LIO system's pose updates. This tight back-end integration enables efficient, real-time state estimation.

V. CONCLUSIONS

We have proposed LLIO, a framework for Lidar-Kinematic-Inertial Odometry. The integration of leg kinematics constraints and ground contact constraints into the back-end of LIO system, which enhances local accuracy and computational efficiency of robots on rough terrain. Our method employs an integrated factor graph optimization approach, incorporating gravity, foot contact, pose, and loop closure factors, which significantly reduce the cumulative error of the global trajectory. Extensive experiments and ablation studies demonstrate that our method is well-suited for legged robots compared to other state-of-the-art LiDAR-based approaches. In the future, we plan to incorporate vision into the ground feature optimization method to further enhance the accuracy of the ground feature constraints.

REFERENCES

[1] B. R. P. Singh and R. Featherstone, "Mechanical shock propagation reduction in robot legs," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1183–1190, 2020.
[2] M. Bloesch, M. Hutter, M. A. Hoepflinger, *et al.*, "State estimation for legged robots-consistent fusion of leg kinematics and imu," *Robot.s: Sci. Syst.*, vol. 17, pp. 17–24, 2013.
[3] M. Bloesch, C. Gehring, P. Fankhauser, *et al.*, "State estimation for legged robots on unstable and slippery terrain," in *2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2013, pp. 6058–6064.
[4] G. Bledt, M. J. Powell, B. Katz, *et al.*, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2018, pp. 2245–2252.
[5] D. Wisth, M. Camurri, and M. Fallon, "Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 309–326, 2022.
[6] G. Ou, D. Li, and H. Li, "Leg-kilo: Robust kinematic-inertial-lidar odometry for dynamic legged robots," *IEEE Robot. Automat. Lett.*, 2024.
[7] S. Yang, Z. Zhang, Z. Fu, and Z. Manchester, "Cerberus: Low-drift visual-inertial-leg odometry for agile locomotion," in *2023 IEEE Int. Conf. Robot. Automat.* IEEE, 2023, pp. 4193–4199.
[8] Y. Kim, B. Yu, E. M. Lee, J.-h. Kim, H.-w. Park, and H. Myung, "Step: State estimator for legged robots using a preintegrated foot velocity factor," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4456–4463, 2022.
[9] S. Yang, Z. Zhang, B. Bokser, and Z. Manchester, "Multi-imu proprioceptive odometry for legged robots," in *2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2023, pp. 774–779.
[10] R. Hartley, J. Mangelson, L. Gan, M. G. Jadidi, *et al.*, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *2018 IEEE Int. Conf. Robot. Automat.* IEEE, 2018, pp. 4422–4429.
[11] S. Song, B. Yu, M. Oh, and H. Myung, "Big-step: Better-initialized state estimator for legged robots with fast and robust ground segmentation," in *2023 23rd Int. Conf. Control, Automat. Syst.* IEEE, 2023, pp. 184–188.
[12] V. Dhédin, H. Li, S. Khorshidi, L. Mack, A. K. C. Ravi, A. Meduri, P. Shah, F. Grimminger, L. Righetti, M. Khadiv, *et al.*, "Visual-inertial and leg odometry fusion for dynamic locomotion," in *2023 IEEE Int. Conf. Robot. Automat.* IEEE, 2023, pp. 9966–9972.
[13] C. Pan, R. Hao, J. Yu, and L. Zhang, "Pose estimation algorithm for quadruped robots based on multi-sensor fusion," in *2024 Int. Symp. Intell. Robot. Syst.* IEEE, 2024, pp. 1–11.
[14] D. Wisth, M. Camurri, and M. Fallon, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *2020 IEEE Int. Conf. Robot. Automat.* IEEE, 2020, pp. 392–398.
[15] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Auton. robots*, vol. 41, pp. 401–416, 2017.
[16] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *2018 IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2018, pp. 4758–4765.
[17] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.
[18] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3317–3324, 2021.
[19] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-lio: Robust high-bandwidth light detection and ranging inertial odometry," *Adv. Intell. Syst.*, vol. 5, no. 7, p. 2200459, 2023.
[20] Q. Zhao, M. Li, Y. Shi, *et al.*, "Liko: Lidar, inertial, and kinematic odometry for bipedal robots," in *2024 IEEE Int. Conf. Robot. Automat.* IEEE, 2024, pp. 1180–1185.
[21] S. Xu, W. Zhang, and L. Zhu, "Klilo: Kalman filter based lidar-inertial-leg odometry for legged robots," in *2024 IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2024, pp. 12 487–12 492.
[22] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *2018 IEEE Int. Conf. Robot. Automat.* IEEE, 2018, pp. 4399–4406.
[23] Y. Cai, W. Xu, and F. Zhang, "ikd-tree: An incremental kd tree for robotic applications," *arXiv preprint arXiv:2102.10808*, 2021.
[24] G.Wenliang. (2020) imu_utils. [Online]. Available: https://github.com/gaowenliang/imu_utils
[25] E. ASL. (2018) lidar_align: A simple method for the extrinsic calibration. [Online]. Available: https://github.com/ethz-asl/lidar_align
[26] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.