# GaRLILEO: Gravity-aligned Radar-Leg-Inertial Enhanced Odometry

Journal Title
XX(X):i–xxvi

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

**Chiyun Noh[1†], Sangwoo Jung[1†], Hanjun Kim[1], Yafei Hu[2], Laura Herlant[2], and Ayoung Kim[1]**

**Abstract**

Deployment of legged robots for navigating challenging terrains (e.g., stairs, slopes, and unstructured environments) has gained increasing preference over wheel-based platforms. In such scenarios, accurate odometry estimation is a preliminary requirement for stable locomotion, localization, and mapping. Traditional proprioceptive approaches, which rely on leg kinematics sensor modalities and inertial sensing, suffer from irrepressible vertical drift caused by frequent contact impacts, foot slippage, and vibrations, particularly affected by inaccurate roll and pitch estimation. Existing methods incorporate exteroceptive sensors such as Light Detection and Ranging (LiDAR) or cameras. Further enhancement has been introduced by leveraging gravity vector estimation to add additional observations on roll and pitch, thereby increasing the accuracy of vertical pose estimation. However, these approaches tend to degrade in feature-sparse or repetitive scenes and are prone to errors from double-integrated IMU acceleration. To address these challenges, we propose **GaRLILEO**, a novel gravity-aligned continuous-time radar-leg-inertial odometry framework. GaRLILEO decouples velocity from the IMU by building a continuous-time ego-velocity spline from SoC radar Doppler and leg kinematics information, enabling seamless sensor fusion which mitigates odometry distortion. In addition, GaRLILEO can reliably capture accurate gravity vectors leveraging a novel soft $\mathcal{S}^2$-constrained gravity factor, improving vertical pose accuracy without relying on LiDAR or cameras. Evaluated on a self-collected real-world dataset with diverse indoor-outdoor trajectories, GaRLILEO demonstrates state-of-the-art accuracy, particularly in vertical odometry estimation on stairs and slopes. We open-source both our dataset and algorithm to foster further research in legged robot odometry and SLAM. https://garlileo.github.io/GaRLILEO/

**Keywords**

Radar, Legged Robot, Odometry, Gravity, SLAM

## 1 Introduction

Legged robots have increasingly attracted attention for their robust mobility and adaptability in harsh environments, where traditional wheeled Unmanned Grounded Vehicle (UGV) systems often struggle. Their ability to traverse stairs, steep slopes, uneven surfaces, and unstructured terrain makes them well-suited for real-world deployment in search-and-rescue, inspection, and exploration tasks (Tranzatto et al. 2022a). To fully leverage these capabilities in practice, it is essential to ensure accurate odometry estimation, which underpins stable locomotion, localization, and mapping in such challenging scenarios.

A common practice for robust state estimation in legged robots is leveraging proprioceptive sensing, which directly captures internal kinematics of the robot through contact measurements, joint encoders, and inertial sensing (Hartley et al. 2018b, 2020; Kim et al. 2021; Yang et al. 2023a; Lin et al. 2023). These proprioceptive approaches capitalize on the direct sensing of robot dynamics as they do not depend on external observations, making them inherently robust to visual or geometrical degradation. Nonetheless, frequent contact impacts, foot slippage, and intense vibrations significantly impair the accuracy of proprioceptive odometry, particularly in the vertical direction.

A natural progression to address this vertical drift is to leverage exteroceptive sensors, such as cameras and LiDARs. Most LiDAR-based methods apply ground segmentation and ground constraints to suppress vertical errors (Shan and Englot 2018; Wei et al. 2021; Seo et al. 2022; Wang et al. 2024). However, these strategies are mainly effective in wide, flat, and structured environments, which differ significantly from the cluttered and irregular terrains targeted by legged robots. Camera-based approaches also frequently rely on planar segmentation and Manhattan world assumptions to constrain vertical error (Li et al. 2020; Shu et al. 2021), yet these assumptions break down in complex natural environments. Moreover, recent studies further indicate that simply introducing planar landmarks may not guarantee direct improvement in managing odometry drift (Arndt et al. 2023), underscoring the limitations of exteroceptive registration in realistic legged robot scenarios.

Using an inertial measurement unit (IMU) with cameras and LiDARs can significantly enhance the state estimation. Among its many advantages, gravity estimation provides

[1] Dept. of Mechanical Engineering, SNU, Seoul, S. Korea
[2] Robotics and AI Institute, Cambridge, USA
[†] The authors contributed equally to this paper

**Corresponding author:**
Ayoung Kim, Dept. of Mechanical Engineering, SNU, Seoul, S. Korea
Email: ayoungk@snu.ac.kr

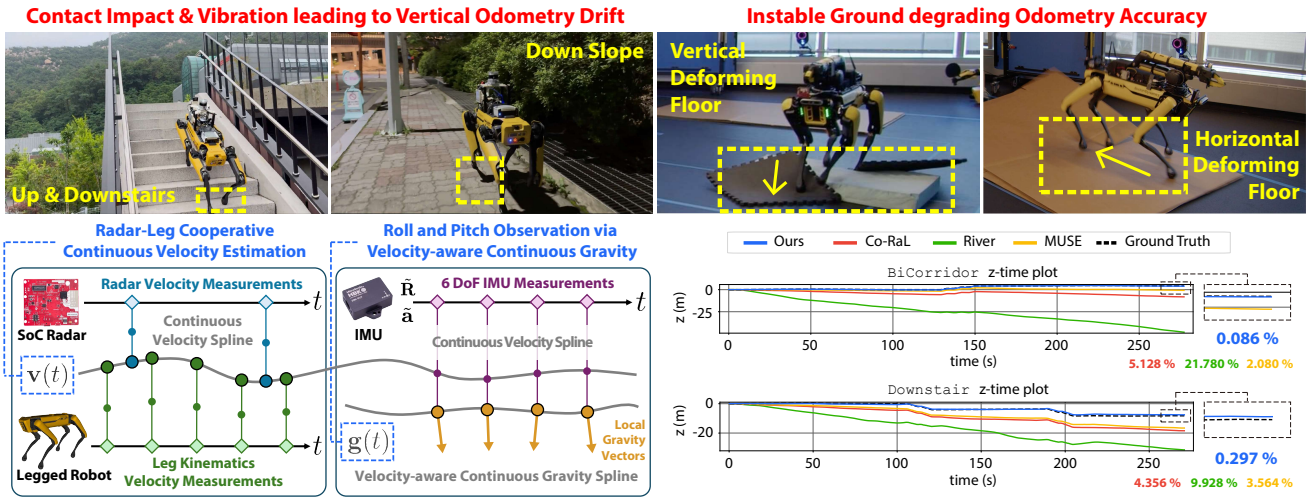



**Figure 1.** Overall preview of GaRLILEO. The four subfigures in the upper row present the problematic situations that quadrupedal robots may encounter while performing real-world tasks, while the yellow letters specify the situations and the red words explain the substantial issues generated from them. Two boxes in the left part of the lower row summarize the major contribution and method of the GaRLILEO, which significantly reduces odometry error, especially in the vertical direction. Two graphs in the right part of the lower row present the short experimental results, showing the accuracy of GaRLILEO in multiple sequences that include loops, sharp turns, and staircases, where most baselines fail to maintain accuracy in odometry estimation.

additional constraint for roll and pitch, enhancing state estimation, a concept that was first introduced in VINS-Mono (Qin et al. 2018). While some studies (Kubelka et al. 2022; Wang et al. 2023; Nemiroff et al. 2023; Ramezani et al. 2022; Agha et al. 2021) report notable improvements, others indicate only marginal performance gains (Burnett et al. 2025). This may be because most existing methods rely on fusing IMU acceleration with pose estimates. Such fusion requires double integration, and the gravity estimate inherently depends on the quality of the pose estimation. Consequently, feeding this pose-dependent gravity estimate back into the state may provide only limited improvement.

A promising alternative for overcoming these limitations is the use of radar. By providing direct velocity measurements, radar can fuse its ego-velocity with IMU acceleration via single integration for local gravity estimation (Noh et al. 2025). While effective, this integration can be limited in legged robot operations, where velocity changes sporadically due to frequent contacts and impacts. Another more intuitive way to integrate radar into the legged robot system is by utilizing the instantaneous ego-velocity derived from the leg kinematics as in Co-RaL (Jung et al. 2024). This approach takes advantage of radar's ability to operate reliably under environmental degradation while also utilizing the high-frequency, proprioceptive information provided by leg kinematics sensors. However, despite these improvements, residual vertical drift persists due to inaccurate roll and pitch estimation. This limitation primarily stems from the exclusive reliance on IMU gyroscopes for orientation.

To bridge the critical gap between proprioceptive odometry and robust gravity estimation, we extend our previous works GaRLIO (Noh et al. 2025) and Co-RaL (Jung et al. 2024) to complete **GaRLILEO**, a **G**ravity-**a**ligned **R**adar-**L**eg-**I**nertia**L** **E**nhanced **O**dometry framework. First, we decouple the IMU from velocity estimation, relying solely on radar and leg kinematics to compute the ego-velocity. This decoupled scheme is particularly advantageous for legged robot systems where noisy accelerations are often recorded during ground contact. Unlike previous radar-based approaches that represent velocity as a discrete state, our method formulates velocity as a continuous-time variable and enables seamless sensor fusion across modalities with different frame rates. We note that this is uniquely feasible in a leg–radar system, as radar directly measures velocity and leg kinematics can directly compute it, eliminating the need for pose-level measurements that would require double integration.

Beyond seamless fusion between radar and leg kinematics, the fundamental challenge of pose estimation of legged robots lies in the contact impacts and vibrations that persistently corrupt gravity estimation. To suppress these sudden, undesired acceleration measurements from the IMU, we employ splines as an inherent filter, bounding the gravity vector within a smoothed, continuous vector space. Furthermore, to naturally constrain the magnitude of the gravity vector during optimization, we introduce $\mathcal{S}^2$ gravity factor. Rather than conventionally restricting the vector to a unit sphere, this factor actively anchors its magnitude to 9.81 while optimizing its direction. This design enables precise gravity estimation, effectively constraining the roll and pitch of a legged robot, even under harsh conditions such as contact impacts, instantaneous slips, and intense vibrations. Our main contributions are as follows:

- The proposed continuous-time proprioceptive state estimation framework effectively overcomes the asynchronous, high-impact nature of radar-Leg-IMU systems. By formulating ego-centric velocity through splines and decoupling it from noisy IMU accelerations, abrupt motion prevalent in legged locomotions is better handled. This continuous formulation inherently filters out instantaneous leg slips and radar noise, providing a stable, distortion-free foundation for downstream gravity and pose

estimation without relying on visual or LiDAR features.

- Our velocity-aware gravity estimation directly attacks the pervasive issue of vertical drift in legged odometry. By integrating a soft $\mathcal{S}^2$-constrained gravity factor within our continuous-time framework, we continuously anchor the gravity vector's magnitude while optimizing its direction. This ensures that even under the intense vibrations caused by foot contacts, the system maintains a precise lock on the local gravity vector, reducing roll and pitch degradation.
- We present a comprehensive, real-world radar-Leg-Inertial dataset which features aggressive elevation changes across stairs, slopes, and slippery indoor/outdoor terrain. Validated against high-fidelity Terrestrial Laser Scanning (TLS) and motion capture ground truth trajectories, GaRLILEO demonstrates state-of-the-art (SOTA) performance, proving exceptionally resilient to z-axis variations where traditional methods fail. To foster further research, both the dataset and framework source code are released to the community.

## 2 Related works

In this section, we review prior work most relevant to our approach. We begin with System on Chip (SoC) radar odometry methods that leverage ego-velocity estimation. We then turn to recent developments in odometry for legged robots, covering both exteroceptive and proprioceptive paradigms. Building on this, we highlight advances in gravity estimation within state estimation frameworks. Lastly, we review continuous-time odometry methods that are particularly relevant to our work.

### 2.1 SoC Radar Odometry

Radars have emerged as a critical sensing modality in robotics, offering robust perception in adverse environments (Harlow et al. 2024). Robotics applications commonly utilize two types of radars: spinning radars and phased-array SoC radars (Kim et al. 2025b). This paper mainly focuses on SoC radars, which employ Frequency modulated continuous wave (FMCW) technology to generate 4D point clouds capturing range, azimuth, elevation, and Doppler radial velocity. While structurally similar to LiDAR data, SoC radar measurements are typically sparser and exhibit lower point precision, posing unique challenges for odometry estimation.

A primary approach to leveraging SoC radar for odometry involves directly estimating ego-velocity using point-wise radial velocity measurements. Early work by Kellner et al. (2013) introduced instantaneous ego-motion estimation using random sample consensus (RANSAC) for outlier rejection and least-squares optimization for a single SoC radar. Building on this foundation, subsequent studies can be categorized into two main directions: those integrating spatial information into ego-velocity estimation and those fusing it with inertial measurements from IMUs.

Several approaches have explored the integration of spatial information from SoC radar point clouds. For example, Michalczyk et al. (2022) utilized stochastic cloning to associate 3D points between consecutive point clouds, thereby enhancing odometry accuracy. Similarly, 4D-iRIOM (Zhuang et al. 2023) combined SoC radar velocity with scan-matching techniques, improving robustness by aligning scan-to-submap registration with Doppler-driven ego-velocity estimates. Recently, Huang et al. (2024) introduced Radar Cross Section (RCS)-based filtering to refine point correspondences. Despite these advancements, the low precision and sparsity of SoC radar point clouds often make registration vulnerable, leading to unbounded drift or failure in odometry estimation under challenging conditions.

In contrast to registration-based methods, other studies have integrated ego-velocity estimation with inertial measurements from IMUs. For instance, Doer and Trommer (2020) employed an extended Kalman filter (EKF), while Park et al. (2021) utilized factor-graph optimization to achieve 6-Degree of Freedom (DoF) odometry in visually degraded environments. Specifically, Park et al. (2021) leveraged two perpendicular SoC radars for 3D ego-velocity estimation and introduced a radar velocity factor for pose-graph simultaneous localization and mapping (SLAM) that incorporates IMU rotation data. More recently, DRIO (Chen et al. 2023) estimated ego-velocity and ground points, achieving robust 2D odometry. Also, Co-RaL (Jung et al. 2024) proposed a 4-DoF optimization strategy to mitigate vertical drift caused by the limited elevation resolution of SoC radars. Similarly, DeRO (Do et al. 2024) employed dead-reckoning with SoC radar ego-velocity and gyroscope data, further refined through an Iterative Extended Kalman Filter (IEKF) with tilt angle estimation based on accelerometers. Recently, River (Chen et al. 2024) tightly fused SoC radar ego-velocity and IMU measurements using a B-spline-based framework, presenting precise velocity estimation.

While prior works have significantly advanced SoC radar odometry by leveraging ego-velocity and spatial information, they face persistent challenges in mitigating vertical drift and ensuring robust performance under the dynamic conditions of legged robot systems. In such scenarios, low-precision vertical velocity measurements from SoC radars and contact-induced noise in IMU acceleration data often contribute to significant vertical drift in odometry estimation. Most existing methods either suffer from unbounded drift due to insufficient vertical constraints or rely on point cloud registration schemes that are highly sensitive to radar noise and sparsity. To overcome these limitations, our proposed method, GaRLILEO, seamlessly integrates radar-derived ego-velocity with high-rate proprioceptive measurements within a continuous-time framework, significantly reducing odometry distortion. Furthermore, our robust proprioceptive-based local gravity estimation scheme effectively mitigates vertical drift, enabling stable and accurate odometry even in dynamic and complex environments.

### 2.2 Leg kinematics Odometry

Compared with traditional wheeled UGV, a legged robot provides two unique sensor modalities: (1) contact sensors, which indicate the contact state of each foot, and (2) joint encoders, which measure the orientation of each joint. Using forward kinematics, the relative position of each foot with respect to the robot base can be computed (Roston and Krotkov 1992), completing leg-based odometry. The leg

odometry can be divided into two categories: methods that leverage exteroceptive sensors such as LiDAR or cameras, and those that focus primarily on fusing proprioceptive sensors alone.

*2.2.1 Exteroceptive Approach* A wide range of odometry frameworks for legged robots have harnessed exteroceptive sensors—such as cameras and LiDAR—in combination with leg kinematics, to capitalize on feature-rich geometric information. Fallon et al. (2014) introduced a pose estimator that fuses inertial and leg kinematics measurements with localization derived from LiDAR and pre-built maps. Similarly, Nobili et al. (2017) fused inertial, leg kinematics, camera, and LiDAR measurements based on the EKF, demonstrating robust state estimation across multiple walking gaits. This line of work was further extended by Pronto (Camurri et al. 2020), which focused on managing time-delayed signals from the vision sensors and fusing leg kinematics velocity from each leg by using a weighted average. VILENS (Wisth et al. 2022) proposed a tightly-coupled, graph-optimization-based approach by fusing camera, LiDAR, leg kinematics sensors, and IMU. This work enabled temporal proprioceptive odometry based on the preintegration factor when exteroceptive sensors failed under extreme conditions. STEP (Kim et al. 2022) improved the stability of stereo camera-based odometry by incorporating leg kinematics, achieving robust performance in dynamic scenes. Similarly, Leg-KILO (Ou et al. 2024) demonstrated stable LiDAR-based odometry by leveraging leg kinematics during dynamic movements of legged robots. Cerberus (Yang et al. 2023b) introduced online calibration of kinematics parameters and contact outlier rejection to reduce drift in camera-IMU-leg kinematics odometry. To address dynamic locomotion behaviors such as jumping and trotting at varying speeds, Dhédin et al. (2023) fused leg odometry from Pronto (Camurri et al. 2020) with IMU frequency speed camera-IMU odometry. Recently, MUSE (Nisticò et al. 2025) integrated foot-slip detection with camera-LiDAR odometry to further enhance robustness. Holistic Fusion (Nubert et al. 2025) provided a unified framework in which leg odometry, especially using the contact frame as a landmark feature, could be seamlessly fused with exteroceptive sensors.

While these exteroceptive sensor-based approaches have shown strong odometry performance—especially when combined with leg kinematics information—they still encounter challenges in environments where perceptual features are sparse, ambiguous, or repetitive. In particular, visual and LiDAR odometry can degrade rapidly in poorly illuminated areas, featureless corridors, reflective surfaces, or in the presence of dense smoke and dust, limiting their reliability in many real-world legged robot applications.

*2.2.2 Proprioceptive Approach* In contrast to exteroceptive approaches, proprioceptive-based odometry estimates the robot pose without dependence on external features. Bloesch et al. (2012) introduced an EKF-based framework that fused leg kinematics measurements and IMU data for state estimation, which was later extended to a Unscented Kalman Filter (UKF) backbone (Bloesch et al. 2013).

Based on the contact theorem that a contact frame is fixed while a contact sensor is on, Hartley et al. (2018b) proposed a forward kinematics factor and a preintegrated contact factor. This was later generalized to a hybrid contact factor that dynamically switches the contact frame based on the assumption that at least one foot is in contact with the ground (Hartley et al. 2018a). The contact kinematics theory was incorporated into an invariant EKF framework, yielding a Lie group-based estimator that achieved globally consistent state estimation using IMU, kinematics, and contact measurements (Hartley et al. 2020). Fink and Semini (2020) additionally fused force and torque sensor data to develop a low-level state estimator, calculating both kinematics and dynamics of the robot. Still, the possible foot slip, even with the contact sensor in place, remains a challenge.

To address foot slip on the contact frame, approaches from various directions have been introduced recently. TSIF (Bloesch et al. 2017) presented a recursive estimation framework minimizing the residual between two consecutive states and demonstrated improved resilience to measurement outliers. Kim et al. (2021) adopted a fixed-lag smoother state estimation on the SO(3) manifold, paired with slip rejection strategies to mitigate kinematics model failures. DRIFT (Lin et al. 2023) combined contact estimation and gyro filtering within an invariant EKF, enabling robust odometry on low-cost legged robots. Yang et al. (2023a) utilized multiple IMUs on each foot to explicitly detect contact and foot slip, overcoming the zero-velocity contact frame assumption prevalent in contact sensor-based proprioceptive odometry studies.

Although proprioceptive odometry is robust to environmental conditions and feature degradation, it suffers from drift over time, particularly in the vertical direction, due to the drastic vibration from contact impact and the lack of an absolute orientation reference. To address this, Co-RaL (Jung et al. 2024) introduced an integration of SoC radar-derived ego-velocity with leg kinematics velocity incorporating rolling contact awareness. However, despite these integrated measurements, vertical drift remains due to inaccurate roll and pitch estimation. This stems from not accounting for IMU acceleration, which contains critical information, such as the gravity vector, essential for accurate odometry.

GaRLILEO improves upon Co-RaL by integrating SoC radar, leg kinematics, and IMU data using a continuous-time B-spline approach and introducing robust continuous velocity-aware gravity estimation. This enables substantially improved robustness of odometry, especially in the vertical direction, compared to Co-RaL and other purely proprioceptive methods.

## 2.3 Local Gravity Estimation

Gravity, with its constant magnitude and direction, provides a physically grounded reference for reliable roll and pitch estimation in odometry and SLAM. Accurate roll and pitch estimation is crucial for mitigating vertical drift, which is stimulated by erroneous leakage of horizontal movement onto the vertical axis. Accordingly, accurate local gravity estimation has emerged as a critical constraint for suppressing accumulated vertical drift errors in state estimation frameworks.

Early gravity estimation in LiDAR odometry typically inferred the local gravity vector from IMU acceleration or employed probabilistic filtering over time. Nebula (Agha et al. 2021) introduced a gravity factor based on IMU acceleration during stationary intervals, constraining roll and pitch in the state estimation. D-LIOM (Wang et al. 2023) and Wildcat (Ramezani et al. 2022) formulated gravity alignment as an optimization constraint using IMU data and exteroceptive-based odometry. Nemiroff et al. (2023) further extended this by jointly optimizing accelerometer intrinsics and the gravity vector. While these velocity-ignorant models presented the potential for local gravity estimation to mitigate odometry vertical drift, they fundamentally relied on correlating pose changes from exteroceptive odometry with IMU acceleration, which amplified errors due to IMU bias and noise through the process of double integration. This limitation hinders robustness in dynamic environments or those prone to slippage. Furthermore, Burnett et al. (2025) reports that these methods offer only marginal performance gains; this is likely due to the inherently pose-dependent nature of velocity-ignorant gravity estimation, which limits the benefit of feeding estimation back into the state.

To overcome these issues, GaRLIO (Noh et al. 2025) explicitly incorporated direct velocity measurements from radar Doppler data for local gravity estimation. By fusing radar-derived ego-velocity with LiDAR odometry, GaRLIO constructed a velocity-aware gravity constraint that significantly enhanced the accuracy of gravity estimation. However, GaRLIO still depended on an initial gravity guess from LiDAR scan registration, making it susceptible in feature-degraded environments or unadaptable to a system without LiDAR. Additionally, mismatches in time intervals between the IMU preintegration and radar ego velocity updates may introduce estimation inconsistencies due to the lack of continuity in the discrete-time sensor-fusion approach.

In contrast, our GaRLILEO framework integrates a continuous velocity-aware local gravity estimation approach, enabling precise and robust estimation of the gravity vector even in visually degraded or featureless environments. Furthermore, by continuously fusing SoC radar velocity and leg kinematics measurements via a time-continuous B-spline optimization framework, GaRLILEO prevents the inconsistency that may arise from discrete sensor fusion across modalities with differing frame rates.

### 2.4 Continuous-time State Estimation

Sensor fusion is essential in robotics to leverage the complementary strengths of heterogeneous sensors. Traditional discrete-time frameworks synchronize each sensor by associating with the nearest available timestamp; however, this introduces motion distortion and overlooks latent state information between sampled states (Talbot et al. 2024).

To address this limitation, recent works have adopted B-spline-based continuous-time trajectory representations, enabling smooth and differentiable state estimation that supports querying poses, velocities, and accelerations at arbitrary time instances. Furgale et al. (2012, 2015) formulated batch estimation of robot trajectories in SE(3) using temporal B-splines, establishing the groundwork for continuous-time sensor fusion.

Ovrén and Forssén (2019); Hug and Chli (2020); Lang et al. (2022); Hug et al. (2022) extended these ideas for vision-inertial fusion, achieving improved robustness to motion distortion and asynchronous measurements. Similarly, Droeschel and Behnke (2018) leveraged B-splines for continuous-time SLAM with LiDAR-inertial system. Later, recent works like (Lv et al. 2023; Lang et al. 2023, 2024) addressed multi-modal SLAM and odometry between LiDAR, camera, and IMU. Jung et al. (2023) further addressed asynchronous fusion for multi-LiDAR-IMU odometry using B-spline approach for continuous-time formulations. For SoC radar-IMU systems, River (Chen et al. 2024) introduced a B-spline-based radar-inertial velocity estimator that could operate robustly under perception-degraded conditions.

Inspired by these recent advances, GaRLILEO presents the first integration of leg kinematics with SoC radar and IMU in a continuous-time B-spline framework. By directly incorporating leg kinematics velocities and radar measurements at their respective timestamps without preintegration, GaRLILEO provides more accurate and robust state estimation. Additionally, this continuous-time framework even enhances the gravity estimation accuracy, reducing the vertical drift, especially in challenging legged robot scenarios where slip and dynamic contact events frequently occur.

## 3 Preliminary

In this section, we summarize the background for the following sections.

### 3.1 Radar Radial Velocity and Ego Velocity

Leveraging the FMCW technique, phased array radar can provide not only the 3D position of the points that they acquire, but also the radial velocity of the point in the sensor coordinate. If a single radar point is generated from a stationary object, the radial velocity of the point can be represented with the ego-centric velocity of the sensor. Let the ego-centric velocity of the sensor is $\mathbf{v}$, the radial velocity of the point is $v_j$, and the 3D position of the point is represented with vector $\mathbf{p}_j$; then their relationship is like follows:

$$v_j = -\frac{\mathbf{p}_j}{\|\mathbf{p}_j\|}\mathbf{v}. \tag{1}$$

Using this radial velocity, we can calculate ego-velocity using RANSAC and least-square optimization processes (Kellner et al. 2013). For a simple illustration, let's assume that the radar acquires only 2D data. Then, equation (1) can be expressed as follows:

$$\begin{aligned} \theta_j &= \tan^{-1}\frac{p_{j,y}}{p_{j,x}} \\ v_j &= -\begin{bmatrix}\cos(\theta_j) & \sin(\theta_j)\end{bmatrix}\mathbf{v}, \end{aligned} \tag{2}$$

where $\mathbf{v} = \begin{bmatrix} v_x & v_y \end{bmatrix}^\top$. Expanding the (2) to all $N$ radar points, the result equation is like follows:

$$\begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) \\ & \vdots \\ \cos(\theta_N) & \sin(\theta_N) \end{bmatrix}\mathbf{v}, \tag{3}$$

while this equation can be expanded to 3D points as follows:

$$\begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix} = \begin{bmatrix} \frac{p_{1,x}}{p_{1,r}} & \frac{p_{1,y}}{p_{1,r}} & \frac{p_{1,z}}{p_{1,r}} \\ & \vdots & \\ \frac{p_{N,x}}{p_{N,r}} & \frac{p_{N,y}}{p_{N,r}} & \frac{p_{N,z}}{p_{N,r}} \end{bmatrix} \mathbf{v}, \tag{4}$$

where the $p_{j,r} = \sqrt{p_{j,x}^2 + p_{j,y}^2 + p_{j,z}^2}$ .

Based on the equation (3) or (4), whether the data is 2D or 3D, if the matrix including the angular information of points is referred to as $\mathbf{M}$, and the vector including the radial velocity of points is referred to as $\mathbf{v}_R$, the ego velocity $\mathbf{v}$ can be calculated by least-square optimization using a pseudo-inverse matrix as $\mathbf{v} = (\mathbf{M}^\top \mathbf{M})^{-1} \mathbf{M}^\top \mathbf{v}_R$, or based on Singular Value Decomposition (SVD) of matrix $\mathbf{M}$.

### 3.2 Leg Kinematics and Ego Velocity

Two key sensors for state estimation of the legged robot are the joint encoders and contact sensors. Joint encoders measure the absolute angle of each joint, being a key proprioceptive sensor for estimating the robot's pose. Contact sensors are positioned at every foot, measuring whether the foot is currently in contact or not.

Using the joint encoder measurement and the physical modeling of each node, the relative coordinate transformation between frames $\mathcal{F}^i$ and $\mathcal{F}^{i+1}$ at each joint can be calculated as follows:

$$\begin{aligned} \mathbf{T}_{i+1}^{i} &= \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ \mathbf{0}^\top & 1 \end{bmatrix} \\ \mathbf{R}_i &= \mathrm{Exp}(\alpha_i \, \mathbf{u}_i{}^\top), \end{aligned} \tag{5}$$

where $\alpha_i$ denotes the joint angle measured by the $i$-th joint encoder, and $\mathbf{u}_i$ is the basis vector of the $i$-th joint expressed in its local frame. Using the $\mathbf{T}$ on every joint, the relative coordinate between the robot base frame and the end-effector (foot) frame can be calculated as follows:

$$\mathbf{T}_{\text{foot}}^{\text{base}} = \mathbf{T}_{\text{hip}}^{\text{base}} \mathbf{T}_{\text{knee}}^{\text{hip}} \mathbf{T}_{\text{foot}}^{\text{knee}}, \tag{6}$$

assuming the hip and knee joints between the base coordinate and the foot coordinate. This chain-style frame calculation (i.e. forward kinematics) allows us to calculate the relative pose of each end-effector based on the base frame of the robot.

Under the no-slip assumption, when a single contact is made on any foot, the contacting frame of the foot should remain static until the contact state is resolved. Using this, during a foot is remaining contact state, the relative velocity of the base frame can be calculated by inverting the forward kinematics function to estimate the pose difference of the base frame during the contact state.

From this forward kinematics, we will derive the ego-centric velocity and use it for the velocity factor, which will be detailed in Section §4.3.

### 3.3 B-Spline Interpolation

A $k$-order B-spline consists of several polynomial segments of degree $k-1$ with at most $C^{k-2}$ continuity (Patrikalakis and Maekawa 2002). This continuous-time spline representation allows evaluating the state at arbitrary timestamps via smooth interpolation, which is particularly useful for fusing asynchronous sensors and reduces the need for explicit sensor-to-sensor time synchronization by enabling direct evaluation at each measurement time. Moreover, for legged robots where gait patterns introduce high-frequency noise, the smoothness of the spline acts as a low-pass filter, alleviating spike artifacts often observed in discrete-time alternatives.

Leveraging these advantages, we parameterize continuous-time trajectories of ego-centric velocity $\mathbf{v}(t) \in \mathbb{R}^3$ and rotation $\mathbf{R}(t) \in \mathrm{SO}(3)$ using third-order ($k$=3) uniform cumulative B-splines, each consisting of second-degree polynomial segments. Due to the continuity property, discontinuities in velocity and acceleration are effectively eliminated. This cumulative B-spline formulation offers $O(k)$ computational complexity in temporal derivatives, making it well suited for real-time odometry (Sommer et al. 2020).

At a given time $t \in [t_i, t_{i+1})$, the velocity $\mathbf{v}(t)$ depends exclusively on $k$ control points due to the local support property of B-splines, and its representation in matrix form is as follows:

$$\mathbf{v}(u) = \begin{bmatrix} \mathbf{v}_i & \mathbf{d}_1^i & \cdots & \mathbf{d}_{k-1}^i \end{bmatrix} \cdot \widetilde{M}^{(k)} \cdot \mathbf{u}^\top, \tag{7}$$

where $\mathbf{v}_i$ denotes the $i$-th control point of $\mathbf{v}(t)$, $\mathbf{d}_j^i = \mathbf{v}_{i+j} - \mathbf{v}_{i+j-1}$, $\mathbf{u} = \begin{bmatrix} 1 & u & \cdots & u^{k-1} \end{bmatrix}$, $u = (t - t_i)/(t_{i+1} - t_i)$ is the normalized time within the knot interval, and $\widetilde{M}^{(k)}$ is the cumulative spline matrix that depends solely on the B-spline order. Since $u$ is the only term that depends on time, differentiating equation (7) with respect to $t$ yields the following expression:

$$\dot{\mathbf{v}}(u) = \frac{1}{\Delta t} \begin{bmatrix} \mathbf{v}_i & \mathbf{d}_1^i & \cdots & \mathbf{d}_{k-1}^i \end{bmatrix} \cdot \widetilde{M}^{(k)} \cdot \dot{\mathbf{u}}^\top. \tag{8}$$

At this point, we'd like to note that, in this paper, the velocity spline is constructed from ego-centric velocity; therefore, $\dot{\mathbf{v}}$ does not directly represent ego-centric acceleration. Computing ego-centric acceleration requires additional considerations about the angular velocity.

Analogous to the $\mathbb{R}^3$ case, a cumulative B-spline of order $k$ defined on a Lie group $\mathcal{L}$ with control points $\mathbf{R}_0, \cdots, \mathbf{R}_N \in \mathcal{L}$ is expressed as:

$$\begin{aligned} \mathbf{R}(u) &= \mathbf{R}_i \cdot \prod_{j=1}^{k-1} \mathrm{Exp}\left(\lambda_j(u) \cdot \mathrm{Log}\left(\mathbf{R}_{i+j-1}^\top \mathbf{R}_{i+j}\right)\right) \\ \dot{\mathbf{R}}(u) &= \mathbf{R}_i \cdot \prod_{j=1}^{k-1} \left\{ \prod_{l=1}^{j-1} A_l(u) \right\} \dot{A}_j(u) \left\{ \prod_{l=j+1}^{k-1} A_l(u) \right\}, \end{aligned} \tag{9}$$

where $A_j(u) = \mathrm{Exp}\left(\lambda_j(u) \cdot \mathrm{Log}\left(\mathbf{R}_{i+j-1}^\top \mathbf{R}_{i+j}\right)\right)$, $\lambda(\tau) = \widetilde{M}^{(k)} \cdot \mathbf{u}(\tau)$, and $\lambda_j$ is $j$−th element of $\lambda$. More details can be found on (Sommer et al. 2020).

## 4 Factor Graph Formulation

### 4.1 State Definition and Notation

In this work, we adopt a robot-centric state to embed the local gravity within the system state, which is defined as follows:

$$
\begin{aligned}
\mathbf{x} &\triangleq \begin{bmatrix} \mathbf{x}_{\mathbf{R}} & \mathbf{x}_{\mathbf{v}} & \mathbf{x}_{\mathbf{g}} & \mathbf{x}_{\mathbf{b_v}} & \mathbf{x}_{\mathbf{b_a}} & | & \mathbf{g}^{\mathtt{G}} & \bar{\mathbf{R}}^{\mathtt{I}_0}_{\mathtt{G}} \end{bmatrix}, \text{where} \\
\mathbf{x}_{\mathbf{R}} &\triangleq \begin{bmatrix} \mathbf{R}^{\mathtt{G}}_{\mathtt{I}_0} & \cdots & \mathbf{R}^{\mathtt{G}}_{\mathtt{I}_{i-1}} \end{bmatrix} \\
\mathbf{x}_{\mathbf{v}} &\triangleq \begin{bmatrix} \mathbf{v}^{\mathtt{I}_0}_{\mathtt{I}_0} & \cdots & \mathbf{v}^{\mathtt{I}_{m-1}}_{\mathtt{I}_{m-1}} \end{bmatrix} \\
\mathbf{x}_{\mathbf{g}} &\triangleq \begin{bmatrix} \mathbf{g}^{\mathtt{I}_0} & \cdots & \mathbf{g}^{\mathtt{I}_{n-1}} \end{bmatrix} \\
\mathbf{x}_{\mathbf{b_v}} &\triangleq \begin{bmatrix} \mathbf{b}_{v_0} & \cdots & \mathbf{b}_{v_{i-1}} \end{bmatrix} \in \mathbb{R}^{2\times i} \\
\mathbf{x}_{\mathbf{b_a}} &\triangleq \begin{bmatrix} \mathbf{b}_{a_0} & \cdots & \mathbf{b}_{a_{i-1}} \end{bmatrix} \in \mathbb{R}^{3\times i} \\
\mathbf{g}^{\mathtt{G}} &\triangleq \begin{bmatrix} 0 & 0 & 9.81 \end{bmatrix}.
\end{aligned} \tag{10}
$$

The system state $\mathbf{x}$ consists of B-spline control points and bias terms. $\mathbf{x_R}$, $\mathbf{x_v}$ and $\mathbf{x_g}$ denote the control points of the B-splines for the $\mathrm{SO}(3)$ orientation $\mathbf{R}(t)$, ego-centric velocity $\mathbf{v}(t)$, and local gravity vector $\mathbf{g}(t)$, respectively. For $\mathbf{g}(t)$, we adopt an $\mathbb{R}^3$ spline parameterization for practical implementation and debugging simplicity compared to an $\mathcal{S}^2$ spline. $\mathbf{R}^{\mathtt{G}}_{\mathtt{I}}$ represents the rotation of the IMU frame expressed in the global frame, while $\mathbf{v}^{\mathtt{I}}_{\mathtt{I}}$ and $\mathbf{g}_{\mathtt{I}}$ denote the ego-centric velocity and the local gravity vector, both expressed in the IMU frame. The bias terms $\mathbf{b}_v$ and $\mathbf{b}_a$ correspond to the velocity state bias and the IMU accelerometer bias, respectively. The global coordinate frame $\mathcal{F}^{\mathtt{G}}$ is defined such that its $z$-axis is aligned with the global gravity vector $\mathbf{g}^{\mathtt{G}}$. The rotation matrix $\bar{\mathbf{R}}^{\mathtt{I}_0}_{\mathtt{G}}$ is used to align the estimated global gravity vector to $\mathbf{g}^{\mathtt{G}}$. Further details on $\mathbf{g}^{\mathtt{G}}$ and $\bar{\mathbf{R}}^{\mathtt{I}_0}_{\mathtt{G}}$ are included in Section §5.1.4.

To achieve sensor fusion with multiple sensor modalities, our algorithm employs an incremental factor graph optimization framework that leverages B-splines for optimizing each control point. The factor graph used in our approach comprises an IMU factor, a velocity factor, and a gravity factor, collectively enabling robust and accurate state estimation.

### 4.2 IMU Factor

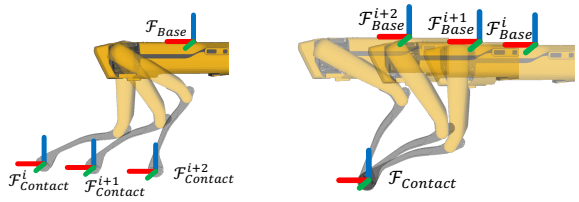
We employ two types of IMU factors: a gyroscope factor $\mathbf{r}_\omega$, and an accelerometer factor $\mathbf{r_a}$. Let $\tilde{\mathbf{a}}_{m_i}$ and $\tilde{\boldsymbol{\omega}}_{m_i}$ denote the acceleration and angular velocity measurements obtained from the IMU accelerometer and gyroscope at time $t_i$, respectively. The IMU measurements are modeled as

$$\boldsymbol{\omega}_{m_i} = \tilde{\boldsymbol{\omega}}_{m_i} - \mathbf{b}_{\omega_i} - \mathbf{n}_{\omega_i} \tag{11}$$

$$\mathbf{a}_{m_i} = \tilde{\mathbf{a}}_{m_i} - \mathbf{b}_{a_i} - \mathbf{n}_{a_i}, \tag{12}$$

where $\boldsymbol{\omega}_{m_i}$ and $\mathbf{a}_{m_i}$ are the true angular velocity and acceleration at time $t_i$; $\mathbf{b}_{\omega_i}$, $\mathbf{b}_{a_i}$ are the gyroscope and accelerometer biases; and $\mathbf{n}_{\omega_i}$, $\mathbf{n}_{a_i}$ are zero-mean Gaussian noises. In our setup, the gyroscope bias is weakly observable, so we set $\mathbf{b}_{\omega_i} \equiv \mathbf{0}$, while the accelerometer bias $\mathbf{b}_{a_i}$ is modeled as a Gaussian-driven random walk. Building on these equations, the residual functions for both IMU factors are formulated as follows, enabling direct optimization of control points for the continuous-time trajectory on rotation and velocity:

$$
\begin{aligned}
\mathbf{r}_\omega(t_i) &= \boldsymbol{\omega}(t) - \tilde{\boldsymbol{\omega}}_{m_i} \\
\mathbf{r_a}(t_i) &= \left(\boldsymbol{\omega}(t) \times \mathbf{v}(t) + \dot{\mathbf{v}}(t) - \mathbf{R}(t)^\top \mathbf{g}^{\mathtt{G}}\right) + \mathbf{b}_{a_i} - \tilde{\mathbf{a}}_{m_i}.
\end{aligned} \tag{13}
$$




(a) Body-centric Leg Locomotion (b) Contact-centric Leg Locomotion

**Figure 2.** Comparison between body-centric and contact-centric leg locomotion of a single leg during the contact state. (a) On body-centric calculation, the end-effector position differs as time passes. Based on the contact information from the contact sensor positioned at every foot, every contact frame should remain static while the contact sensor is on. Therefore, using the forward kinematics, the ego-centric velocity of the robot base can be calculated as in (b).

### 4.3 Velocity Factor

Our system leverages two sources of ego-centric velocity: a radar-derived velocity and a leg kinematics-based velocity. For the radar sensor, assuming that the $j-$th target point of radar measurements $\tilde{\mathbf{p}}^{\mathtt{r}}_j$ is stationary, the Doppler measurement $\tilde{v}^{\mathtt{r}}_{m,j}$ equals the magnitude of the ego-velocity projected onto the unit line-of-sight vector to the target point and its conversion into the IMU coordinate frame is expressed as:

$$\tilde{v}^{\mathtt{r}}_{m_i,j} = -\frac{(\tilde{\mathbf{p}}^{\mathtt{r}}_j)^\top}{\left\|\tilde{\mathbf{p}}^{\mathtt{r}}_j\right\|} \left(\mathbf{R}^{\mathtt{I}}_{\mathtt{r}}\right)^\top \left(\mathbf{v}_i + \lfloor \boldsymbol{\omega}^{\mathtt{I}}_i \rfloor_\times \mathbf{t}^{\mathtt{I}}_{\mathtt{r}}\right), \tag{14}$$

where $[\mathbf{R}^{\mathtt{I}}_{\mathtt{r}}, \mathbf{t}^{\mathtt{I}}_{\mathtt{r}}]$ is the extrinsic matrix from radar frame $\mathcal{F}^{\mathtt{r}}$ to IMU frame $\mathcal{F}^{\mathtt{I}}$ and $\mathbf{v}_i$, $\boldsymbol{\omega}^{\mathtt{I}}_i$ are true ego-centric velocity and angular velocity at time $t_i$. Because static points predominate in most environments, this formulation provides highly robust velocity estimates; however, the limited point resolution of radar introduces noticeable inaccuracies, particularly along the elevation axis, as mentioned in CoRaL (Jung et al. 2024).

The translation between the robot's body frame $\mathcal{F}^{\mathtt{B}}$ and the contact frame $\mathcal{F}^{\mathtt{C}}$ at time $t_i$ can be obtained using the following forward-kinematics function $\mathbf{f}_p$ :

$$\mathbf{t}^{\mathtt{B_i}}_{\mathtt{C_i}} = \mathbf{f}_p\left(\tilde{\boldsymbol{\alpha}}_i\right), \tag{15}$$

where the $\tilde{\boldsymbol{\alpha}}$ is the vector of joint angle measurements from the encoders on every joint at time $t_i$. Joint position and velocity are obtained from the leg-joint encoders, and the driver calculates which foot is in contact with the ground *. These measurements enable computation of ego-centric velocity in the robot's body frame,

$$\mathbf{v}^{\mathtt{B_i}}_{\mathtt{B_i}} = -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}}_i)\dot{\tilde{\boldsymbol{\alpha}}}_i - \boldsymbol{\omega}^{\mathtt{B}} \times \mathbf{f}_p(\tilde{\boldsymbol{\alpha}}_i), \tag{16}$$

where $\mathbf{J}_p(\cdot) \in \mathbb{R}^{3\times 3}$ is a Jacobian of $\mathbf{f}_p(\cdot)$ (Wisth et al. 2022). Empirically, the joint velocity reported by the encoders equals the finite difference approximation of the joint positions; however, under highly nonlinear joint trajectories,

*The Spot provides this information by measuring the ground reaction forces applied to each foot and provides this information

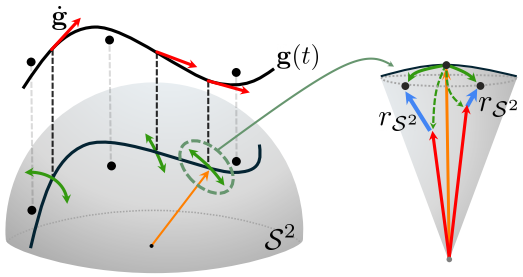



**Figure 3.** Soft $\mathcal{S}^2$-constrained gravity factor $r_{\mathcal{S}^2}$. Prior to optimization, the orange vectors are initialized through gravity spline extrapolation. During optimization, the factor (blue) constrains vectors that would otherwise drift in the $\mathbb{R}^3$ manifold (red), steering them to lie on the $\mathcal{S}^2$ surface (green).

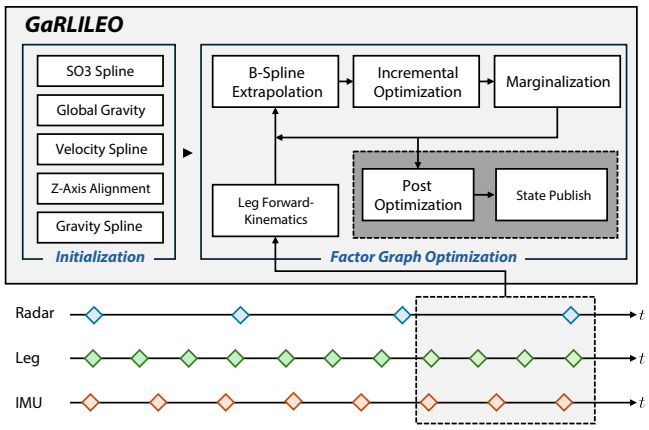



**Figure 4.** Overview pipeline of GaRLILEO

the velocity obtained in this manner diverges from the actual velocity. Therefore, we computed the ego-centric velocity by differentiating the leg translation function $\mathbf{f}_p(\tilde{\boldsymbol{\alpha}})$ derived from joint positions only as follows:

$$\mathbf{v}_{\mathrm{B_i}}^{\mathrm{B_i}} = -\frac{\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}_{i+1}) - \mathbf{f}_p(\tilde{\boldsymbol{\alpha}}_i)}{t_{i+1} - t_i}, \tag{17}$$

which is intuitively explained in Figure 2. Using the contact-centric leg kinematic locomotion, the ego-centric velocity of the system can be estimated. While this approach provides accurate estimates when the contact frame remains fixed to the ground, practical deployments must account for leg slip. Accordingly, we augment the leg kinematics factor with a velocity bias term to account for slip-induced drift of the leg kinematics-based velocity, particularly on slippery or deformable surfaces. Because slippage occurs primarily along the horizontal $x-$ and $y-$directions, the velocity bias is modeled exclusively in these two dimensions. Taking these considerations into account, the corresponding leg kinematics factor $\mathbf{r_L}$ and the radar factor $\mathbf{r_R}$ are defined as

$$\begin{aligned}
\mathbf{r_L}(t_i) &= \left(\mathbf{R}_{\mathrm{B}}^{\mathrm{I}}\right)^{\top}\left(\mathbf{v}(t_i) + \lfloor\boldsymbol{\omega}(t_i)\rfloor_{\times}\mathbf{t}_{\mathrm{B}}^{\mathrm{I}}\right) - \mathbf{v}_{\mathrm{B_i}}^{\mathrm{B_i}} - \begin{bmatrix}\mathbf{b}_v & 0\end{bmatrix} \\
\mathbf{r_R}(t_i) &= \sum_{j\in[1,q]}\left[-\frac{(\tilde{\mathbf{p}}_j^{\mathrm{r}})^{\top}}{\|\tilde{\mathbf{p}}_j^{\mathrm{r}}\|}\left(\mathbf{R}_{\mathrm{r}}^{\mathrm{I}}\right)^{\top}\left(\mathbf{v}(t_i) + \lfloor\boldsymbol{\omega}(t_i)\rfloor_{\times}\mathbf{t}_{\mathrm{r}}^{\mathrm{I}}\right)\right] \\
&\quad + \sum_{j\in[1,q]}\left[-\tilde{v}_{m_i,j}^{\mathrm{r}}\right],
\end{aligned} \tag{18}$$

where $[\mathbf{R}_{\mathrm{B}}^{\mathrm{I}}, \mathbf{t}_{\mathrm{B}}^{\mathrm{I}}]$ is the extrinsic matrix from robot base frame to IMU frame, and $q$ refers number of radar targets.

### 4.4 Gravity Factor

To mitigate the LiDAR-dependent and discrete-time sensor fusion limitations of GaRLIO (Noh et al. 2025) and enable robust gravity estimation for legged robot platforms, we adopt a B-spline-based, velocity-aware gravity factor $\mathbf{r_G}$ to parameterize continuous-time state functions as follows:

$$\mathbf{r_G}(t_i) = \sum_{i,\, j\in W}\left\|\mathbf{g}(t_i) - \frac{\tilde{\mathbf{R}}_{m_i}^{\top}\tilde{\mathbf{R}}_{m_j}\mathbf{v}(t_j) - \mathbf{v}(t_i) - \beta_j^i}{t_j - t_i}\right\|^2, \tag{19}$$

where $t_i$ and $t_j$ denote the timestamps of the $i$-th and $j$-th IMU measurements included in window $W$, respectively. $\tilde{\mathbf{R}}_m$ denotes the onboard AHRS orientation measurement obtained from internal filter of IMU by fusing magnetometer with standard inertial measurements, while $\beta_j^i = \int_{t\in[t_i,t_j)}\tilde{\mathbf{R}}_m(t)(\tilde{\mathbf{a}}_t - \mathbf{b}_{a_t})dt$. Furthermore, to mitigate erroneous gravity estimation caused by IMU noise, we also introduce a gravity spline equipped with a smoothing constraint $\mathbf{r}_{\mathcal{S}^2}$ (Figure 3). The gravity spline control points are subject to a fixed-norm constraint. Still, since we parameterize local gravity using splines on an $\mathbb{R}^3$ manifold rather than an $\mathcal{S}^2$ manifold, the interpolated B-spline segments between control points are not explicitly norm-constrained. To resolve this, a novel first-order derivative factor is defined and applied to the spline as follows:

$$\mathbf{r}_{\mathcal{S}^2}(t) = \dot{\mathbf{g}}(t) + \lfloor\boldsymbol{\omega}(t)\rfloor_{\times}\mathbf{g}(t), \tag{20}$$

which enforces a soft norm constraint throughout the spline trajectory.

## 5 Continuous Radar-Leg-IMU Fusion

Figure 4 illustrates the overall pipeline of GaRLILEO. The algorithm performs state estimation using three distinct splines representing $\mathrm{SO}(3)$ rotation, ego velocity, and local gravity. In the initialization stage, the three splines and global gravity vector are recovered from the input data, similar to River (Chen et al. 2024). After transforming the data into a gravity-aligned coordinate frame, a tightly coupled incremental factor graph optimization is executed. Subsequently, a rotation refinement procedure using the estimated local gravity is applied to the marginalized older states, followed by odometry estimation through a dead reckoning approach.

### 5.1 Initialization

*5.1.1 SO(3) Spline Initialization* The initial $\mathrm{SO}(3)$ spline is constructed by applying the factor $\mathbf{r}_\omega$, using IMU gyroscope measurements as constraints. The initial global frame is set as the initial frame of IMU. The initialization is formulated as

$$\min_{\mathbf{x_R}} \sum_{k\in W_{init}}\|\mathbf{r}_\omega(t_k)\|^2. \tag{21}$$

*5.1.2 Global Gravity Initialization* Our method refines rotation by estimating local gravity and comparing it with global gravity. Therefore, accurate estimation of the global gravity reference is critical. For global gravity estimation,

two different methods are popularly exploited: dynamic initialization and stationary initialization methods.

The first type is *dynamic initialization* introduced in River (Chen et al. 2024), which estimates global gravity using an initial $\mathrm{SO}(3)$ spline and radar measurements. Here, ego-velocities are computed from radar scans collected during the initialization phase, and then are converted into the global frame using the initial $\mathrm{SO}(3)$ spline. The global gravity is estimated using the following equation:

$$\begin{aligned} \min_{\mathbf{g}_{\mathrm{I}_0}} \quad & \sum_{k\in W_{init}} \|\mathbf{r_{dg}}(t_k)\|^2, \\ \text{s.t.} \quad & \left\|\mathbf{g}^{\mathrm{I}_0}\right\| = 9.81, \\ \text{where} \quad & \mathbf{r_{dg}}(t_k) = \frac{\alpha^{\mathrm{k}}_{\mathrm{k+1}} - \mathbf{R}(t_{k+1})\mathbf{v}^{\mathrm{I_{k+1}}}_{\mathrm{I_{k+1}}} + \mathbf{R}(t_{\mathrm{k}})\mathbf{v}^{\mathrm{I_k}}_{\mathrm{I_k}}}{t_{k+1}-t_k} - \mathbf{g}^{\mathrm{I}_0} \\ & \alpha^{\mathrm{k}}_{\mathrm{k+1}} = \int_{t\in[\mathrm{r_k},\mathrm{r_{k+1}})} \mathbf{R}(t)(\tilde{\mathbf{a}}_t - \widehat{\mathbf{b}}_{a_t})dt. \end{aligned} \tag{22}$$

However, due to the highly nonlinear characteristics of IMU acceleration measurements from contact impact during legged robot locomotion, estimating global gravity dynamically at the radar frame rate using only the initial $\mathrm{SO}(3)$ spline results in inaccurate estimation. Therefore, a more stable method for estimating the global gravity is required on the legged robot UGV system.

Instead, we adopted a *stationary initialization* method for precise global gravity estimation. By computing the mean and variance of ego-velocities derived from accumulated radar scans during the initialization phase, the stationary condition can be determined using the following equation:

$$\mathrm{mean}(\mathbf{v}^{\mathrm{I_k}}_{\mathrm{I_k}}) < \tau_1 \quad \& \quad \mathrm{tr}(\mathrm{Var}(\mathbf{v}^{\mathrm{I_k}}_{\mathrm{I_k}})) < \tau_2. \tag{23}$$

Under stationary conditions, the accelerometer measurements directly correspond to gravity. The mean of these measurements $\bar{\tilde{\mathbf{a}}}$ serves as a prior factor, which is combined with the factor from the dynamic initialization method to calculate global gravity:

$$\begin{aligned} \mathbf{r_{sg}}(t_k) &= w_1 \cdot (\bar{\tilde{\mathbf{a}}} - \mathbf{g}^{\mathrm{I}_0}) \\ &+ w_2 \cdot \left(\frac{\alpha^{\mathrm{k}}_{\mathrm{k+1}} - \mathbf{R}(t_{k+1})\mathbf{v}^{\mathrm{I_{k+1}}}_{\mathrm{I_{k+1}}} + \mathbf{R}(t_{\mathrm{k}})\mathbf{v}^{\mathrm{I_k}}_{\mathrm{I_k}}}{t_{k+1}-t_k} - \mathbf{g}^{\mathrm{I}_0}\right). \end{aligned} \tag{24}$$

Since this approach yields a more stable and accurate global gravity estimation compared to dynamic initialization on a legged robot, every sequence included in the dataset of this paper begins stationary, thereby enabling the use of static initialization.

*5.1.3 Velocity Spline Initialization* The radar, IMU, and leg kinematics information are leveraged to construct the ego-centric velocity spline. During this process, velocity and acceleration biases are simultaneously initialized by solving the following Least Square (LSQ) problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad \Bigg\{ & \sum_{k\in I} w_\omega \|\mathbf{r}_\omega(t_k)\|^2 + \sum_{k\in I} w_a \|\mathbf{r_a}(t_k)\|^2 + \\ & \sum_{k\in L} w_L \|\mathbf{r_L}(t_k)\|^2 + \sum_{k\in R} w_R \|\mathbf{r_R}(t_k)\|^2 \Bigg\}. \end{aligned} \tag{25}$$

To maintain consistency, the $\mathrm{SO}(3)$ rotation spline is fixed during the initialization process of the velocity spline.

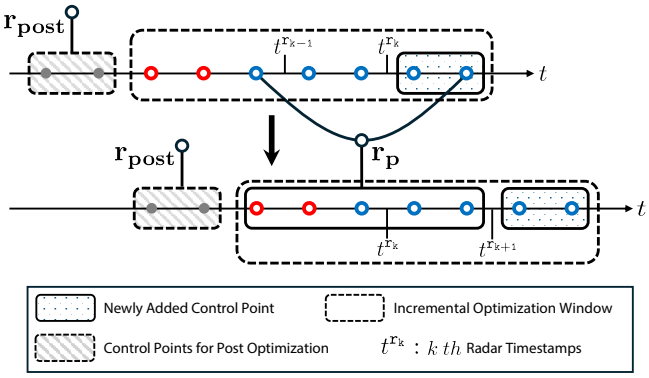



**Figure 5.** Factor Graph Overview. At each iteration, the number of control points marginalized (grey) equals the number newly added in the preceding window (red); the remaining control points are carried forward as a prior factor for the next solve (blue). The marginalized control points (grey) are then subjected to a post-optimization stage that refines the SO(3) spline.

*5.1.4 Z-Axis Alignment with Global Gravity* The gravity vector has only 2-DoF observations, roll and pitch direction (Kubelka et al. 2022). Thus, to ensure optimization updates are confined to observable axes without explicit constraints, the coordinate frame is transformed by aligning its $z$-axis with the global gravity vector. Consequently, the $\mathrm{SO}(3)$ splines undergo the same coordinate transformation:

$$\begin{aligned} \mathbf{x_R} &= (\bar{\mathbf{R}}^{\mathrm{I}_0}_{\mathrm{G}})^\top \cdot \mathbf{x_R} \cdot \bar{\mathbf{R}}^{\mathrm{I}_0}_{\mathrm{G}}, \\ \text{where} \quad \mathbf{g}^{\mathrm{G}} &= (\bar{\mathbf{R}}^{\mathrm{I}_0}_{\mathrm{G}})^\top \cdot \mathbf{g}^{\mathrm{I}_0}. \end{aligned} \tag{26}$$

*5.1.5 Gravity Spline Initialization* The final phase of initialization is on gravity spline. We initialize the gravity spline by recovering it through control points computed as

$$\mathbf{g}^{\mathrm{I_k}} = (\mathbf{R}^{\mathrm{G}}_{\mathrm{I_k}})^\top \mathbf{g}^{\mathrm{G}}, \tag{27}$$

using the previously initialized $\mathrm{SO}(3)$ splines.

## 5.2 Factor Graph Optimization

*5.2.1 Incremental Optimization* Following initialization and spline recovery, incremental factor graph optimization is performed on the B-spline control points, tightly coupling IMU, leg kinematics, and radar measurements. When new control points are introduced, an equal number of the oldest points in the window are marginalized, and the information of the remaining non-marginalized control points is condensed into a prior factor and propagated to the next optimization window (Figure 5). The end time of the window is determined by the timestamp of the incoming radar topic, and optimization begins after the three B-splines are linearly extended and their control points appended. The factor graph comprises both gravity factors $\mathbf{r_G}, \mathbf{r}_{\mathcal{S}^2}$; an IMU gyroscope factor $\mathbf{r}_\omega$; radar and leg kinematics velocity factors $\mathbf{r_R}, \mathbf{r_L}$; a bias-prior factor $\mathbf{r_b}$; a marginalization-prior factor $\mathbf{r_p}$; and an end-tail factor $\mathbf{r_e}$. The optimization problem associated with the proposed system is defined as

follows:

$$\begin{aligned}\min_{\mathbf{X}} \sum_{k\in W_i} \Big\{ & w_g \, \mathbf{r_G}(t_k) + w_{\mathcal{S}^2} \|\mathbf{r}_{\mathcal{S}^2}(t_k)\|^2 + w_\omega \|\mathbf{r}_\omega(t_k)\|^2 + \\ & w_L \|\mathbf{r_L}(t_k)\|^2 + w_R \cdot \rho_r(\|\mathbf{r_R}(t_k)\|^2) \Big\} \\ & + w_b \|\mathbf{r}_{\mathbf{b}i}\|^2 + w_p \|\mathbf{r}_{\mathbf{p}_i}\|^2 + w_e \|\mathbf{r}_{\mathbf{e}i}\|^2, \\ \text{s.t.} \quad & \|\mathbf{g}^{\mathtt{I}_i}\| = 9.81,\end{aligned} \tag{28}$$

where $\rho_r$ is the Cauchy loss function, employed to mitigate the influence of non-static object-oriented radar points. The bias-prior factor and the end-tail factor, which are not addressed in Section §4, are defined as follows:

$$\begin{aligned}\mathbf{r}_{\mathbf{b}i} &= \begin{bmatrix} \mathbf{b}_{a_i} - \mathbf{b}_{a_{i-1}} \\ \mathbf{b}_{v_i} - \mathbf{b}_{v_{i-1}} \end{bmatrix} \\ \mathbf{r}_{\mathbf{e}i} &= \begin{bmatrix} \mathbf{v}^{\mathtt{I}_{m-1}}_{\mathtt{I}_{m-1}} - 2\cdot\mathbf{v}^{\mathtt{I}_{m-2}}_{\mathtt{I}_{m-2}} + \mathbf{v}^{\mathtt{I}_{m-3}}_{\mathtt{I}_{m-3}} \\ \mathrm{Log}\left(\left(\mathbf{R}^{\mathtt{G}}_{\mathtt{I}_{1-2}}\right)^\top \mathbf{R}^{\mathtt{G}}_{\mathtt{I}_{1-3}} \left(\mathbf{R}^{\mathtt{G}}_{\mathtt{I}_{1-2}}\right)^\top \mathbf{R}^{\mathtt{G}}_{\mathtt{I}_{1-1}}\right) \end{bmatrix}.\end{aligned} \tag{29}$$

Dependencies between sensor measurements and each spline during the optimization of (28) are illustrated in Figure 6. To mitigate the contact-induced noise of IMU acceleration degrading the velocity spline, we decoupled the velocity spline from IMU. Instead, we leverage continuous-time ego-centric velocity spline generated from SoC-radar and leg kinematics measurements, improving the robustness of local gravity estimation compared with prior methods. Within each window $W_i$, the biases $b_{v_i}, b_{a_i}$ are assumed to remain constant, and an end-tail factor minimizes endpoint instability of the splines (Chen et al. 2024).

*5.2.2 Marginalization* At each optimization step, we marginalize out the states and measurements that moved outside of the sliding window, summarizing them into a single prior factor. By reusing historical constraints in this condensed form, the estimator retains observability of active variables while bounding the graph size, achieving computational efficiency for real-time performance. By linearizing all factors about the current best estimate of the $i$-th window, we obtain an equation as follows:

$$\begin{bmatrix} \mathbf{H}_{\alpha\alpha} & \mathbf{H}_{\alpha\beta} \\ \mathbf{H}_{\beta\alpha} & \mathbf{H}_{\beta\beta} \end{bmatrix} \begin{bmatrix} \mathbf{x}_\alpha \\ \mathbf{x}_\beta \end{bmatrix} = \begin{bmatrix} \mathbf{b}_\alpha \\ \mathbf{b}_\beta \end{bmatrix}, \tag{30}$$

where $\mathbf{x}_\beta$ represents the states which are marginalized out, while $\mathbf{x}_\alpha$ indicates the states retained in the optimization. By applying the Schur complement (Sibley et al. 2010), we reformulate the equation as follows:

$$(\mathbf{H}_{\alpha\alpha} - \mathbf{H}_{\alpha\beta}\mathbf{H}^{-1}_{\beta\beta}\mathbf{H}_{\beta\alpha}) \cdot \mathbf{x}_\alpha = \mathbf{b}_\alpha - \mathbf{H}_{\alpha\beta}\mathbf{H}^{-1}_{\beta\beta}\mathbf{b}_\beta, \tag{31}$$

leading to the marginalization-prior factor $\mathbf{r_p}$ as:

$$\begin{aligned}\mathbf{r}_{\mathbf{p}_i} = (\mathbf{H}_{\alpha\alpha} - \mathbf{H}_{\alpha\beta}\mathbf{H}^{-1}_{\beta\beta}\mathbf{H}_{\beta\alpha}) \cdot (\mathbf{x}_{\alpha,i} - \mathbf{x}_{\alpha,i-1}) \\ -(\mathbf{b}_\alpha - \mathbf{H}_{\alpha\beta}\mathbf{H}^{-1}_{\beta\beta}\mathbf{b}_\beta).\end{aligned} \tag{32}$$

*5.2.3 Post Optimization* After incorporating the optimized local gravity state from (28), we refine the rotation states of the relevant static control points. This procedure is applied to the control points marginalized in the previous optimization window, as shown in Figure 5. In this step, we update only the rotation states while keeping others fixed:

$$\begin{aligned}\min_{\mathbf{x_R}} \quad & \sum \|\mathbf{r_{post}}(k)\|^2, \\ \text{where} \quad & \mathbf{r_{post}}(k) = \mathbf{R}^{\mathtt{G}}_{\mathtt{I}_k}\mathbf{g}_{\mathtt{I}_k} - \mathbf{g}_{\mathtt{G}}.\end{aligned} \tag{33}$$

Since the global $z$-axis is aligned with gravity, the above equation provides information only about roll and pitch but not yaw; rotation about the gravity axis remains unobservable. After the post-optimization step, the robot's pose is estimated using dead reckoning based on the optimized states. The ego-centric velocity is first transformed into the global frame, and subsequently, the robot position is computed using the following equation:

$$\mathbf{p}(t_k) = \mathbf{p}(t_{k-1}) + \int_{t\in[t_{k-1},t_k)} \mathbf{R}(t)\cdot\mathbf{v}(t)dt. \tag{34}$$

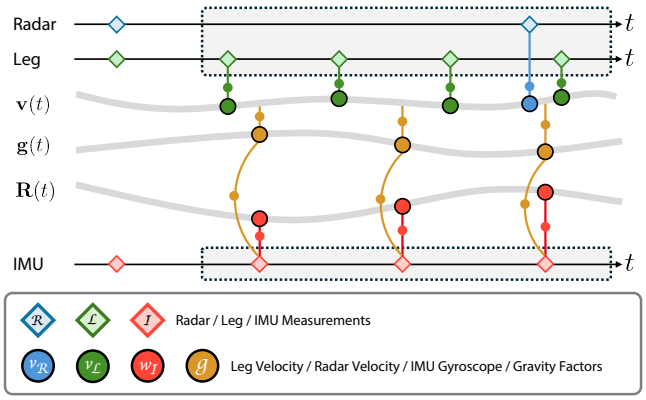



**Figure 6.** Relationship between sensor data and splines during incremental optimization. Gray-shaded boxes indicate sensor measurements active within the sliding optimization window; colored lines denote inter-spline and measurement-spline dependencies. In GaRLILEO, velocity is decoupled from the IMU, constructing a continuous-time ego-velocity spline from radar and leg-kinematics measurements. This decoupling is particularly advantageous for legged robots, where ground contact induces noisy accelerations on IMU.

## 6 Radar-Leg-IMU Dataset

### 6.1 System Configuration

The overall sensor configuration and the coordinate frames of each sensor are illustrated in Figure 7, while detailed specifications for each sensor are provided in Table 1. Two different data acquisition setups are employed in this work, both built on Spot, a quadrupedal robot from Boston Dynamics, which provides joint encoder and contact sensor data at 150 Hz. An IWR1843BOOST mmWave radar module captures 4D radar point clouds with a maximum range of 11 m and a range resolution of 4.8 cm, while a 3DM-GV7-AHRS IMU from Microstrain operates at 100 Hz to provide inertial measurements including onboard AHRS orientation measurement. Both systems share the same radar and IMU sensor models.

*SNU System* The SNU system, used for experiments at Seoul National University, collects data in diverse conditions, including both indoor and outdoor environments. To obtain a baseline trajectory, an OS1-32 LiDAR (maximum range 150 m) is mounted, and a TLS-generated

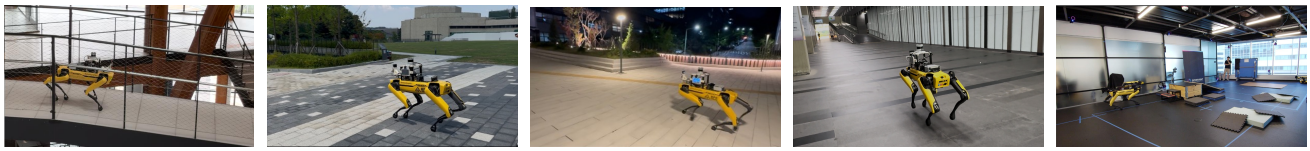

**Figure 7.** SNU and RAI sensor system deployment. Both systems include the same TI-mmWave radar, Microstrain IMU, and Boston Dynamics Spot quadrupedal robot, but are attached with different extrinsics.

**Table 1.** The Sensor Specifications.

| Sensor | Manufacture | Model | Topic name | Frequency | Description |
|---|---|---|---|---|---|
| Legged robot | Boston Dynamics | Spot | /joint_states | 150 Hz | Sensor Measurements |
| | | | /spot/status/feet | 150 Hz | |
| Radar | Texas Instruments | IWR1843BOOST | /ti_mmwave/radar_scan_pcl_0 | 20 Hz | |
| IMU | MicroStrain | 3DM-GV7-AHRS | /imu | 100 Hz | |
| LiDAR | Ouster | OS1-32 | /ouster/points | 10 Hz | Ground-truth Reference |
| Laser scanner | Leica | RTC360 | - | - | |

**Table 2.** The Description for Each Sequence.

| Sequence | Path Length (m) | Elevation Change (m) | Duration (s) | Outdoor | Indoor | Stair | Slope | # of Loop |
|---|---|---|---|---|---|---|---|---|
| `Atrium` | 109.93 | - | 124.50 | ✗ | ✓ | ✗ | ✗ | 1 |
| `BridgeLoop` | 161.17 | 1.72 | 187.20 | ✗ | ✓ | ✓ | ✓ | 3 |
| `CorriLoop` | 208.68 | - | 229.40 | ✗ | ✓ | ✗ | ✗ | 2 |
| `BiCorridor` | 240.82 | 4.72 | 277.29 | ✗ | ✓ | ✓ | ✗ | 1 |
| `Downstair` | 233.75 | 8.81 | 270.90 | ✗ | ✓ | ✓ | ✗ | 2 |
| `Upstair` | 197.22 | 9.37 | 227.89 | ✗ | ✓ | ✓ | ✓ | 1 |
| `SlopeStair` | 273.37 | 10.06 | 307.49 | ✓ | ✓ | ✓ | ✓ | 1 |
| `Overpass` | 169.17 | 7.23 | 213.49 | ✓ | ✗ | ✓ | ✗ | 1 |
| `Tunnel` | 247.94 | - | 277.00 | ✓ | ✓ | ✗ | ✗ | 1 |
| `Quad` | 447.83 | 10.72 | 503.69 | ✓ | ✗ | ✓ | ✓ | 1 |
| `MoCap-E` | 44.91 | 0.57 | 139.10 | ✗ | ✓ | ✓ | ✓ | 2 |
| `MoCap-H` | 42.48 | 0.60 | 79.46 | ✗ | ✓ | ✓ | ✓ | 2 |



(a) `Upstair` (b) `Quad` (c) `Overpass` (d) `Tunnel` (e) `MoCap`

**Figure 8.** Environmental examples of the acquired sequences. Diverse environments are included in each sequence to consider various situations that the quadrupedal robot may encounter in a real-world mission.

map is used as ground-truth reference (see Section §6.4). Data acquisition and logging are performed onboard using the Spot CORE, equipped with an Intel 8th Gen i5 processor and 16 GB of DDR4 RAM.

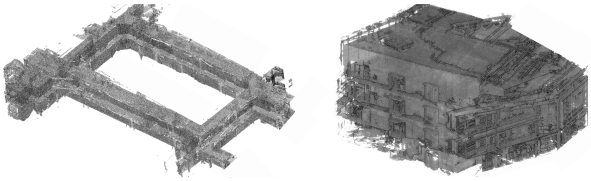
*RAI System* The RAI system, used for experiments at the Robotics and AI Institute, operates entirely within an indoor motion-capture environment, providing a controlled experimental setup. While sharing the radar and IMU configurations of the SNU system, the LiDAR is omitted, and ground truth is obtained from the motion capture system. Data is acquired and logged onboard with an NVIDIA Jetson AGX Orin, including a 12-core Arm Cortex-v8.2 CPU and 64 GB of LPDDR5 RAM.

## 6.2 *Details of Each Sequence*

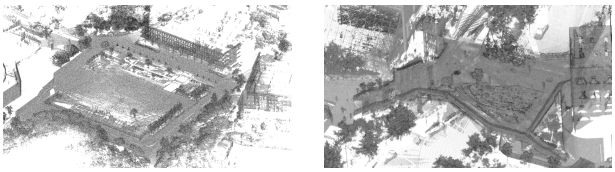
An overview of the twelve sequences is given in Table 2, and their environments are illustrated in Figure 8; details of each sequence are as follows:

- Atrium: A large flat floor indoor atrium with floor-to-ceiling glass and a high ceiling, where the expansive open floor and long glass facades induce strong specular reflections, multipath, and low parallax on radar sensor.
- BridgeLoop: An indoor sequence with three repetitions that traverses pedestrian bridges and short stair segments around an open atrium, stressing robustness in wide, low-parallax spaces.

- CorriLoop: A narrow, rectangular indoor corridor traversed twice; long straight segments with repeating doors/walls and a glossy floor create perceptual aliasing, while four sharp 90° turns stress turn handling and loop consistency.
- BiCorridor: A two-level corridor running in the same building as `CorriLoop`. One loop on the first floor, then a stair ascent and a second loop on the adjacent floor in the reverse direction. The narrow rectangular hallways contain long, low-parallax segments with repeating doors/walls that induce geometric degeneracy and perceptual aliasing, while the stairs introduce vertical motion and contact disturbances. This sequence stresses robustness to aliasing, direction reversal, and floor-to-floor consistency in constrained indoor spaces.
- Downstair: A multi-floor indoor sequence that starts on the second level with a long rectangular loop, descends one flight to traverse an extended straight corridor, then descends again to finish with a smaller rectangular loop. Wide, glossy hallways and two prolonged downward staircases stress perception during extended descent, floor-to-floor transitions, and low-parallax segments.
- Upstair: An indoor ascent sequence in the same building as `BridgeLoop`, climbing three floors by alternately traversing pedestrian bridges and short stair flights. The sequence includes upward motion—yaw changes occur between the bridge and stair segments—stressing vertical translation, stair negotiation, and transitions across landings in a wide, low-parallax space.
- SlopeStair: A mixed indoor–outdoor traverse with a long downhill ramp, multiple upstair flights, and doorway/corridor transitions, stressing robustness to large elevation changes, lighting shifts, and abrupt structural/surface variations.
- Overpass: This sequence takes place on outdoor stairs and a pedestrian overpass, with lamps and reflective paving. It stresses robustness to open-air transitions and repeated elevation changes across steps, ramps, and long sidewalk segments.
- Tunnel: This sequence traverses a long, semi-open tunnel lined with glass façades and repetitive concrete pillars. The path includes gentle ramps and a tight U-turn, stressing robustness under feature-degenerate geometry and illumination changes.
- Quad: This sequence traverses an outdoor campus quad with broad paved plazas, tiled walkways, curbs, and stairs. It stresses robustness to feature-sparse open areas and repetitive textures while handling stair climbing, ramps, and outdoor slopes.
- MoCap-E: This sequence has two loops in the indoor Motion Capture (MoCap) room, each including a short stair–slope vertical motion zone and a cushion zone with two soft cushions that disturb leg odometry. In the second loop, a folded box at the slippery zone is deliberately dragged, breaking the assumption of a stationary floor and causing a marked discrepancy between leg kinematics and other sensors.



(a) Indoor Sequences. Left: `BiCorridor`, Right: `Upstair`



(b) Outdoor Sequences. Left: `Quad`, Right: `Overpass`

**Figure 9.** Examples of ground truth TLS map on SNU sequences. Leveraged for generating ground truth trajectory.

- MoCap-H: This sequence shares the same layout as `MoCap-E` but is run at a higher speed. In the slippery zone, the robot even moves forward while the floor shifts backward due to dragging, further violating the stationary floor assumption and amplifying the leg kinematics disagreement.

## 6.3 Extrinsic Calibration of Sensor Systems

Extrinsic calibration between Spot, radar, and IMU is required for accurate odometry estimation. On both sensor systems, we leveraged the CAD model of each system to acquire the exact extrinsic parameters between the sensors. As included in Figure 7, the radar attached to the RAI system is slightly biased to the right side of the robot, and the IMU is attached perpendicularly compared with the SNU system. For more detailed information about the extrinsic calibration parameter, please refer to the project homepage.

## 6.4 Ground Truth Trajectory Generation

*6.4.1 SNU Sequences* Accurate 6-DoF ground truth poses are essential for evaluating various robotic tasks, including state estimation and SLAM. Unlike previous studies, our deployments span both indoor and outdoor environments over several hundred meters, requiring millimeter-level precision. These stringent requirements render traditional ground truth sources, such as LiDAR-IMU-based references (Jung et al. 2024) and RTK-GNSS systems (Geiger et al. 2013; Barnes et al. 2020; Kim et al. 2025a), unsuitable. While MoCap systems can provide high-frequency and high-precision pose estimates, their limited workspace makes them impractical for large-scale deployments (Doer and Trommer 2021). Some prior works (Tranzatto et al. 2022b) utilize survey-grade maps as ground truth references by performing scan-to-map matching of deskewed LiDAR points using synchronized inertial and ranging sensors. However, for legged robots, continuous dynamic motion during data acquisition significantly degrades LiDAR deskewing and motion estimation accuracy.

To address this, we adopt the approach proposed in (Hu et al. 2024), which combines FAST-LIO2 (Xu et al. 2022) odometry and loop closure factors with a degeneration-aware map factor derived from dense prior maps. As

illustrated in Figure 9, prior maps are collected using a Leica RTC360. This graph-based formulation enables accurate pose estimation even in degenerate and stationary conditions, thereby providing reliable ground truth for our evaluation.

Because this ground truth framework leverages FAST-LIO2 (Xu et al. 2022), a tightly coupled LiDAR-IMU SLAM as an odometry front-end, precise extrinsic calibration between those two sensors $\mathbf{T}_I^L$ is essential. We perform this extrinsic calibration using the robust method of Zhu et al. (2022), which applies diverse rotational motions across all three axes to accumulate sufficient motion data, allowing rapid and accurate estimation without initial parameter guesses. The method directly computes spatial and temporal offsets from unsynchronized data, yielding high-precision LiDAR-IMU extrinsics required for our ground-truth estimation.

*6.4.2 RAI Sequences* For MoCap sequences acquired using the RAI sensor system, we utilized the MoCap system to obtain a highly precise ground truth trajectory. As these sequences were collected exclusively in a controlled indoor environment, the MoCap odometry could be reliably used as reference. The experiment was conducted in a $13.5\ \mathrm{m} \times 5\ \mathrm{m} \times 3\ \mathrm{m}$ motion-capture room equipped with 20 Vicon Valkyrie VK 16 cameras, ensuring complete coverage of the space. The system streamed motion capture data at $120\ \mathrm{Hz}$ using Vicon Tracker 4.3 software.

## 7 Experiment Results

In this section, we evaluate the performance of GaRLILEO against SOTA odometry algorithms that utilize SoC radar, IMU, and leg kinematics. The experiments are conducted on a self-collected real-world dataset.

Odometry accuracy is assessed using the root mean square error (RMSE) of Absolute Pose Error (APE) and Relative Pose Error (RPE), each decomposed into translational and rotational components. The units are as follows: $\mathrm{APE}_t$ $(\mathrm{m})$, $\mathrm{APE}_r$ $(^\circ)$, $\mathrm{RPE}_t$ $(\mathrm{m/m})$, and $\mathrm{RPE}_r$ $(^\circ/\mathrm{m})$. To specifically evaluate vertical drift in odometry, we report the z-axis APE ($\mathrm{APE}_z$). All evaluations are performed using the Evo Trajectory Evaluator (Grupp 2017), a widely adopted open-source toolkit for odometry benchmarking in robotics.

The comparative analysis is organized by the baselines' sensor configurations, while GaRLILEO is evaluated in its full configuration to report system-level performance, i.e., the benefit of fusing leg kinematics, radar, and IMU. First, we compare GaRLILEO against recent SoC radar-IMU odometry methods, including Co-RaL (Jung et al. 2024), which additionally integrates the leg kinematics velocity factor. Next, we also evaluate GaRLILEO against open-source leg kinematics-IMU fusion odometry methods. Every parameter is adopted from the official implementation, except that the robot-specific parameters of legged robots are modified based on the official Unified Robot Description Format (URDF) file of Boston Dynamics SPOT. Detailed descriptions of the baseline algorithms and comprehensive evaluation results are provided in the following subsections.

After comparing GaRLILEO directly with the baselines, we conducted detailed ablation studies on the modules of GaRLILEO. Specifically, we analyzed the complementary effect between radar and leg kinematics, the contribution of gravity factors to both local gravity vector estimation and odometry, and the impact of the velocity bias term.

### 7.1 Radar-IMU Odometry Comparison

In this subsection, we compare the performance of GaRLILEO with that of five recent SoC Radar-Inertial Odometry (RIO) methods. The baseline methods are listed as follows:

- **River** (Chen et al. 2024): A B-spline-based continuous velocity estimator that fuses SoC radar and IMU, employing dead reckoning to compute full odometry.
- **Co-RaL** (Jung et al. 2024): A cooperative odometry algorithm integrating SoC radar, IMU, and leg kinematics velocity, designed to operate robustly across diverse environments.
- **EKF-RIO** (Doer and Trommer 2020): An EKF-based odometry method that fuses SoC radar and IMU data.
- **DeRO** (Do et al. 2024): A dead reckoning based RIO method that combines SoC radar ego-velocity and gyroscope data using an IEKF, with accelerometer-based tilt angle estimation.

We evaluate the performance of GaRLILEO and baseline methods, with a focus on odometry accuracy in diverse environments. Table 3 presents quantitative results for APE and RPE metrics. Below, we elaborate on the results for each sequence. Some sample trajectories and qualitative evaluation are illustrated in Figure 10. For both quantitative and qualitative analysis, GaRLILEO achieves superior odometry accuracy across most metrics, particularly excelling in vertical accuracy ($\mathrm{APE}_z$).

*1) Atrium:* We first evaluate methods on the `Atrium` sequence, a flat environment with a single loop, minimal vertical motion, where contact-induced drift is negligible. River and DeRO, which rely on dead reckoning, show large $\mathrm{APE}_t$ and $\mathrm{APE}_z$ due to insufficient drift mitigation from the high vibration of the legged robot UGV. EKF-RIO achieves improved accuracy on most metrics by tightly fusing SoC radar–derived ego-velocity with IMU measurements via an EKF-based approach. However, it exhibits a higher $\mathrm{APE}_r$ than DeRO. Co-RaL, which integrates the leg kinematics velocity preintegration factor, achieves the second-lowest $\mathrm{APE}_z$ among the methods. GaRLILEO, leveraging accurate local gravity estimation, achieves an $\mathrm{APE}_z$ only about 5% of Co-RaL's while also notably reducing errors in every other metric, making a big gap with baselines.

*2) BridgeLoop:* We next assess the `BridgeLoop`, where the robot completes triple loops in a consistent rotational direction while going over gentle slope bridges and short stairs multiple times. In `BridgeLoop`, noise in roll and pitch estimation from contact uncertainty leads to substantial vertical drift, as reflected in the elevated $\mathrm{APE}_z$ of River and DeRO. EKF-RIO suffers from significantly high $\mathrm{APE}_r$ due to the limited orientation observability of IMU-only fusion, which becomes more severe in multi-loop trajectories. In contrast, Co-RaL achieves the second lowest $\mathrm{APE}_r$ via the 4-DoF radar factor, enhancing both translational and rotational accuracy. GaRLILEO provides the most robust

**Table 3.** Evaluation on radar-based methods. **Bold** numbers indicate the smallest error, and underlined numbers denote the second smallest error within each metric.

| Sequence | RMSE | Ours | Radar based method | | | | Sequence | RMSE | Ours | Radar based method | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | River | Co-RaL | EKF-RIO | DeRO | | | | River | Co-RaL | EKF-RIO | DeRO |
| Atrium | $APE_t$ | **0.816** | 15.473 | 2.606 | 6.679 | 15.127 | SlopeStair | $APE_t$ | **2.359** | 38.438 | 8.222 | - | - |
| | $APE_r$ | **1.715** | 25.154 | 4.393 | 12.496 | 9.602 | | $APE_r$ | **2.805** | 49.587 | 9.954 | - | - |
| | $RPE_t$ | **0.055** | 0.213 | 0.132 | 0.162 | 0.203 | | $RPE_t$ | **0.061** | 0.248 | 0.203 | - | - |
| | $RPE_r$ | **0.554** | 0.942 | 0.780 | 0.722 | 1.572 | | $RPE_r$ | 1.051 | 1.049 | **0.920** | - | - |
| | $APE_z$ | **0.106** | 12.508 | 2.225 | 4.908 | 15.072 | | $APE_z$ | **0.665** | 24.701 | 2.789 | - | - |
| BridgeLoop | $APE_t$ | **1.193** | 16.930 | 4.562 | 7.125 | 19.793 | Overpass | $APE_t$ | **1.526** | 11.210 | 8.486 | - | - |
| | $APE_r$ | **2.719** | 108.904 | 7.959 | 30.850 | 14.468 | | $APE_r$ | **4.043** | 25.574 | 17.416 | - | - |
| | $RPE_t$ | **0.080** | 0.183 | 0.149 | 0.132 | 0.169 | | $RPE_t$ | **0.091** | 0.281 | 0.230 | - | - |
| | $RPE_r$ | 1.058 | 2.159 | 1.022 | **0.936** | 1.694 | | $RPE_r$ | 1.227 | 1.136 | **0.952** | - | - |
| | $APE_z$ | **0.117** | 13.422 | 4.131 | 5.302 | 19.728 | | $APE_z$ | **1.074** | 8.034 | 7.728 | - | - |
| CorriLoop | $APE_t$ | **1.627** | 28.885 | 6.576 | 9.283 | 18.049 | Tunnel | $APE_t$ | **3.523** | 29.515 | 7.939 | 26.800 | 40.231 |
| | $APE_r$ | **5.676** | 45.570 | 6.245 | 28.555 | 15.863 | | $APE_r$ | **2.849** | 31.844 | 8.362 | 33.427 | 31.507 |
| | $RPE_t$ | **0.066** | 0.245 | 0.129 | 0.180 | 0.178 | | $RPE_t$ | **0.083** | 0.196 | 0.262 | 0.316 | 0.196 |
| | $RPE_r$ | **0.738** | 1.002 | 0.944 | 0.884 | 0.902 | | $RPE_r$ | **0.440** | 0.752 | 0.638 | 0.541 | 0.683 |
| | $APE_z$ | **0.168** | 25.794 | 5.846 | 4.001 | 17.595 | | $APE_z$ | **0.548** | 18.991 | 2.872 | 6.139 | 36.107 |
| BiCorridor | $APE_t$ | **1.425** | 55.161 | 7.631 | 10.293 | 23.500 | Quad | $APE_t$ | **7.347** | - | 14.395 | - | - |
| | $APE_r$ | **5.519** | 97.208 | 7.232 | 27.697 | 13.141 | | $APE_r$ | **3.356** | - | 11.615 | - | - |
| | $RPE_t$ | **0.063** | 0.252 | 0.114 | 0.185 | 0.220 | | $RPE_t$ | **0.080** | - | 0.298 | - | - |
| | $RPE_r$ | 0.885 | 2.371 | **0.799** | 0.838 | 0.840 | | $RPE_r$ | 0.838 | - | **0.532** | - | - |
| | $APE_z$ | **0.226** | 30.646 | 7.087 | 5.497 | 23.003 | | $APE_z$ | **2.143** | - | 3.478 | - | - |
| Downstair | $APE_t$ | **3.916** | 66.838 | 8.250 | 30.835 | 29.710 | MoCap-E | $APE_t$ | **0.852** | 0.916 | - | 2.156 | 3.305 |
| | $APE_r$ | **3.415** | 111.422 | 8.029 | 36.447 | 30.487 | | $APE_r$ | **3.373** | 5.738 | - | 6.146 | 33.163 |
| | $RPE_t$ | **0.099** | 0.180 | 0.111 | 0.229 | 0.166 | | $RPE_t$ | **0.184** | 0.239 | - | 0.328 | 0.373 |
| | $RPE_r$ | 1.080 | 1.304 | 0.940 | **0.875** | 1.619 | | $RPE_r$ | **1.508** | 1.922 | - | 2.492 | 11.822 |
| | $APE_z$ | **0.530** | 14.934 | 6.096 | 4.251 | 25.466 | | $APE_z$ | **0.103** | 0.236 | - | 1.776 | 2.555 |
| Upstair | $APE_t$ | **1.496** | 20.198 | 7.887 | 10.044 | 31.236 | MoCap-H | $APE_t$ | **0.880** | 1.145 | 2.889 | 2.121 | 6.626 |
| | $APE_r$ | **4.048** | 14.445 | 7.626 | 28.159 | 13.299 | | $APE_r$ | 1.981 | 8.814 | **1.501** | 7.416 | 66.105 |
| | $RPE_t$ | **0.071** | 0.198 | 0.143 | 0.143 | 0.195 | | $RPE_t$ | **0.204** | 0.250 | 0.318 | 0.325 | 0.513 |
| | $RPE_r$ | 0.933 | 0.906 | 1.015 | **0.808** | 2.119 | | $RPE_r$ | 1.665 | 2.219 | **0.686** | 1.995 | 27.176 |
| | $APE_z$ | **0.514** | 19.964 | 7.317 | 8.871 | 31.169 | | $APE_z$ | **0.118** | 0.450 | 2.404 | 1.646 | 4.782 |

pose estimates, presenting the lowest errors in every metric except $RPE_r$, while $RPE_r$ is at a similar level to the second-best baseline, Co-RaL. Notably, the vertical drift $APE_z$ is almost identical to the flat sequence `Atrium`, expressing the vertical robustness of GaRLILEO on multiple turns.

*3) CorriLoop:* The `CorriLoop` sequence, consisting of two loops on flat indoor terrain, shows similar trends. While River and EKF-RIO suffer from significant rotational errors, GaRLILEO effectively suppresses them through accurate roll and pitch estimation. Although Co-RaL achieves comparable $APE_r$ due to the radar factor, their $APE_z$ remains about 35× higher than GaRLILEO's, reflecting limited roll and pitch observability.

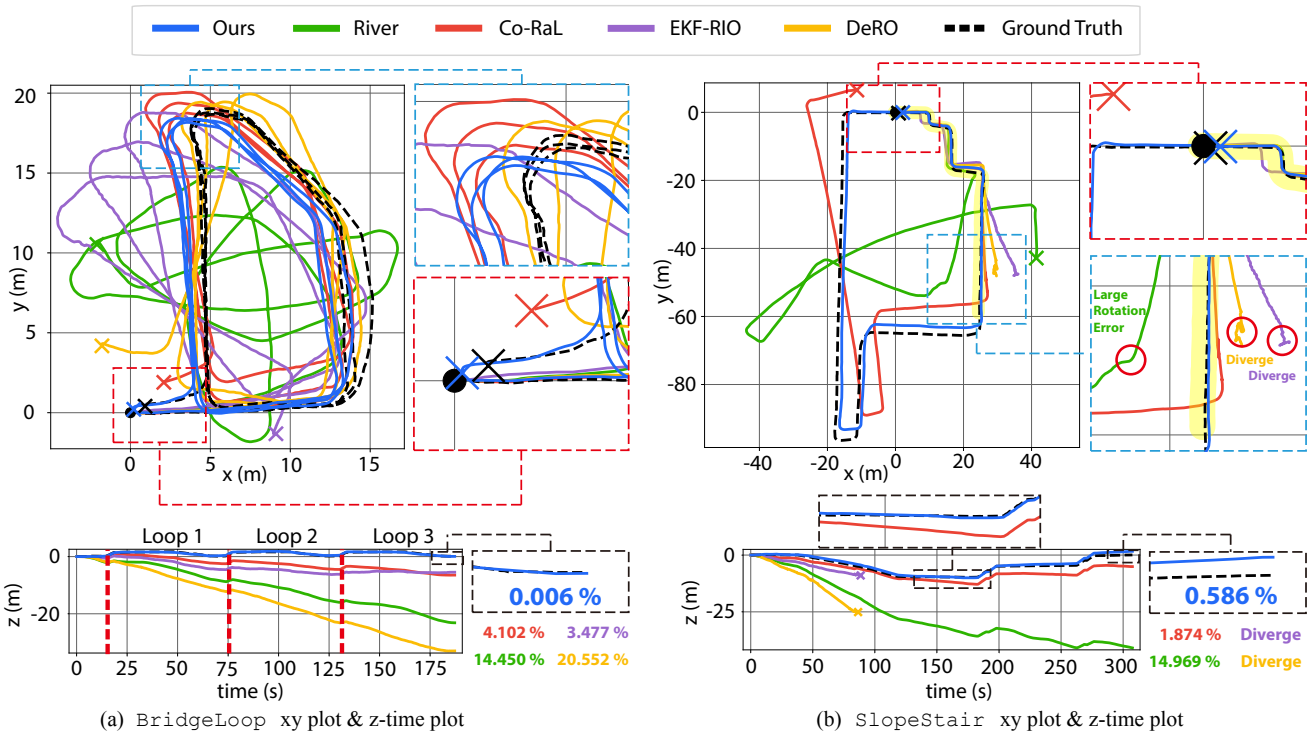
*4) BiCorridor:* The `BiCorridor` sequence combines a loop with stair ascents, traversing two floors with opposite rotational directions. Despite environmental similarities to `CorriLoop`, the sharp rotations and stair climbs substantially degrade odometry accuracy, particularly along the vertical axis. This results in high $APE_t$ and $APE_z$ for simple dead-reckoning methods, such as River and DeRO. Interestingly, EKF-RIO achieves lower $APE_t$ than Co-RaL, but suffers from much higher $APE_r$. In contrast, GaRLILEO effectively addresses these challenges by continuously fusing leg kinematics, IMU, and SoC radar, maintaining sub-meter $APE_z$.

*5) Downstair:* In the `Downstair` sequence, only GaRLILEO and Co-RaL converge successfully, while River, EKF-RIO, and DeRO return erroneous odometry results due to noisy IMU measurements caused by rapid and repetitive impacts during stair descent. Among the convergent methods, GaRLILEO and Co-RaL achieve the lowest $APE_t$, demonstrating stable and precise estimation. With its local gravity factor, GaRLILEO achieves sub-meter $APE_z$, which is lower than 10% of the second best result from Co-RaL, significantly outperforming all prior radar–IMU methods.

*6) Upstair:* The `Upstair` sequence, which involves multiple stair ascents and a sloped bridge in an open space, exhibits similar trends. Methods without leg kinematics input, such as River and EKF-RIO, suffer from pronounced vertical drift due to the lack of leg kinematics constraints. Because this sequence consists of the same directional looped trajectories with stair climbing in each loop, EKF-RIO behaves similarly to the `BridgeLoop` and `CorriLoop` sequences. The second-best result, Co-RaL, shows vertical errors nearly 20× larger than GaRLILEO, underscoring the effectiveness of our continuous-time local gravity estimation for precise roll and pitch correction.

*7) SlopeStair:* As the system moves from indoors to outdoors and back in during this sequence, severe drift at the corner before re-entry causes EKF-RIO and DeRO to diverge. This highlights the limitations of SoC radar–IMU-only systems during indoor/outdoor transitions. Although River converges, their vertical errors ($APE_z$) remain significantly higher than those of methods incorporating leg kinematics, highlighting the importance of kinematics sensing for stable and accurate odometry. Co-RaL, fusing leg kinematics with radar–IMU data, achieves the second-best performance across most metrics, demonstrating the benefits of multimodal integration. Finally, GaRLILEO delivers the most accurate odometry overall, achieving sub-meter-level vertical accuracy.

*8) Overpass:* In the `Overpass` sequence, staircases appear before and after the overpass; when the robot is on the stairs, fewer radar returns are observed because the sensor




(a) `BridgeLoop` xy plot & z-time plot (b) `SlopeStair` xy plot & z-time plot

**Figure 10.** Radar–Inertial baseline odometry results on the (a) `BridgeLoop` and (b) `SlopeStair` sequences. Red dotted lines in (a, bottom) indicate the starting points of each loop in `BridgeLoop` sequence, while the faint yellow region in (b, top) denotes the outdoor segment of the `SlopeStair` sequence. The red zoomed-in views highlight that GaRLILEO and Co-RaL—both integrating SoC radar and leg kinematics—converge more closely to the ground-truth final position. The colored numbers on the right-hand side of the bottom plots show relative start-to-end vertical drift with respect to the full path length, where GaRLILEO achieves substantially lower drift. The cyan zoomed view in (a) illustrates that GaRLILEO (blue) follows the most consistent turning trajectory across repeated loops. In (b), the red circles indicate the failure points of baseline odometries during outdoor-to-indoor transitions, whereas radar–leg fused odometries, GaRLILEO, and Co-RaL maintain robust estimation. Both subfigures qualitatively illustrate the odometry accuracy of GaRLILEO, particularly in suppressing vertical drift.

is angled skyward. Similar to the `SlopeStair` sequence, EKF-RIO and DeRO fail to converge, underscoring their limitations in handling sharp turns in open environments. River achieves reduced vertical drift ($\text{APE}_z$) compared to the `SlopeStair` sequence. This improvement, however, is largely attributed to the shorter trajectory and smaller elevation change of the `Overpass` sequence itself. Co-RaL demonstrates slightly better accuracy than River due to the additional fusion of leg kinematics, though the improvement is marginal. In contrast, GaRLILEO robustly manages outdoor contact-induced drift, reducing vertical error to lower than 15% and achieving substantial improvement in overall APE compared to all baselines.

*9) Tunnel:* In the `Tunnel` sequence, which features a long passage with a sharp U-turn, GaRLILEO maintains sub-meter $\text{APE}_z$ and achieves the lowest error on every metric. Co-RaL, benefiting from radar–leg kinematics fusion, ranks second. River and DeRO exhibit severe vertical drift due to contact impacts and limited observability. EKF-RIO shows less drift than these methods, but still larger vertical errors than leg kinematics-fused methods.

*10) Quad:* The `Quad` sequence, the longest and most challenging dataset, includes stairs, slopes, and substantial elevation changes. Frequent contact drift, multiple dynamic objects, outdoor stairs and slopes, and a reduced number of radar points make accurate pose estimation particularly difficult. River, EKF-RIO, and DeRO fail to converge due to sparse radar returns. Due to its factor graph framework that adaptively relies on the more reliable sensor modality, Co-RaL succeeds in converging and even outperforms GaRLILEO on $\text{RPE}_r$. Nevertheless, GaRLILEO achieves a notably lower error on all other metrics through precise local gravity estimation, while exhibiting slightly higher but competitive $\text{RPE}_r$ compared to Co-RaL. These results highlight that GaRLILEO consistently provides robust and accurate pose estimation across diverse environments, owing to its local gravity model.

*11) MoCap-E:* Both `MoCap` sequences include two challenging test scenarios: (*i*) a slippery zone with a backward-moving floor in the second loop and (*ii*) a cushion zone included in both loops. These environments are designed to degrade leg odometry. In the slippery zone, as the contact frame continuously moves, leg kinematics produce erroneous horizontal velocity estimates. In the cushion zone, where the contact frame moves downward while the contact sensor remains active, leg kinematics yield incorrect vertical velocity estimates.

In `MoCap-E`, Co-RaL fails to converge due to discrepancies between leg kinematics and radar. EKF-RIO and DeRO achieve similar $\text{APE}_t$ and $\text{APE}_z$, though DeRO exhibits higher $\text{APE}_r$ and $\text{RPE}_r$ since it relies solely on IMU-based orientation estimation. River achieves the second-best performance, showing its potential in controlled indoor environments. Still, GaRLILEO delivers the most accurate

**Table 4.** Evaluation on Leg Kinematic based methods

| Sequence | RMSE | Ours | Leg Kinematic based method | | | | Sequence | RMSE | Ours | Leg Kinematic based method | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Pronto | MUSE | DRIFT | Holistic | | | | Pronto | MUSE | DRIFT | Holistic |
| Atrium | $APE_t$ | **0.816** | 7.849 | 3.297 | 7.228 | 7.407 | SlopeStair | $APE_t$ | **2.359** | - | 35.519 | 20.004 | 30.420 |
| | $APE_r$ | **1.715** | 21.448 | 13.597 | 5.225 | 17.631 | | $APE_r$ | **2.805** | - | 54.144 | 12.821 | 40.790 |
| | $RPE_t$ | **0.055** | 0.293 | 0.090 | 0.153 | 0.151 | | $RPE_t$ | **0.061** | - | 0.151 | 0.180 | 0.205 |
| | $RPE_r$ | **0.554** | 0.700 | 0.925 | 0.846 | 1.209 | | $RPE_r$ | **1.051** | - | 1.063 | 1.064 | 1.274 |
| | $APE_z$ | **0.106** | 0.592 | 0.685 | 6.460 | 5.303 | | $APE_z$ | **0.665** | - | 0.967 | 15.480 | 8.427 |
| BridgeLoop | $APE_t$ | **1.193** | 6.684 | 4.195 | 9.350 | 5.037 | Overpass | $APE_t$ | **1.526** | 4.039 | 6.507 | 9.785 | 5.920 |
| | $APE_r$ | **2.719** | 16.739 | 26.513 | 5.175 | 25.208 | | $APE_r$ | **4.043** | 5.367 | 10.373 | 8.268 | 16.034 |
| | $RPE_t$ | **0.080** | 0.256 | 0.200 | 0.212 | 0.162 | | $RPE_t$ | **0.091** | 0.139 | 0.191 | 0.173 | 0.125 |
| | $RPE_r$ | 1.058 | **0.950** | 0.967 | 1.180 | 1.823 | | $RPE_r$ | 1.227 | **0.582** | 0.972 | 1.126 | 1.594 |
| | $APE_z$ | **0.117** | 5.160 | 2.250 | 9.206 | 3.870 | | $APE_z$ | **1.074** | 1.227 | 2.442 | 8.326 | 2.424 |
| CorriLoop | $APE_t$ | **1.627** | 8.988 | 5.811 | 12.379 | 14.780 | Tunnel | $APE_t$ | **3.523** | 6.171 | 10.194 | 18.074 | 28.576 |
| | $APE_r$ | **5.676** | 25.691 | 18.871 | 10.887 | 25.942 | | $APE_r$ | **2.849** | 13.369 | 11.199 | 7.302 | 41.034 |
| | $RPE_t$ | **0.066** | 0.200 | 0.127 | 0.154 | 0.173 | | $RPE_t$ | **0.083** | 0.158 | 0.105 | 0.159 | 0.199 |
| | $RPE_r$ | 0.738 | **0.711** | 1.028 | 0.851 | 1.358 | | $RPE_r$ | **0.440** | 0.482 | 0.500 | 0.744 | 0.903 |
| | $APE_z$ | **0.168** | 1.519 | 2.514 | 11.352 | 12.668 | | $APE_z$ | **0.548** | 4.161 | 5.425 | 15.080 | 9.741 |
| BiCorridor | $APE_t$ | **1.425** | 15.078 | 15.035 | 11.867 | 13.990 | Quad | $APE_t$ | **7.347** | 11.851 | - | 36.969 | 122.381 |
| | $APE_r$ | **5.519** | 44.116 | 51.048 | 7.055 | 38.014 | | $APE_r$ | **3.356** | **6.218** | - | 15.488 | 79.279 |
| | $RPE_t$ | **0.063** | 0.296 | 0.152 | 0.164 | 0.216 | | $RPE_t$ | **0.080** | 0.157 | - | 0.155 | 0.152 |
| | $RPE_r$ | 0.885 | **0.840** | 0.918 | 0.851 | 1.419 | | $RPE_r$ | 0.838 | **0.543** | - | 0.784 | 0.969 |
| | $APE_z$ | **0.226** | 3.236 | 2.880 | 11.389 | 7.868 | | $APE_z$ | **2.143** | 2.924 | - | 21.621 | 4.012 |
| Downstair | $APE_t$ | **3.916** | 11.613 | 8.323 | 13.448 | 30.346 | MoCap-E | $APE_t$ | **0.852** | 1.416 | 1.313 | 4.588 | 2.335 |
| | $APE_r$ | **3.415** | 10.791 | 8.595 | 8.825 | 43.444 | | $APE_r$ | **3.373** | 5.441 | 4.984 | 7.150 | 30.734 |
| | $RPE_t$ | **0.099** | 0.187 | 0.182 | 0.159 | 0.174 | | $RPE_t$ | **0.184** | 0.246 | 0.201 | 0.211 | 0.188 |
| | $RPE_r$ | 1.080 | **0.658** | 1.095 | 0.977 | 1.637 | | $RPE_r$ | **1.508** | 1.844 | 1.921 | 2.166 | 3.534 |
| | $APE_z$ | **0.530** | 2.490 | 4.843 | 11.141 | 5.601 | | $APE_z$ | **0.103** | 1.010 | 0.818 | 4.190 | 0.501 |
| Upstair | $APE_t$ | **1.496** | 6.238 | 5.307 | 10.832 | 7.238 | MoCap-H | $APE_t$ | **0.880** | 1.670 | 0.900 | 4.004 | 2.798 |
| | $APE_r$ | **4.048** | 21.379 | 27.242 | 4.352 | 30.934 | | $APE_r$ | **1.981** | 5.237 | 6.727 | 4.744 | 31.101 |
| | $RPE_t$ | **0.071** | 0.139 | 0.179 | 0.208 | 0.141 | | $RPE_t$ | 0.204 | 0.177 | 0.195 | 0.193 | **0.169** |
| | $RPE_r$ | 0.933 | **0.647** | 0.979 | 1.121 | 1.643 | | $RPE_r$ | **1.665** | 1.945 | 1.917 | 2.225 | 5.109 |
| | $APE_z$ | **0.514** | 4.485 | 3.206 | 10.676 | 5.574 | | $APE_z$ | **0.118** | 1.404 | 0.283 | 3.704 | 0.711 |

odometry, effectively handling leg kinematics failures in specific zones and enhancing overall odometry estimation, especially in the vertical direction, by precisely estimating the gravity vector. This robustness stems from its B-spline-based continuous odometry scheme, which ensures stable trajectory estimation, even in the presence of discrepancies between sensor modalities.

*12) MoCap-H:* In `MoCap-H`, the results follow a similar trend. Because the slippery zone that caused Co-RaL to diverge in `MoCap-E` is shorter, Co-RaL successfully converges over the whole trajectory. However, due to persistent velocity discrepancies between radar and leg kinematics, Co-RaL, which integrates both modalities, performs slightly worse than EKF-RIO and River in terms of $APE_t$ and $APE_z$. Despite such discrepancies, GaRLILEO again produces the most accurate odometry by leveraging both sensor modalities while robustly handling modality failures that impair cooperative estimation.

## 7.2 Leg Kinematics Odometry Comparison

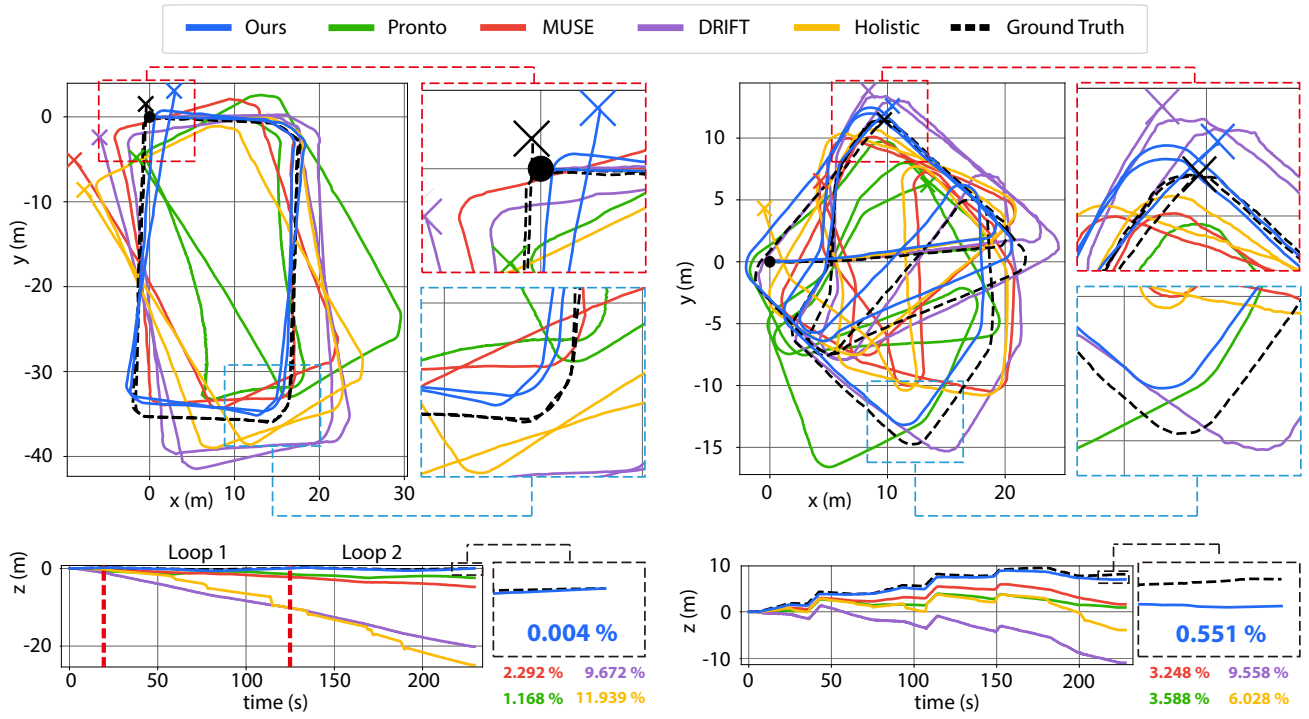
In this subsection, we compare GaRLILEO against four proprioceptive odometry methods that fuse IMU and leg kinematics sensors, including leg joint encoders and contact sensors. The baseline methods are described as follows:

- **Pronto** (Camurri et al. 2020): A proprioceptive-only version of Pronto, enabling real-time odometry estimation at high frequency, compatible with control loop.
- **MUSE** (Nisticò et al. 2025): A proprioceptive-only version of MUSE, a recent leg kinematics fused odometry that leverages a foot-slip detection algorithm for enhanced robustness.
- **Drift** (Lin et al. 2023): An invariant EKF-based leg odometry algorithm that fuses contact estimation and gyroscope filtering, designed for low-cost legged robots.
- **Holistic** (Nubert et al. 2025): A proprioceptive-only version of Holistic Fusion, leveraging foot contact points as landmark measurements. Our leg kinematics velocity estimation result is attached for the front-end.

This subsection evaluates the odometry accuracy of proprioceptive methods, with a focus on their performance in diverse environments. Table 4 presents quantitative results for APE and RPE metrics. GaRLILEO consistently achieves the most accurate odometry estimates across most sequences, primarily due to its integration of SoC radar-derived ego-velocity, which effectively mitigates the challenges posed by leg contact drift.

*1) Atrium:* Pronto, Drift, and Holistic achieve similar $APE_t$, while Pronto, through weighted averaging of leg kinematics velocities, attains the most accurate $APE_z$, demonstrating the reliability of leg kinematics in this environment. MUSE, which incorporates a slip detection algorithm, achieves a comparable $APE_z$ to Pronto and even lower $APE_t$, indicating the presence of slip or drift in the contact frame even in low-dynamic indoor settings. GaRLILEO achieves the most accurate overall odometry, presenting the lowest error on every metric.

*2) BridgeLoop and CorriLoop:* The `BridgeLoop` and `CorriLoop` sequences emphasize the importance of accurate roll and pitch estimation, particularly in mitigating vertical drift. As shown in Table 4, MUSE is more robust than Pronto in terms of $APE_t$. This highlights the advantage of a physical contact sensor that remains




(a) `CorriLoop` xy plot & z-time plot (b) `Upstair` xy plot & z-time plot

**Figure 11.** Leg-Kinematics baseline odometry results on the (a) `CorriLoop` and (b) `Upstair` sequences. Red dotted lines in (a, bottom) indicate the starting points of each loop in `CorriLoop` sequence. Both red zoomed-in views in (a) and (b) highlight that GaRLILEO integrating SoC radar and leg kinematics converge more closely to the ground-truth final position. The colored numbers on the right-hand side of the bottom plots show relative end-to-end vertical drift with respect to the full path length, where GaRLILEO achieves substantially lower drift compared with other baselines. Cyan zoomed view in (a) illustrates that GaRLILEO (blue) follows the most consistent turning trajectory across repeated loops. Similarly, in (b), the cyan zoomed view indicates that GaRLILEO (blue) follows the most accurate corner odometry during the upstairs with repeated turns. Both subfigures qualitatively show the odometry accuracy of GaRLILEO, particularly in terms of vertical drift suppression.

reliable indoors, including stairways and slopes. Drift and Holistic yield less accurate odometry, as they either assume a constant contact frame to correct IMU drift or treat contact frames as landmark features. In contrast, GaRLILEO robustly mitigates contact slip effects through radar velocity estimation and a velocity bias term, achieving the most accurate odometry overall, particularly in the vertical direction, owing to precise roll and pitch observation and an accurate gravity estimation scheme.

*3) BiCorridor:* In the `BiCorridor` sequence, although the environment is similar to `CorriLoop`, the results differ due to the inclusion of an upstairs section with a narrow turn. Pronto and MUSE produce relatively inaccurate odometry, as indicated by large $\mathrm{APE}_r$ from diverging rotational estimates. A similar trend is observed in Holistic, although its errors are more biased toward the vertical direction. Interestingly, Drift achieves lower $\mathrm{APE}_r$ than the other baselines due to its gyro filter scheme, but still suffers from substantial vertical drift. In contrast, GaRLILEO delivers the lowest error except $\mathrm{RPE}_r$, owing to precise gravity estimation that effectively suppresses divergence, particularly in the vertical axis.

*4) Downstair and Upstair:* The `Downstair` and `Upstair` sequences enable a comparison between heavy stair descent and ascent. Among the leg kinematics-only baselines, Drift exhibits the largest $\mathrm{APE}_t$ and $\mathrm{APE}_z$, but achieves relatively low $\mathrm{APE}_r$. Its gyro filter effectively mitigates horizontal errors from contact impacts on stairs; however, its vulnerability in roll and pitch estimation leads to substantial vertical divergence. Due to the repetitive rotation during stair ascent, Pronto and MUSE show much higher $\mathrm{APE}_r$ in `Upstair` compared with `Downstair`, as they rely on IMU-based rotational velocity for orientation estimation. All comparison methods record higher $\mathrm{APE}_t$ in the downstairs sequence, indicating that stair descent induces stronger contact drift. Despite these challenges, GaRLILEO achieves accurate vertical pose estimation with $\mathrm{APE}_z$ around $50\mathrm{cm}$, delivering the most accurate and robust results through radar-based velocity estimation, particularly in stair environments.

*5) SlopeStair:* In the `SlopeStair` sequence, leg-odometry performance degrades severely on staircases and outdoor-slope segments, especially just before re-entry indoors, at which point Pronto fails to converge. Drift and Holistic succeed in converging by relying on the contact sensor, but their trajectories diverge significantly. Interestingly, MUSE yields even worse $\mathrm{APE}_t$, yet maintains sub-meter $\mathrm{APE}_z$. This reflects the effectiveness of MUSE's slip detection module in handling vertical drift caused by contact slips, while horizontal drift—especially in yaw—remains difficult to suppress. In contrast, consistent with the radar–IMU experiments, GaRLILEO achieves the lowest error on every metrics in this sequence.

*6) Overpass:* In the `Overpass` sequence, the most severe contact slips in leg odometry occur on the stair sections positioned before and after the overpass. Drift achieves competitive $\mathrm{APE}_r$, but suffers from large vertical drift, resulting in the highest $\mathrm{APE}_t$. MUSE and Holistic

provide similar accuracy, particularly in $APE_t$ and $APE_z$. Pronto, although excluding its contact estimator, produces comparably robust odometry, especially in $APE_r$ and $APE_z$. In contrast, GaRLILEO achieves the lowest $APE_t$ and $APE_z$, demonstrating robustness to contact impacts during stair ascent and descent through its local gravity–based roll and pitch estimation.

*7) Tunnel:* In the `Tunnel` sequence, a single sharp U-turn largely determines the overall APE level, as it is the only rotational motion in the entire sequence. Drift achieves competitive horizontal rotation accuracy, recording the second-lowest $APE_r$. However, it fails to maintain robust horizontal orientation, resulting in the highest $APE_z$. Holistic yields the highest $APE_r$, and this single divergence in orientation estimation also leads to the largest $APE_t$, despite a moderate $APE_z$. Pronto and MUSE achieve similar $APE_z$, incorporating joint-encoder-based contact estimation and slip detection, respectively. In contrast, GaRLILEO delivers the most robust odometry across all metrics, owing to precise roll and pitch estimation before and after the sharp U-turn. This is evident in its lowest $APE_z$, which is approximately 13% of the second-best method.

*8) Quad:* The `Quad` sequence involves substantial contact drift, as it covers the longest path among all datasets and includes outdoor stairs and a long slope. Holistic shows low vertical APE ($APE_z$); however, orientation divergence causes overall odometry failure, as indicated by the highest $APE_t$ and $APE_r$. Drift produces more precise orientation estimates thanks to its gyro filtering scheme, yet still suffers from large $APE_t$ due to significant vertical drift. MUSE fails to converge over the full trajectory, unable to handle divergence in orientation estimation. Interestingly, Pronto achieves the best performance in both $APE_t$ and $APE_r$, demonstrating the potential of joint-encoder-based contact estimation, particularly in long outdoor environments. Even on this long outdoor sequence, GaRLILEO delivers the most accurate vertical estimation with its local-gravity model.

*9) MoCap-E and MoCap-H:* In the `MoCap` sequences, discrepancies in leg odometry arise in both horizontal and vertical directions, as also observed in the radar-based experiments. Drift records the highest $APE_t$ and $APE_z$, caused by vertical divergence in the cushion zone and horizontal errors in the sliding zone. Notably, all leg kinematics–IMU baselines exhibit limited accuracy in the sliding zone, since they lack a sensor modality capable of capturing the robot's state in a dynamic floor. Compared with Holistic, both Pronto and MUSE achieve lower $APE_t$, showing greater robustness to contact failure through joint encoder–based contact estimation or slip detection. Among these, MUSE achieves the lowest $APE_t$ and $APE_z$, as its slip detection module effectively mitigates contact failure, leading to more robust odometry estimation. Across the two `MoCap` sequences, GaRLILEO is most accurate, thanks to radar-derived ego-velocity fed into the velocity spline, especially in slippery sections where leg-kinematics-based estimation fails.

**Table 5.** Complementary effects of radar and leg kinematics on SNU sequences. **L+I** refers to the leg kinematics-only and **R+I** refers to the radar-only version of the full GaRLILEO (**R+L+I**).

| | | R+L+I | R+I | L+I | EKF-RIO | MUSE |
|---|---|---|---|---|---|---|
| Atrium | $APE_t$ | **0.816** | 9.793 | 1.423 | 6.679 | 3.297 |
| | $RPE_t$ | **0.055** | 0.191 | 0.085 | 0.162 | 0.090 |
| | $APE_z$ | **0.106** | 9.771 | 0.192 | 4.908 | 0.685 |
| | $APE_{xy}$ | 0.809 | **0.661** | 1.410 | 4.530 | 3.225 |
| BridgeLoop | $APE_t$ | 1.193 | 11.802 | **1.051** | 7.125 | 4.195 |
| | $RPE_t$ | **0.080** | 0.167 | 0.143 | 0.132 | 0.200 |
| | $APE_z$ | **0.117** | 11.773 | 0.187 | 5.302 | 2.250 |
| | $APE_{xy}$ | 1.187 | **0.827** | 1.034 | 4.760 | 3.541 |
| CorriLoop | $APE_t$ | 1.627 | 17.758 | **1.539** | 9.283 | 5.811 |
| | $RPE_t$ | **0.066** | 0.202 | 0.093 | 0.180 | 0.127 |
| | $APE_z$ | 0.168 | 17.667 | **0.096** | 4.001 | 2.514 |
| | $APE_{xy}$ | 1.619 | 1.801 | **1.536** | 8.377 | 5.238 |
| BiCorridor | $APE_t$ | 1.425 | 23.785 | **1.277** | 10.293 | 15.035 |
| | $RPE_t$ | **0.063** | 0.220 | 0.117 | 0.185 | 0.152 |
| | $APE_z$ | 0.226 | 23.740 | **0.144** | 5.497 | 2.880 |
| | $APE_{xy}$ | 1.407 | 1.472 | **1.269** | 8.702 | 14.757 |
| Downstair | $APE_t$ | 3.961 | 10.315 | **3.933** | 30.835 | 8.323 |
| | $RPE_t$ | **0.099** | 0.169 | 0.123 | 0.229 | 0.182 |
| | $APE_z$ | **0.530** | 10.262 | 0.738 | 4.251 | 4.843 |
| | $APE_{xy}$ | 3.926 | **1.049** | 3.863 | 30.541 | 6.768 |
| Upstair | $APE_t$ | **1.496** | 17.446 | 1.633 | 10.044 | 5.307 |
| | $RPE_t$ | **0.071** | 0.185 | 0.135 | 0.143 | 0.179 |
| | $APE_z$ | **0.514** | 17.339 | 0.575 | 8.871 | 3.206 |
| | $APE_{xy}$ | **1.405** | 1.932 | 1.529 | 4.711 | 4.230 |
| SlopeStair | $APE_t$ | **2.359** | 22.781 | 3.452 | - | 35.519 |
| | $RPE_t$ | **0.061** | 0.237 | 0.120 | - | 0.151 |
| | $APE_z$ | **0.665** | 21.894 | 1.142 | - | 0.967 |
| | $APE_{xy}$ | **2.264** | 6.295 | 3.257 | - | 35.506 |
| Overpass | $APE_t$ | **1.526** | 6.531 | 2.048 | - | 6.507 |
| | $RPE_t$ | **0.091** | 0.329 | 0.135 | - | 0.191 |
| | $APE_z$ | 1.074 | 6.248 | **0.410** | - | 2.442 |
| | $APE_{xy}$ | **1.084** | 1.902 | 2.007 | - | 6.031 |
| Tunnel | $APE_t$ | **3.523** | 17.355 | 3.950 | 26.800 | 10.194 |
| | $RPE_t$ | 0.083 | 0.185 | **0.079** | 0.316 | 0.105 |
| | $APE_z$ | **0.548** | 4.836 | 0.941 | 6.139 | 5.425 |
| | $APE_{xy}$ | **3.481** | 16.668 | 3.836 | 26.088 | 8.631 |
| Quad | $APE_t$ | **7.347** | 53.857 | 9.662 | - | - |
| | $RPE_t$ | **0.080** | 0.290 | 0.084 | - | - |
| | $APE_z$ | **2.143** | 37.264 | 4.701 | - | - |
| | $APE_{xy}$ | **7.028** | 38.883 | 8.442 | - | - |

## 7.3 Complementary Effects of Radar and Leg Kinematics

To analyze the complementary roles of radar and leg kinematics, we compare three GaRLILEO configurations: the full system **R+L+I**, a radar-IMU-only variant **R+I** in which the leg kinematics factor $\mathbf{r}_L$ is disabled, and a leg-IMU-only variant **L+I** in which the radar factor $\mathbf{r}_R$ is disabled. We additionally report EKF-RIO and MUSE for the SNU sequences, and River and MUSE for the RAI sequences, as they are the best-performing representative radar-based and leg-based comparisons of each sensor configuration.

*1) SNU Sequences:* Table 5 summarizes the results on the SNU sequences. Overall, the sensor-subset variants **R+I** and **L+I** exhibit complementary strengths. When radar returns are sufficiently informative, as is often the case in indoor environments, **R+I** can provide competitive horizontal constraints in terms of $APE_{xy}$ on several sequences, whereas **L+I** substantially improves vertical robustness in terms of $APE_z$ by leveraging high-rate proprioceptive velocity information. By fusing both modalities, **R+L+I** attains the best accuracy on most sequences, reflecting cooperative fusion that emphasizes the more reliable modality when the other becomes less informative.

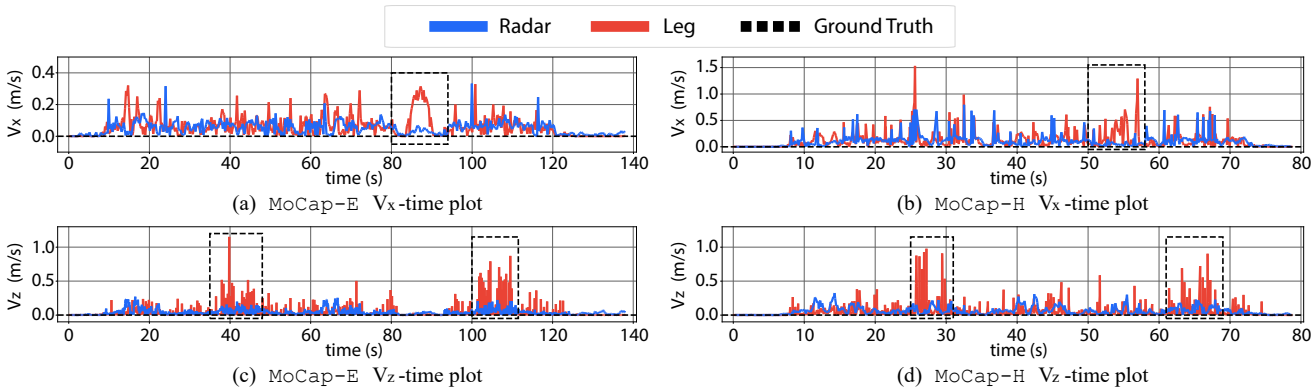



**Figure 12.** $v_x$-time and $v_z$-time Velocity Error (difference from ground truth) in the MoCap-E and MoCap-H sequences. A dotted square on the (a) and (b) includes a slippery zone where the floor moves backwards, while two dotted squares on the (c) and (d) include cushion zones where the leg kinematics velocity presents a high impact on the vertical direction.

Regarding the modality-specific comparisons, **L+I** is consistently competitive with, and often outperforms MUSE on SNU sequences, suggesting that our continuous-time formulation and factor design provide strong proprioceptive odometry. In practice, the radar input is available at 20 Hz, and using radar alone is not sufficient to fully realize the benefits of our proposed modules; as a result, **R+I** tends to perform at a similar level to dedicated radar-IMU pipeline, EKF-RIO on several sequences.

*2) RAI Sequences:* To further evaluate complementarity under conditions where leg-kinematics-based odometry degrades, we conduct additional analysis on the MoCap-E and MoCap-H sequences, which include intentional perturbations in leg odometry, as visualized in Figure 12. Quantitative results are reported in Table 6.

On MoCap-E sequence, where the robot stays stop on the slippery zone during the floor is moving to backward, significant difference between radar-aided methods (i.e. **R+L+I**, **R+I**, and River) presents comparably more accurate $\text{APE}_{xy}$ then leg kinematic only methods (i.e. **L+I** and MUSE), proving the radar sensor is working cooperatively with leg kinematic sensors when the stationary plane assumption breaks. In $\text{APE}_z$, River presents lower error compared with MUSE, as the cushion zone ruins the leg kinematic estimation. Similarly, as can be found from Figure 13 (a), only **R+I** keeps stable odometry during the cushion zone, while leg kinematic aided ones goes slightly upper direction. However, due to the local gravity spline which is more benefited by the high frequency egocentric-velocity measurement of leg kinematics, $\text{APE}_z$ of **R+I** is higher than other versions of GaRLILEO. Still, the combination of both sensors, full GaRLILEO, presents the most accurate odometry estimation. It can be concluded that radar and leg kinematics can work in complementary ways to estimate the robot's odometry in a slippery environment accurately.

On MoCap-H sequence, though the existence of cushion zone and slippery zone is similar, details like maintaining walking on slippery zone and higher overall robot speed are different. As can be found from Figure 13 (b), **L+I** fails at a similar point due to the broken stationary floor assumption in the slippery zone. Because of this, **L+I** results in higher $\text{APE}_{xy}$ compared with other GaRLILEO variants. On vertical drift, the tendency of $\text{APE}_z$ is similar to MoCap-E, but the upper direction bias of leg kinematics in the cushion zone is notably mild because of the locomotion speed difference. Due to the higher speed of the robot, the effect of the cushion less emerges and leads to a highly accurate $\text{APE}_z$ of **L+I**. In conclusion, exploiting the biased accuracy of **R+I** and **L+I** on $\text{APE}_{xy}$ and $\text{APE}_z$ each, **R+L+I** presents most accurate $\text{APE}_t$ overall, while lying between **R+I** and **L+I** on other APE metrics. This tendency to rely more on relatively more accurate sensor modalities can be inferred as the cooperative effect of GaRLILEO.

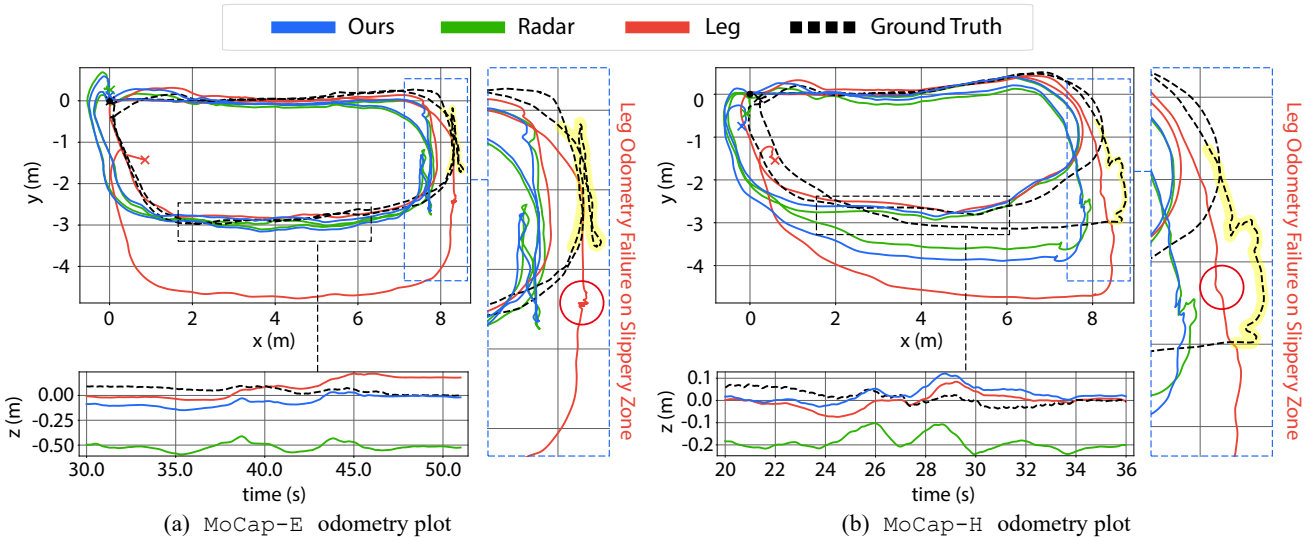
**Table 6.** Complementary effects of radar and leg kinematic on RAI sequences. **L+I** refers to the leg kinematics-only and **R+I** refers to the radar-only version of the full GaRLILEO (**R+L+I**).

| | | R+L+I | R+I | L+I | River | MUSE |
|---|---|---|---|---|---|---|
| MoCap-E | $\text{APE}_t$ | **0.852** | 1.010 | 1.171 | 0.916 | 1.313 |
| | $\text{RPE}_t$ | **0.184** | 0.243 | 0.318 | 0.239 | 0.201 |
| | $\text{APE}_z$ | **0.103** | 0.563 | 0.256 | 0.236 | 0.818 |
| | $\text{APE}_{xy}$ | 0.846 | **0.838** | 1.143 | 0.885 | 1.027 |
| MoCap-H | $\text{APE}_t$ | **0.880** | 1.038 | 0.920 | 1.145 | 0.900 |
| | $\text{RPE}_t$ | 0.204 | 0.242 | 0.220 | 0.250 | **0.195** |
| | $\text{APE}_z$ | 0.118 | 0.643 | **0.102** | 0.236 | 0.283 |
| | $\text{APE}_{xy}$ | 0.871 | **0.814** | 0.915 | 1.120 | 0.854 |

## 7.4 *Effect of Gravity Factors on State Estimation Accuracy*

In this subsection, we evaluate the effect of our gravity factors $\mathbf{r}_{\mathcal{S}^2}, \mathbf{r}_{\mathbf{post}}$ on two different sub-experiments: 1) effect of the soft $\mathcal{S}^2$-constrained gravity factor, and 2) effect of the post optimization.

*1) Effect of the Soft $\mathcal{S}^2$-Constrained Gravity Factor:* In the first sub-experiment, we aimed to highlight the effectiveness of the soft $\mathcal{S}^2$-constrained gravity factor by comparing the accuracy of the estimated local gravity vector between the full model of GaRLILEO: **Ours** and the model without the factor: **w/o** $\mathbf{r}_{\mathcal{S}^2}$. Ground-truth local gravity $\mathbf{g}_t^*$ was obtained by (i) rigidly aligning the ground truth pose to the estimated trajectory with Evo Trajectory Evaluator (Grupp 2017), (ii) B-spline interpolation of the aligned poses at the estimating timestamps, and (iii) rotating the global gravity vector into the local frame using aligned ground truth orientation as




(a) MoCap-E odometry plot (b) MoCap-H odometry plot

**Figure 13.** xy plot and z-time plot of MoCap sequences. (a) and (b) are xy and z-time odometry plot of MoCap-E and MoCap-H sequences, respectively. Blue dotted box: zoomed view of slippery zone where leg odometry fails temporarily, especially on the xy plane. The yellow-highlighted area is the exact slippery zone. Black dotted box: zoomed side view of cushion zone where leg odometry fails temporarily, especially on the vertical direction. Comparing the radar-only, leg kinematics-only, and full module versions of GaRLILEO to check the cooperative effect of radar and leg sensors in the slippery zone to check the cooperative work between radar and leg kinematics.

follows:

$$\mathbf{g}_t^* = \mathbf{R}_t^\top \mathbf{g}_0, \quad \text{where} \quad \mathbf{g}_0 = \begin{bmatrix} 0 \\ 0 \\ -9.81 \end{bmatrix}. \tag{35}$$

To evaluate roll and pitch observation accuracy, we use the mean angular error of the estimated local gravity vectors over each sequence.

As detailed in Table 7, the ablation without the soft $\mathcal{S}^2$ gravity factor (**w/o** $\mathbf{r}_{\mathcal{S}^2}$) estimates local gravity with errors exceeding twice those of **Ours**. This behavior is closely related to gravity-norm preservation during continuous-time estimation. Although we clamp the gravity control points to have magnitude $9.81\ \mathrm{m/s^2}$, gravity is parameterized with an $\mathbb{R}^3$ B-spline rather than an $\mathcal{S}^2$ spline; hence, the interpolated B-spline segment between adjacent control points is not inherently norm-preserving. As a result, without the soft $\mathcal{S}^2$ factor, $\|\mathbf{g}(t)\|$ may exhibit local norm drops between control points, which degrade the reliability of local gravity direction estimation, especially under strong contact-induced vibrations.

To make this explicit, we conducted additional experiments to measure and compare the gravity-norm behavior with and without the soft $\mathcal{S}^2$ factor $\mathbf{r}_{\mathcal{S}^2}$. Specifically, we compute the minimum, maximum, mean, and standard deviation of the gravity vector's magnitude $\|\mathbf{g}(t)\|$ over each sequence, which are summarized in Table 8. The minimum captures the worst-case local norm dip, while the standard deviation reflects the overall fluctuation. Additionally, we plot the gravity-norm behavior over runtime on seq. SlopeStair in Figure 14.

**Table 7.** Effect of $\mathbf{r}_{\mathcal{S}^2}$ Factor on Estimating Local Gravity

| ° (deg) | Atrium | BridgeLoop | CorriLoop | BiCorridor |
|---|---|---|---|---|
| **Ours** | **1.507** | **2.000** | **1.438** | **1.483** |
| **w/o** $\mathbf{r}_{\mathcal{S}^2}$ | 4.815 | 4.229 | 2.735 | 4.651 |
| ° (deg) | Downstair | Upstair | SlopeStair | Overpass |
| **Ours** | **1.727** | **1.812** | **2.147** | **1.682** |
| **w/o** $\mathbf{r}_{\mathcal{S}^2}$ | 4.637 | 5.092 | 5.436 | 3.703 |
| ° (deg) | Tunnel | Quad | MoCap-E | MoCap-H |
| **Ours** | **1.851** | **2.702** | **4.244** | **3.904** |
| **w/o** $\mathbf{r}_{\mathcal{S}^2}$ | 2.923 | 4.040 | 5.046 | 5.718 |

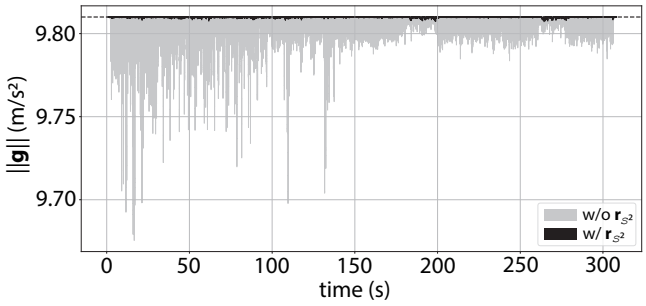



**Figure 14.** Gravity-norm behavior over time in SlopeStair sequence (w/ vs. w/o $\mathbf{r}_{\mathcal{S}^2}$). Enabling equation (20) suppresses local norm dips of $\|\mathbf{g}(t)\|$ between adjacent control points, keeping $\|\mathbf{g}(t)\|$ much closer to $9.81\ \mathrm{m/s^2}$.

**Table 8.** Gravity magnitude statistics ($\mathrm{m/s^2}$) without and with the $\mathbf{r}_{\mathcal{S}^2}$ factor.

| | w/o $\mathbf{r}_{\mathcal{S}^2}$ | | | | w/ $\mathbf{r}_{\mathcal{S}^2}$ | | | |
|---|---|---|---|---|---|---|---|---|
| Sequence | Min | Max | Mean | Std. | Min | Max | Mean | Std. |
| Atrium | 9.7250 | 9.8100 | 9.8046 | $6.43\times10^{-3}$ | 9.8088 | 9.8100 | 9.8099 | $7.53\times10^{-5}$ |
| BridgeLoop | 9.6489 | 9.8100 | 9.8004 | $1.11\times10^{-2}$ | 9.8069 | 9.8100 | 9.8099 | $1.90\times10^{-4}$ |
| CorriLoop | 9.6888 | 9.8100 | 9.7960 | $1.43\times10^{-2}$ | 9.8081 | 9.8100 | 9.8099 | $8.87\times10^{-5}$ |
| BiCorridor | 9.7335 | 9.8100 | 9.7974 | $1.08\times10^{-2}$ | 9.8070 | 9.8100 | 9.8099 | $1.41\times10^{-4}$ |
| Downstair | 9.7123 | 9.8100 | 9.8044 | $9.14\times10^{-3}$ | 9.8047 | 9.8100 | 9.8099 | $2.75\times10^{-4}$ |
| Upstair | 9.6334 | 9.8100 | 9.7866 | $2.27\times10^{-2}$ | 9.8026 | 9.8100 | 9.8099 | $2.18\times10^{-4}$ |
| SlopeStair | 9.6722 | 9.8100 | 9.8013 | $1.03\times10^{-2}$ | 9.8071 | 9.8100 | 9.8099 | $2.08\times10^{-4}$ |
| Overpass | 9.7189 | 9.8100 | 9.8036 | $8.90\times10^{-3}$ | 9.8060 | 9.8100 | 9.8099 | $2.57\times10^{-4}$ |
| Tunnel | 9.6534 | 9.8100 | 9.8015 | $1.04\times10^{-2}$ | 9.8073 | 9.8100 | 9.8099 | $1.10\times10^{-4}$ |
| Quad | 9.7016 | 9.8100 | 9.8053 | $8.17\times10^{-3}$ | 9.8076 | 9.8100 | 9.8099 | $1.34\times10^{-4}$ |
| MoCap-E | 9.7940 | 9.8100 | 9.8087 | $1.74\times10^{-3}$ | 9.8085 | 9.8100 | 9.8099 | $1.30\times10^{-4}$ |
| MoCap-H | 9.6579 | 9.8100 | 9.7988 | $1.44\times10^{-2}$ | 9.7959 | 9.8100 | 9.8098 | $4.44\times10^{-4}$ |

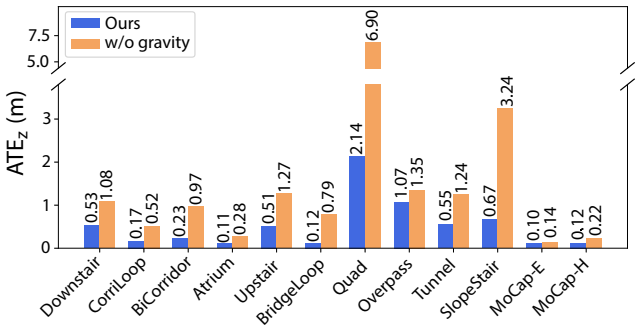



**Figure 15.** Effect of $\mathbf{r}_{\mathbf{post}}$ on vertical pose estimation. Vertical odometry accuracy is notably enhanced when $\mathbf{r}_{\mathbf{post}}$ is added to refine the rotation using the estimated local gravity in a post optimization step. This result supports the idea that precise local gravity estimation can provide roll and pitch observations, thereby mitigating vertical drift in odometry.

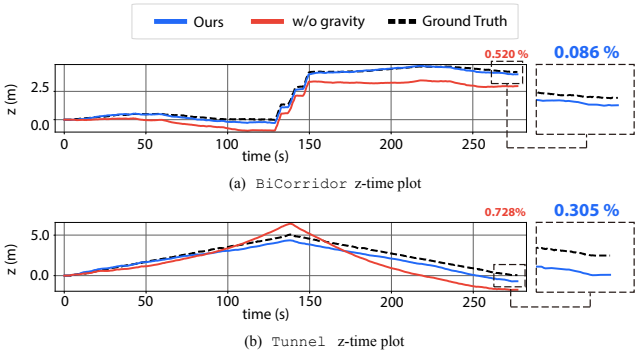



**Figure 16.** Z-time odometry plot including comparison between with and without the post optimization. $\mathbf{r}_{\mathbf{post}}$ factor prevents odometry from diverging in the vertical direction.

As shown in Figure 14, soft $\mathcal{S}^2$ factor $\mathbf{r}_{\mathcal{S}^2}$ suppresses local norm dips between adjacent control points. Table 8 further quantifies this effect across every sequence: without $\mathbf{r}_{\mathcal{S}^2}$, the minimum $\|\mathbf{g}(t)\|$ can drop to $9.633\ \mathrm{m/s^2}$ and the standard deviation is on the order of $10^{-2}\ \mathrm{m/s^2}$, whereas with $\mathbf{r}_{\mathcal{S}^2}$ the minimum dip improves to $9.796\ \mathrm{m/s^2}$ and the standard deviation decreases to the order of $10^{-4}\ \mathrm{m/s^2}$. The maximum stays at $9.81\ \mathrm{m/s^2}$ because the B-spline evaluates $\mathbf{g}(t)$ as a convex weighted combination of control points with nonnegative weights that sum to one, and all control points are clamped to have magnitude $9.81\ \mathrm{m/s^2}$.

*2) Effect of the Post Optimization:* In the second sub-experiment, we compared the direct vertical drift APE$_z$ with and without the rotation refinement via post optimization. As mentioned in the previous sub-experiment, this level of accurate local gravity is directly translated into more precise observability of roll and pitch. As shown in Figure 15, the addition of the $\mathbf{r}_{\mathbf{post}}$ factor notably improves odometry accuracy in the vertical direction. Specifically, APE$_z$ is consistently reduced on every sequence; for example, on `Quad`, the vertical drift is reduced about 5 m through the addition of post optimization process. Example graphs of `BiCorridor` and `Tunnel` sequences are presented in Figure 16. Notably, even without loop-closure or point cloud-based registration schemes, GaRLILEO achieves APE$_z$ lower than 1 m on most sequences.

Taken together, these two sub-experiments demonstrate that GaRLILEO maintains a small, single-digit level of accuracy in local gravity estimation, even under aggressive legged robot locomotion, and that this precision is sufficient to deliver robust, low-drift vertical-state estimation on stairs, slopes, and even on slippery, potentially deformable surfaces.

## 7.5 Effect of Velocity Bias

In this subsection, we evaluate the odometry performance of GaRLILEO with and without the velocity bias. The results are summarized in Table 9, where the error metric is APE$_t$ projected onto the $xy$ plane. This 2D metric is used because the velocity bias operates only in the horizontal directions, where foot slip occurs parallel to the ground plane, resulting in horizontal velocity discrepancies.

As shown in Table 9, more than half of the sequences exhibit improved accuracy with the bias, while the others perform better without it. Specifically, in the `Downstair`, `Upstair`, `MoCap-E`, and `MoCap-H` sequences, excluding the velocity bias yields more accurate $xy$ plane odometry. A common feature of these datasets is the presence of long indoor stairs or challenging environments that frequently degrade leg odometry.

By contrast, in most indoor sequences, adding the bias improves accuracy, as radar sensors acquire denser, more reliable measurements from nearby static objects. Qualitative odometry is shown in Figure 17, which demonstrates enhanced odometry accuracy for both indoor and outdoor sequences.

On indoor flat sequences, such as `Atrium` and `CorriLoop`, the velocity bias term improves odometry accuracy by accounting for potential contact slip using radar target radial velocity information. Due to the relatively rich radar targets in indoor environments, velocity bias may provide more reliable velocity information to the system's velocity factors. Similarly, on `Tunnel` and `Overpass` sequences, where the surrounding library building or overpass (including the ceiling) enables the radar to collect sufficient reflections, the result is that odometry got enhanced even if a part of the trajectory traverses open outdoor space.

In `BridgeLoop` and `BiCorridor` sequences, where short stairs are included in the trajectory, present partially enhanced odometry estimation, but their enhancement proportion is a slight fall back compared with simple flat indoor sequences. This is due to the direction of the radar sensor when the robot is traversing staircases. As the radar sensor points upward, it may detect fewer radar targets, since fewer objects may potentially exist in that direction. This phenomenon occurs in `Downstair` and `Upstair` sequences, where the longest stair is included in both upstairs

**Table 9.** Effect of Velocity Bias on APE$_t$ projected onto the $xy$ plane

| m (meter) | Atrium | BridgeLoop | CorriLoop | BiCorridor |
|---|---|---|---|---|
| **ours** | **0.809** | **1.187** | **1.619** | **1.207** |
| **w/o bias** | 1.375 | 1.297 | 2.329 | 1.831 |
| m (meter) | Downstair | Upstair | SlopeStair | Overpass |
| **ours** | 3.926 | 1.405 | **2.264** | **1.084** |
| **w/o bias** | **3.752** | **0.938** | 4.395 | 1.936 |
| m (meter) | Tunnel | Quad | MoCap-E | MoCap-H |
| **ours** | **3.481** | **7.028** | 0.846 | 0.885 |
| **w/o bias** | 4.652 | 7.114 | **0.794** | **0.805** |

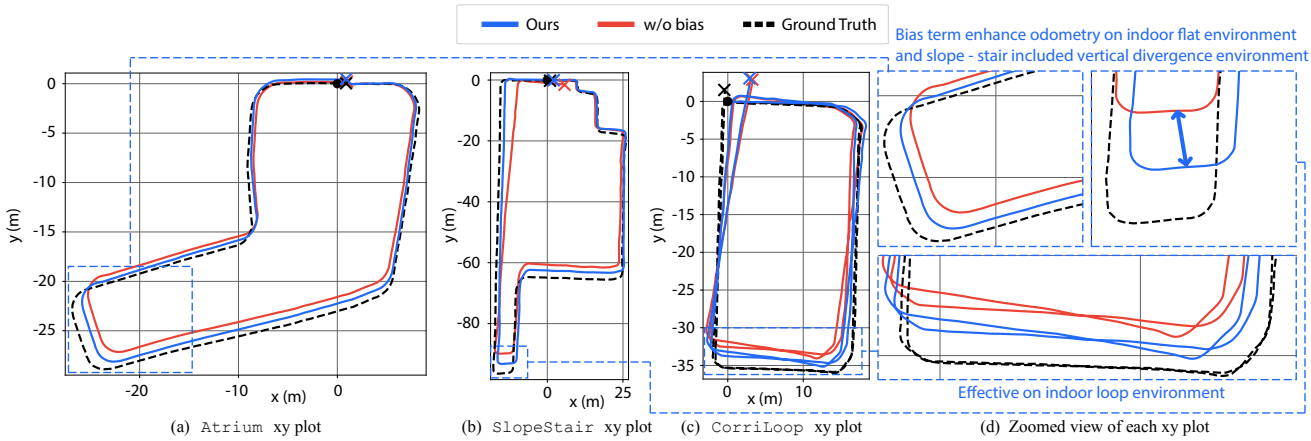



**Figure 17.** $xy$ plane plot of odometry estimation with and without velocity bias. On `Atrium`, `SlopeStair`, and `CorriLoop` sequences, which are presented in (a), (b), and (c), velocity bias makes odometry more robust as radial velocity information of each radar target point mitigates the error of leg velocity occurring from inaccurate contact and joint encoder measurement. A more detailed zoomed view of each subfigure, showing the effect of the velocity bias term, is included in (d).

and downstairs directions, which even degrades the final odometry estimation when a velocity bias is added. Still, as shown in Table 9, the performance degradation in those cases is less drastic than the performance enhancement on other sequences.

One interesting point about this experiment is that even the `SlopeStair` sequence includes a long upstairs region, similar to the downstairs area of `Downstair`; however, the final odometry estimate was enhanced by almost 50%. This indirectly demonstrates the effect of the velocity bias factor on mitigating mild leg odometry failures, such as temporal contact slip or contact impact, using radar target velocity information. Even though the possibility of odometry degradation in the stair region exists, the velocity bias factor meaningfully handles the contact failure in the slope area, where almost half of the `SlopeStair` sequence is designed. This can be qualitatively verified from Figure 17 (b) and Figure 17 (d), as the odometry accuracy degrades after the downstair region, particularly without a bias term, due to temporal contact slip or impact. The `Quad` sequence also includes the upstair and down-slope regions during the traverse, but the velocity bias effect is shown comparably mild, as the slope in this sequence is much milder compared with the `SlopeStair` sequence.

In the two `MoCap` sequences, leg odometry deviates from ground truth in the $xy$ plane due to the slippery zone. Because the bias term is designed to change smoothly, it cannot compensate for abrupt or inconsistent slips, resulting in slight performance degradation.

Overall, the velocity bias can enhance odometry by reducing the discrepancy between radar and leg kinematics when leg kinematics temporally fail due to unanticipated contact issues. Still, a potential limitation of the velocity bias on drastically varying discrepancies remains. This effect is particularly evident in places where the static floor assumption fails, such as `MoCap` sequences. To address this, future work may consider alternative radar configurations (e.g., including ground reflections within the field of view) or an adaptive mechanism that enables or disables the bias term temporally depending on the environment in which the system operates.

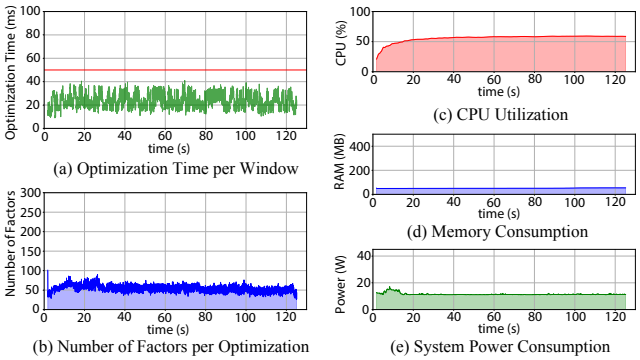



**Figure 18.** End-to-end edge device test of GaRLILEO on NVIDIA Jetson AGX Orin with `Atrium` sequence. (a) Optimization time per window during factor graph optimization. Red line denotes 50 ms, which stands for speed of SoC radar sensor input. (b) Number of factors in the factor graph per single optimization. (c) CPU Utilization of GaRLILEO. (d) Memory Consumption of GaRLILEO. (e) Power consumption of NVIDIA Jetson AGX Orin while running GaRLILEO.

## 7.6 Real-time Capability on Edge Device

To demonstrate the real-time capability and computational efficiency of our framework, we conducted an experiment by deploying GaRLILEO on an onboard NVIDIA Jetson AGX Orin for the `Atrium` Sequence. A key factor in achieving this lightweight performance is the sliding window marginalization scheme, detailed in Section §5.2.2. This ensures the factor graph size does not grow unbounded over time. As shown in Figure 18 (b), an average of 54 residual factors processed per single optimization step, while graph size is internally bounded to a maximum of 21 active control points. This consistency is maintained throughout the entire estimation process.

This bounded graph size directly translates to efficient computation frequencies. During the experiment, the end-to-end computation time per sliding window optimization averaged 23.277 ms, with a maximum recorded value of 41.097 ms. Because the incoming radar scans operate at 20 Hz, the system is required to complete its optimization within a 50 ms window. As illustrated by the red

threshold line in Figure 18 (a), GaRLILEO's computation time consistently stays below this, guaranteeing real-time deployment capabilities on the edge device without dropping sensor frames.

Furthermore, the overall system throughput and resource consumption demonstrates the stability of GaRLILEO during operation. On the NVIDIA Jetson AGX Orin, GaRLILEO's CPU utilization averaged $55.856\,\%$, while memory consumption remained at an average of $51.013$ MB. The average system power usage during execution was $11.521$ W, while maximum at $17.198$ W. As depicted in Figure 18 (c), Figure 18 (d), and Figure 18 (e), the CPU, memory, and power metrics do not exhibit drastic growth over the algorithm runtime. Instead, they maintain a stable and predictable level, confirming GaRLILEO is suitable for deployment on power-constrained systems.

## 8 Conclusion

In this work, we presented GaRLILEO, a continuous-time radar-leg-IMU odometry framework that leverages B-spline modeling, local gravity estimation, and horizontal velocity bias correction to achieve robust and accurate state estimation in challenging environments. By tightly coupling radar-derived ego velocity with leg kinematics and inertial measurements, our method addresses the limitations of existing odometry pipelines that suffer from vertical drift, contact slip, and sensor modality degradation.

Extensive experiments across diverse sequences, including long indoor loops, multi-story staircases, outdoor slopes, and environments with severe contact disturbances, demonstrated that GaRLILEO consistently outperforms other SOTA baselines in odometry accuracy. Notably, its continuous local gravity estimation significantly reduces vertical drift. At the same time, the B-spline-based fusion scheme ensures robustness against modality-specific failures such as radar sparsity or leg kinematics slips.

To evaluate the complementary effect of radar and leg kinematics sensors, we tested the performance of the radar-only and leg-only versions of GaRLILEO at `MoCap` sequences and compared their odometry accuracy. Through this experiment, we verified that both sensor modalities have their own strength points in horizontal and vertical direction accuracy. Consequently, the B-spline-based sensor fusion of GaRLILEO effectively blends the different modalities to enhance the odometry estimation result.

Through two sub-experiments, we validated the effect of proposing local gravity factors on both the accuracy of local gravity estimation itself and odometry in the vertical direction. These experiments demonstrate how GaRLILEO can achieve sub-meter vertical odometry accuracy without relying on loop closure or point cloud registration algorithms, highlighting that reliable local gravity estimation is crucial for robust, vertically low-drift odometry estimation in stairs, slopes, and challenging terrains.

We further analyzed the effect of the velocity bias term and verified that it provides clear benefits in structured indoor environments, while in sparse or highly dynamic outdoor scenes, it may lead to potential performance degradation. This suggests that environment-aware adaptation or improved radar hardware configurations could further enhance robustness.

Overall, our results highlight the potential of continuous-time multimodal fusion to enable legged robots to navigate reliably in complex real-world environments.

### 8.1 Limitations and Future Extensions

Though GaRLILEO demonstrates robust odometry estimation across diverse indoor and outdoor environments, several limitations remain. First, our framework prefers denser radar points for stable velocity spline estimation. In open outdoor areas such as the `Quad` sequence, sparse radar returns degrade velocity bias adaptation and limit accuracy. Furthermore, due to the limited observation of yaw orientation, which relies on the IMU measurements, an inevitable odometry drift occurs in the horizontal direction as the trajectory becomes longer.

To mitigate the above limitations, future work will explore adaptive gravity post-optimization, which dynamically enables or disables the term depending on the reliability of the optimized gravity vector. For classification, fusion with exteroceptive sensors, such as cameras and LiDAR, or friction estimation based on recent contact tactile sensors can be exploited. Furthermore, to expand the field of view and improve odometry robustness, a multi-SoC radar system can be leveraged, which may provide additional yaw orientation observations, as mentioned in Yoon et al. (2023), or enable point cloud registration.

Additionally, the current framework relies on static covariances during optimization. To address the unpredictability of legged-robot environments, future work will explore adaptive covariance estimation that dynamically adjusts factor weights based on real-world environmental context and sensor degradation, thereby further enhancing robustness.

#### Acknowledgements

This work was supported by the Technology Innovation Program (1415187329, 20024355, Development of autonomous driving connectivity technology based on sensor-infrastructure cooperation) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea), and in part by the Robotics and AI (RAI) Institute.

## References

Agha A, Otsu K, Morrell B, Fan DD, Thakker R, Santamaria-Navarro A, Kim SK, Bouman A, Lei X, Edlund J et al. (2021) Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge. *arXiv preprint arXiv:2103.11470* .

Arndt C, Sabzevari R and Civera J (2023) Do planar constraints improve camera pose estimation in monocular slam? In: *Proceedings of the IEEE International Conference on Computer Vision*. pp. 2221–2230.

Barnes D, Gadd M, Murcutt P, Newman P and Posner I (2020) The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 6433–6438.

Bloesch M, Burri M, Sommer H, Siegwart R and Hutter M (2017) The two-state implicit filter recursive estimation for mobile robots. *IEEE Robotics and Automation Letters* 3(1): 573–580.

Bloesch M, Gehring C, Fankhauser P, Hutter M, Hoepflinger MA and Siegwart R (2013) State estimation for legged robots on unstable and slippery terrain. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 6058–6064.

Bloesch M, Hutter M, Hoepflinger MA, Leutenegger S, Gehring C, Remy CD and Siegwart R (2012) State estimation for legged robots-consistent fusion of leg kinematics and imu. In: *Proceedings of the Robotics: Science & Systems Conference*. pp. 17–24.

Burnett K, Schoellig AP and Barfoot TD (2025) Continuous-time radar-inertial and lidar-inertial odometry using a gaussian process motion prior. *IEEE Transactions on Robotics* 41: 1059–1076.

Camurri M, Ramezani M, Nobili S and Fallon M (2020) Pronto: A multi-sensor state estimator for legged robots in real-world scenarios. *Frontiers in Robotics and AI* 7: 68.

Chen H, Liu Y and Cheng Y (2023) Drio: Robust radar-inertial odometry in dynamic environments. *IEEE Robotics and Automation Letters* 8(9): 5918–5925.

Chen S, Li X, Li S, Zhou Y and Wang S (2024) River: A tightly-coupled radar-inertial velocity estimator based on continuous-time optimization. *IEEE Robotics and Automation Letters* 9(7): 6107–6114.

Dhédin V, Li H, Khorshidi S, Mack L, Ravi AKC, Meduri A, Shah P, Grimminger F, Righetti L, Khadiv M et al. (2023) Visual-inertial and leg odometry fusion for dynamic locomotion. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 9966–9972.

Do HV, Kim YH, Lee JH, Lee MH and Song JW (2024) Dero: Dead reckoning based on radar odometry with accelerometers aided for robot localization. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 8547–8554.

Doer C and Trommer GF (2020) An ekf based approach to radar inertial odometry. In: *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligence System*. pp. 152–159.

Doer C and Trommer GF (2021) Yaw aided radar inertial odometry using manhattan world assumptions. In: *Proceedings of the IEEE Saint Petersburg International Conference on Integrated Navigation System*. pp. 1–9.

Droeschel D and Behnke S (2018) Efficient continuous-time slam for 3d lidar-based online mapping. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 5000–5007.

Fallon MF, Antone M, Roy N and Teller S (2014) Drift-free humanoid state estimation fusing kinematic, inertial and lidar sensing. In: *Proceedings of the IEEE International Conference on Humanoid Robots*. pp. 112–119.

Fink G and Semini C (2020) Proprioceptive sensor fusion for quadruped robot state estimation. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 10914–10920.

Furgale P, Barfoot TD and Sibley G (2012) Continuous-time batch estimation using temporal basis functions. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 2088–2095.

Furgale P, Tong CH, Barfoot TD and Sibley G (2015) Continuous-time batch trajectory estimation using temporal basis functions. *International Journal of Robotics Research* 34(14): 1688–1710.

Geiger A, Lenz P, Stiller C and Urtasun R (2013) Vision meets robotics: The kitti dataset. *International Journal of Robotics Research* 32(11): 1231–1237.

Grupp M (2017) evo: Python package for the evaluation of odometry and slam. https://github.com/MichaelGrupp/evo.

Harlow K, Jang H, Barfoot TD, Kim A and Heckman C (2024) A new wave in robotics: Survey on recent mmwave radar applications in robotics. *IEEE Transactions on Robotics* .

Hartley R, Ghaffari M, Eustice RM and Grizzle JW (2020) Contact-aided invariant extended kalman filtering for robot state estimation. *International Journal of Robotics Research* 39(4): 402–430.

Hartley R, Jadidi MG, Gan L, Huang JK, Grizzle JW and Eustice RM (2018a) Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 3783–3790.

Hartley R, Mangelson J, Gan L, Jadidi MG, Walls JM, Eustice RM and Grizzle JW (2018b) Legged robot state-estimation through combined forward kinematic and preintegrated contact factors. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 4422–4429.

Hu X, Zheng L, Wu J, Geng R, Yu Y, Wei H, Tang X, Wang L, Jiao J and Liu M (2024) Paloc: Advancing slam benchmarking with prior-assisted 6-dof trajectory generation and uncertainty estimation. *IEEE/ASME Transactions on Mechatronics* 29(6): 4297–4308. DOI:10.1109/TMECH.2024.3362902.

Huang Q, Liang Y, Qiao Z, Shen S and Yin H (2024) Less is more: Physical-enhanced radar-inertial odometry. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 15966–15972.

Hug D, Bänninger P, Alzugaray I and Chli M (2022) Continuous-time stereo-inertial odometry. *IEEE Robotics and Automation Letters* 7(3): 6455–6462.

Hug D and Chli M (2020) Hyperslam: A generic and modular approach to sensor fusion and simultaneous localization and mapping in continuous-time. In: *Proceedings of the IEEE International Conference on 3D Vision*. pp. 978–986.

Jung M, Jung S and Kim A (2023) Asynchronous multiple lidar-inertial odometry using point-wise inter-lidar uncertainty propagation. *IEEE Robotics and Automation Letters* 8(7): 4211–4218.

Jung S, Yang W and Kim A (2024) Co-ral: Complementary radar-leg odometry with 4-dof optimization and rolling contact. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 13289–13296.

Kellner D, Barjenbruch M, Klappstein J, Dickmann J and Dietmayer K (2013) Instantaneous ego-motion estimation using doppler radar. In: *Proceedings of the IEEE Intelligent Transportation Systems Conference*. pp. 869–874.

Kim H, Jung M, Noh C, Jung S, Song H, Yang W, Jang H and Kim A (2025a) Hercules: Heterogeneous radar dataset in complex urban environment for multi-session radar slam. In: *Proceedings of the IEEE International Conference on Robotics*

*and Automation*. pp. 4649–4656.

Kim H, Jung M, Yang W and Kim A (2025b) Sherloc: Synchronized heterogeneous radar place recognition for cross-modal localization. *arXiv preprint arXiv:2506.15175* .

Kim JH, Hong S, Ji G, Jeon S, Hwangbo J, Oh JH and Park HW (2021) Legged robot state estimation with dynamic contact event information. *IEEE Robotics and Automation Letters* 6(4): 6733–6740.

Kim Y, Yu B, Lee EM, Kim Jh, Park Hw and Myung H (2022) Step: State estimator for legged robots using a preintegrated foot velocity factor. *IEEE Robotics and Automation Letters* 7(2): 4456–4463.

Kubelka V, Vaidis M and Pomerleau F (2022) Gravity-constrained point cloud registration. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 4873–4879.

Lang X, Chen C, Tang K, Ma Y, Lv J, Liu Y and Zuo X (2023) Coco-lic: continuous-time tightly-coupled lidar-inertial-camera odometry using non-uniform b-spline. *IEEE Robotics and Automation Letters* 8(11): 7074–7081.

Lang X, Li L, Wu C, Zhao C, Liu L, Liu Y, Lv J and Zuo X (2024) Gaussian-lic: Real-time photo-realistic slam with gaussian splatting and lidar-inertial-camera fusion. *arXiv preprint arXiv:2404.06926* .

Lang X, Lv J, Huang J, Ma Y, Liu Y and Zuo X (2022) Ctrl-vio: Continuous-time visual-inertial odometry for rolling shutter cameras. *IEEE Robotics and Automation Letters* 7(4): 11537–11544.

Li Y, Brasch N, Wang Y, Navab N and Tombari F (2020) Structure-slam: Low-drift monocular slam in indoor environments. *IEEE Robotics and Automation Letters* 5(4): 6583–6590.

Lin TY, Li T, Tong W and Ghaffari M (2023) Proprioceptive invariant robot state estimation. *arXiv preprint arXiv:2311.04320* .

Lv J, Lang X, Xu J, Wang M, Liu Y and Zuo X (2023) Continuous-time fixed-lag smoothing for lidar-inertial-camera slam. *IEEE/ASME Transactions on Mechatronics* 28(4): 2259–2270.

Michalczyk J, Jung R and Weiss S (2022) Tightly-coupled ekf-based radar-inertial odometry. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 12336–12343.

Nemiroff R, Chen K and Lopez BT (2023) Joint on-manifold gravity and accelerometer intrinsics estimation for inertially aligned mapping. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 1388–1394.

Nisticò Y, Soares JCV, Amatucci L, Fink G and Semini C (2025) Muse: A real-time multi-sensor state estimator for quadruped robots. *IEEE Robotics and Automation Letters* .

Nobili S, Camurri M, Barasuol V, Focchi M, Caldwell D, Semini C and Fallon M (2017) Heterogeneous sensor fusion for accurate state estimation of dynamic legged robots. In: *Proceedings of the Robotics: Science & Systems Conference*.

Noh C, Yang W, Jung M, Jung S and Kim A (2025) Garlio: Gravity enhanced radar-lidar-inertial odometry. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 9869–9875.

Nubert J, Tuna T, Frey J, Cadena C, Kuchenbecker KJ, Khattak S and Hutter M (2025) Holistic fusion: Task-and setup-agnostic robot localization and state estimation with factor graphs. *arXiv preprint arXiv:2504.06479* .

Ou G, Li D and Li H (2024) Leg-kilo: Robust kinematic-inertial-lidar odometry for dynamic legged robots. *IEEE Robotics and Automation Letters* .

Ovrén H and Forssén PE (2019) Trajectory representation and landmark projection for continuous-time structure from motion. *International Journal of Robotics Research* 38(6): 686–701.

Park YS, Shin YS, Kim J and Kim A (2021) 3d ego-motion estimation using low-cost mmwave radars via radar velocity factor for pose-graph slam. *IEEE Robotics and Automation Letters* 6(4): 7691–7698.

Patrikalakis NM and Maekawa T (2002) *Shape interrogation for computer aided design and manufacturing*, volume 15. Springer.

Qin T, Li P and Shen S (2018) Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics* 34(4): 1004–1020.

Ramezani M, Khosoussi K, Catt G, Moghadam P, Williams J, Borges P, Pauling F and Kottege N (2022) Wildcat: Online continuous-time 3d lidar-inertial slam. *arXiv preprint arXiv:2205.12595* .

Roston G and Krotkov E (1992) Dead reckoning navigation for walking robots. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1. pp. 607–612.

Seo DU, Lim H, Lee S and Myung H (2022) Pago-loam: Robust ground-optimized lidar odometry. In: *Proceedings of the International Conference on Ubiquitous Robots*. IEEE, pp. 1–7.

Shan T and Englot B (2018) Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 4758–4765.

Shu F, Xie Y, Rambach J, Pagani A and Stricker D (2021) Visual slam with graph-cut optimized multi-plane reconstruction. In: *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*. pp. 165–170.

Sibley G, Matthies L and Sukhatme G (2010) Sliding window filter with application to planetary landing. *Journal of field robotics* 27(5): 587–608.

Sommer C, Usenko V, Schubert D, Demmel N and Cremers D (2020) Efficient derivative computation for cumulative b-splines on lie groups. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 11148–11156.

Talbot W, Nubert J, Tuna T, Cadena C, Dümbgen F, Tordesillas J, Barfoot TD and Hutter M (2024) Continuous-time state estimation methods in robotics: A survey. *arXiv preprint arXiv:2411.03951* .

Tranzatto M, Dharmadhikari M, Bernreiter L, Camurri M, Khattak S, Mascarich F, Pfreundschuh P, Wisth D, Zimmermann S, Kulkarni M, Reijgwart V, Casseau B, Homberger T, Petris PD, Ott L, Tubby W, Waibel G, Nguyen H, Cadena C, Buchanan R, Wellhausen L, Khedekar N, Andersson O, Zhang L, Miki T, Dang T, Mattamala M, Montenegro M, Meyer K, Wu X, Briod A, Mueller M, Fallon M, Siegwart R, Hutter M and Alexis K (2022a) Team cerberus wins the darpa subterranean challenge: Technical overview and lessons learned. URL https://arxiv.org/abs/2207.04914.

Tranzatto M, Miki T, Dharmadhikari M, Bernreiter L, Kulkarni M, Mascarich F, Andersson O, Khattak S, Hutter M, Siegwart R et al. (2022b) Cerberus in the darpa subterranean challenge. *Science Robotics* 7(66): eabp9742.

Wang S, Cao F, Wang T, Shao S and Liu L (2024) Robust ground constrained slam for mobile robot with sparse-channel lidar. *IEEE Transactions on Intelligent Vehicles* .

Wang Z, Zhang L, Shen Y and Zhou Y (2023) D-LIOM: Tightly-coupled direct lidar-inertial odometry and mapping. *IEEE Transactions on Multimedia* 25: 3905–3920. DOI:10.1109/TMM.2022.3168423.

Wei X, Lv J, Sun J and Pu S (2021) Ground-slam: Ground constrained lidar slam for structured multi-floor environments. *arXiv preprint arXiv:2103.03713* .

Wisth D, Camurri M and Fallon M (2022) Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots. *IEEE Transactions on Robotics* 39(1): 309–326.

Xu W, Cai Y, He D, Lin J and Zhang F (2022) Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics* 38(4): 2053–2073.

Yang S, Zhang Z, Bokser B and Manchester Z (2023a) Multi-imu proprioceptive odometry for legged robots. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 774–779.

Yang S, Zhang Z, Fu Z and Manchester Z (2023b) Cerberus: Low-drift visual-inertial-leg odometry for agile locomotion. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. pp. 4193–4199.

Yoon DJ, Burnett K, Laconte J, Chen Y, Vhavle H, Kammel S, Reuther J and Barfoot TD (2023) Need for speed: Fast correspondence-free lidar-inertial odometry using doppler velocity. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 5304–5310.

Zhu F, Ren Y and Zhang F (2022) Robust real-time lidar-inertial initialization. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 3948–3955.

Zhuang Y, Wang B, Huai J and Li M (2023) 4d iriom: 4d imaging radar inertial odometry and mapping. *IEEE Robotics and Automation Letters* 8(6): 3246–3253.