KISS-IMU: Self-supervised Inertial Odometry with Motion-balanced Learning and Uncertainty-aware Inference

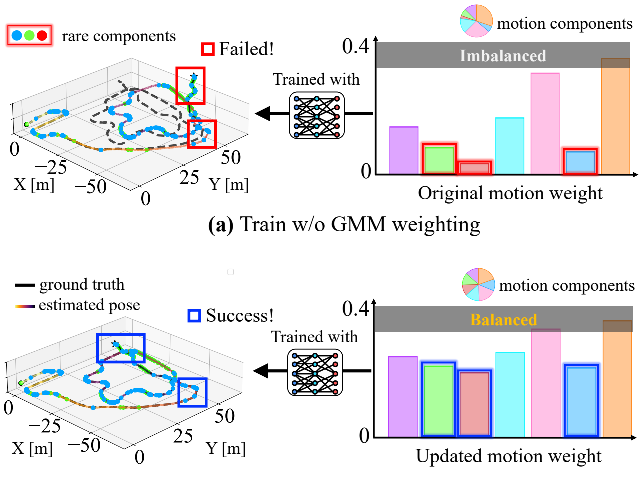
Jiwon Choi[1], Hogyun Kim[1], Geonmo Yang[1], Juhui Lee[1], and Younggun Cho[1†]

***Abstract*— Inertial measurement units (IMUs), which provide high-frequency linear acceleration and angular velocity measurements, serve as fundamental sensing modalities in robotic systems. Recent advances in deep neural networks have led to remarkable progress in inertial odometry. However, the heavy reliance on ground truth data during training fundamentally limits scalability and generalization to unseen and diverse environments. We propose *KISS-IMU*, a novel self-supervised inertial odometry framework that eliminates ground truth dependency by leveraging simple LiDAR-based ICP registration and pose graph optimization as a supervisory signal. Our approach embodies two key principles: keeping the IMU *stable* through motion-aware balanced training and keeping the IMU *strong* through uncertainty-driven adaptive weighting during inference. To evaluate performance across diverse motion patterns and scenarios, we conducted comprehensive experiments on various real-world platforms, including quadruped robots. Importantly, we train only the IMU network in a self-supervised manner, with LiDAR serving solely as a lightweight supervisory signal rather than requiring additional learnable processes. This design enables the framework to ensure robustness without relying on joint multi-modal learning or ground truth supervision. Code will be made publicly available as open-source after the review process.**

I. Introduction

High-frequency measurements of linear acceleration and angular velocity, provided by an inertial measurement unit (IMU), serve as a fundamental basis for analyzing robot motion dynamics and estimation uncertainty. Such dense temporal information enables detailed characterization of motion dynamics, which lower-rate exteroceptive sensors fail to capture, providing a reliable basis for understanding and modeling complex robot behaviors. Consequently, the IMU has become an indispensable sensing modality for understanding and modeling ego-motion across environments [1].

With recent advances in deep neural networks, inertial odometry (IO) has demonstrated remarkable capabilities by leveraging powerful learning paradigms across various robotic applications [2–12]. However, a critical bottleneck remains: the heavy reliance on ground truth data for training. While ground truth provides reliable supervision, obtaining accurate pose data in real-world environments is particularly challenging and resource-intensive.



(a) Train w/o GMM weighting

(b) Train w/ GMM weighting

Fig. 1: *KISS-IMU* performance on unseen `LAWN` sequence (trained on `Forest` sequence, both from `DiTer++` dataset). (a) Training without a Gaussian mixture model (GMM) weighting: Imbalanced motion components lead to learning biased toward dominant motion patterns, resulting in trajectory drift due to poor generalization to diverse motions. The method particularly fails in regions with motion components that are rare in the training set, where accurate pose estimation becomes challenging (red box). (b) Training with GMM weighting: Our balanced motion weighting approach achieves uniform coverage across motion components, preventing bias toward dominant patterns. The updated motion weights successfully address challenging regions (blue box). This motion-aware training demonstrates effective performance even with limited training data and better generalization to unseen environments.

Recent efforts have attempted to address this limitation through self-supervised learning approaches [10]. However, these methods still require joint training across multiple modalities (e.g., visual and inertial), resulting in tightly coupled network architectures that tie performance to specific environmental conditions. Consequently, this approach inherits the same scalability and generalization limitations as existing supervised methods, leaving these fundamental challenges unresolved.

To overcome these limitations, we propose *KISS-IMU*, a novel *self-supervised* IO framework designed to *'Keep IMU Stable and Strong'*. Our main contributions are as follows:

- **Self-supervised Learning Paradigm**: We introduce a novel training approach that eliminates ground truth dependency through selective fusion of LiDAR-based registration and pose graph optimization (PGO) to generate reliable pseudo-labels. This enables scalable deployment in unseen and diverse environments.
- **Stable IO through Motion-Aware Training**: We develop

[1] Jiwon Choi, [1] Hogyun Kim, [1] Geonmo Yang, [1] Juhui Lee, and [1†] Younggun Cho are with the Electrical and Computer Engineering, Inha University, Incheon, South Korea. `[jiwon2, hg.kim, ygm7422, dlwngml6635]@inha.edu, yg.cho@inha.ac.kr`

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (RS-2023-00302589) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-00448).

a Gaussian mixture model (GMM)-based motion clustering technique that ensures balanced learning across diverse motion patterns as illustrated in Fig. 1. This approach makes the IMU learning stable by preventing bias toward dominant motions and enhancing comprehensive motion understanding for improved generalization to unseen environments.

- **Strong IO through Uncertainty-Driven Inference**: We design an adaptive mechanism that analyzes IMU-based state uncertainty, keeping the IMU state estimation strong by dynamically adjusting weights to maintain robust performance under varying conditions.
- **Comprehensive Evaluation**: Through extensive evaluation across multiple datasets under varying training data percentages, we demonstrate competitive generalization capabilities with minimal training data compared to existing state-of-the-art approaches.

To the best of our knowledge, this work is the first framework that learns IO in a self-supervised manner using non-learnable supervision, distinguishing it from existing approaches that require either ground truth supervision or learnable supervision.

II. Related Works

A. Inertial Odometry For Complex Motions

To address the challenges of complex motions in domains such as pedestrians, drones, and quadruped robots, inertial odometry (IO) has been extensively advanced through data-driven methods. In pedestrian scenarios, IONet [2] directly regressed motion using neural networks, and this approach was later improved by RoNIN [3] through the use of multiple network architectures. However, both methods suffered from limited interpretability, as measurement noise was not explicitly modeled. To overcome this limitation, TLIO [4] and LLIO [5] predicted uncertainty and integrated it with extended Kalman filter (EKF) for robust performance.

Unlike pedestrian IO, where periodic gait patterns can be effectively learned, unmanned aerial vehicle (UAV) domains present a more challenging environment without such regularities. To better capture real-world dynamics, Drone IO [7] predicted displacement and improved measurement updates, rather than depending on assumptions of simplified motion models such as AI-IMU [6]. AirIMU [8] jointly modeled IMU noise and uncertainty, employing PGO with uncertainty-aware covariance injection. AirIO [9] further analyzed the representational richness of motion in the body frame and developed a tightly coupled EKF framework.

Similarly, the field expanded to include quadruped robots, which presented unique challenges. Despite their promising mobility, their dynamic gaits introduced vibrations across all axes and impact-induced oscillations from ground contact, making motion analysis particularly challenging. While approaches such as [11] addressed quadruped robot state estimation, these complex dynamics still demanded further investigation.

Despite remarkable progress in learnable IO, existing methods still face a critical limitation: they rely on supervised learning with ground truth data requiring centimeter-level accuracy. This dependency necessitates motion capture systems or other high-precision setups. Such restricted settings exacerbate the tendency of data-driven methods to overfit specific motion patterns and environments observed during training. This fundamentally hinders generalization to new scenarios and diverse motions. iSLAM [10] alleviated this dependency through self-supervised learning, but its reliance on a learnable visual modality left generalization challenges unresolved.

B. IMU Motion Analysis for Enhanced State Estimation

Sophisticated IMU motion analysis has proven crucial for robust state estimation. FAST-LIO2 [13] established efficient LiDAR-inertial fusion but suffered from degradation under aggressive motions with IMU saturation. Point-LIO [14] addressed this by treating IMU measurements as signals and conducting separate saturation checks for each channel, enabling accurate localization during extreme motions. TartanIMU [12] demonstrated how learned motion pattern clustering across platforms could substantially improve IMU-based state estimation. The cluster separation in feature space revealed the model's ability to capture motion-specific dynamics.

These works have consistently shown that motion analysis techniques—decomposing complex patterns and adapting to motion dynamics—directly impact state estimation performance. Inspired by these developments, our approach adapts similar motion analysis principles for self-supervised learning scenarios where ground truth supervision is unavailable.

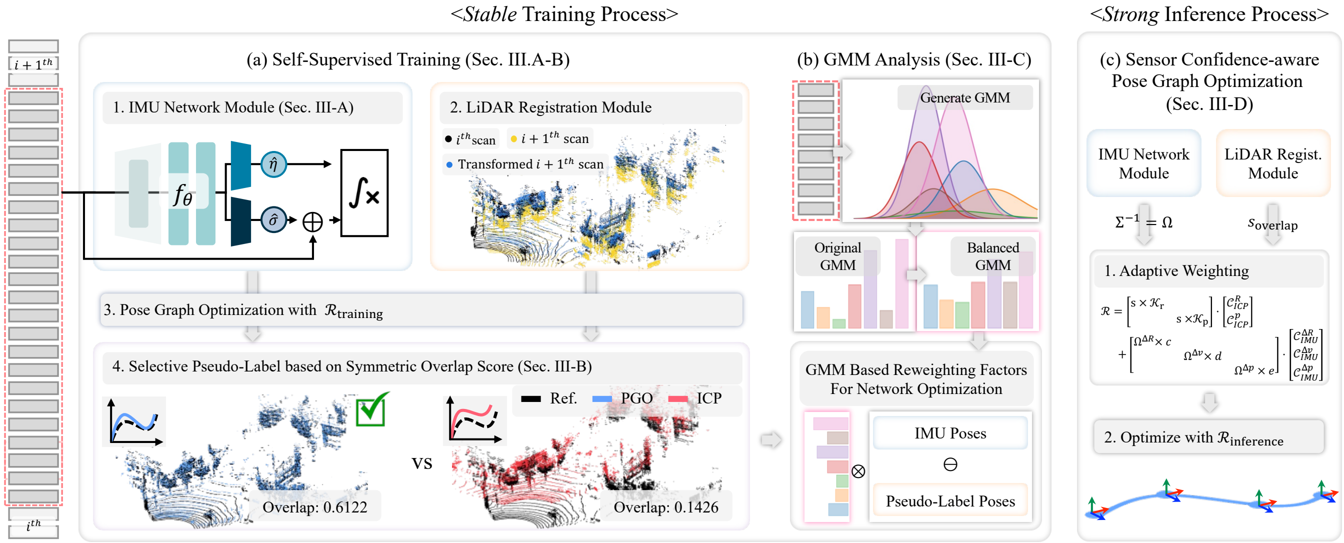
III. KISS-IMU: Self-supervised, Stable, and Strong Inertial Odometry

We propose *KISS-IMU*, a *self-supervised* framework that learns *stable* and *strong* inertial odometry (IO) without ground truth supervision, as illustrated in Fig. 2. Our approach eliminates ground truth dependency by leveraging LiDAR registration, iterated closest point (ICP) [15, 16], and PGO to generate reliable pseudo-labels from segment-wise relative motion ($\Delta\mathbf{R}, \Delta\mathbf{v}, \Delta\mathbf{p}$); see Sec. III-B. To achieve *stable* IO, we employ GMM-based reweighting during training, which addresses motion distribution imbalances and ensures comprehensive learning across all motion patterns, avoiding bias toward dominant motions; see Sec. III-C. For *strong* IO, our network jointly estimates correction parameters and uncertainty measures for IMU measurements, enabling adaptive weighting of PGO during inference under varying conditions; see Sec. III-D.

A. IMU Integration and Covariance Propagation

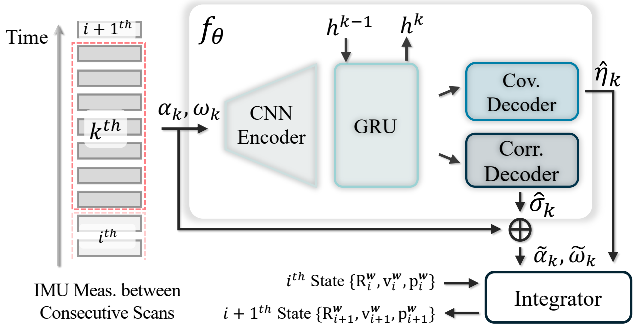
Our framework leverages the temporal correspondence between IMU and LiDAR measurements to utilize self-supervised learning. For consecutive LiDAR scans between i^{th} time t_i and $i+1^{th}$ time t_{i+1}, we aggregate IMU measurements $\mathcal{M}_{i,i+1} = \{\mathbf{m}_k\}_{k=1}^{K}$ within the time interval $[t_i, t_{i+1}]$, where the k^{th} measurement $\mathbf{m}_k = [\boldsymbol{\omega}_k, \boldsymbol{\alpha}_k]^T \in \mathbb{R}^6$ comprises angular velocity, i.e., $\boldsymbol{\omega}_k \in \mathbb{R}^3$, and linear acceleration, i.e., $\boldsymbol{\alpha}_k \in \mathbb{R}^3$.

Unlike learning-free methods that rely on predefined noise models, we utilize a neural network, f_θ (illustrated in Fig. 3),



<Stable Training Process>
<Strong Inference Process>
i + 1th
ith
(a) Self-Supervised Training (Sec. III.A-B)
1. IMU Network Module (Sec. III-A)
f_θ
$\hat{\eta}$
$\hat{\sigma}$
$\int\times$
2. LiDAR Registration Module
ith scan
i + 1th scan
Transformed i + 1th scan
3. Pose Graph Optimization with $\mathcal{R}_{\text{training}}$
4. Selective Pseudo-Label based on Symmetric Overlap Score (Sec. III-B)
Ref.
PGO
ICP
vs
Overlap: 0.6122
Overlap: 0.1426
(b) GMM Analysis (Sec. III-C)
Generate GMM
Original GMM
Balanced GMM
GMM Based Reweighting Factors For Network Optimization
IMU Poses
⊖
Pseudo-Label Poses
⊗
(c) Sensor Confidence-aware Pose Graph Optimization (Sec. III-D)
IMU Network Module
LiDAR Regist. Module
$\Sigma^{-1} = \Omega$
s_{overlap}
1. Adaptive Weighting
2. Optimize with $\mathcal{R}_{\text{inference}}$

Fig. 2: Our inertial odometry (IO) framework embodies the *Keep IMU Stable and Strong* philosophy through three main components: (a) Self-supervised training employs an IMU network module to predict corrections and uncertainties, while a LiDAR registration module generates geometric constraints. Pose graph optimization (PGO) combines both modalities, followed by selective pseudo-label generation based on symmetric overlap scores to ensure reliable supervision without ground truth. (b) GMM analysis demonstrates how we achieve stable IO through motion pattern clustering. The original GMM distribution reveals imbalanced motion patterns, while our balancing strategy achieves uniform coverage across diverse motions. GMM-based reweighting factors guide network optimization by emphasizing underrepresented motion components, ensuring comprehensive motion learning. (c) Sensor confidence-aware PGO shows how we maintain strong IO through adaptive weighting. Learned uncertainties from IMU and LiDAR modules enable dynamic confidence adjustment during inference, maintaining robust performance across varying motion and sensor conditions.



Time
i + 1th
kth
ith
IMU Meas. between Consecutive Scans
f_θ
h^{k-1}
h^k
α_k, ω_k
CNN Encoder
GRU
Cov. Decoder
Corr. Decoder
$\hat{\eta}_k$
$\hat{\sigma}_k$
$\tilde{\alpha}_k, \tilde{\omega}_k$
ith State $\{R_i^W, v_i^W, p_i^W\}$
i + 1th State $\{R_{i+1}^W, v_{i+1}^W, p_{i+1}^W\}$
Integrator

Fig. 3: Our proposed network architecture. A CNN-GRU encoder processes raw IMU measurements (i.e., $\mathbf{m}_k = [\boldsymbol{\omega}_k, \boldsymbol{\alpha}_k]^T$) from measurement set $\mathcal{M}_{i,i+1}$ to extract features, which are then decoded to estimate learned corrections $\hat{\boldsymbol{\sigma}}_k$ and uncertainties $\hat{\boldsymbol{\eta}}_k$. These outputs are combined with the previous state through an integrator module to produce corrected measurements (i.e., $\tilde{\mathbf{m}}_k = \mathbf{m}_k + \hat{\boldsymbol{\sigma}}_k$) and their associated uncertainties (i.e., $\hat{\boldsymbol{\eta}}_k$).

to adaptively estimate and learn both measurement-specific corrections and uncertainties as follows:

$$\{\hat{\boldsymbol{\sigma}}_k, \hat{\boldsymbol{\eta}}_k\}_{k=1}^{K} = f_\theta(\mathcal{M}_{i,i+1}), \tag{1}$$

where $\hat{\boldsymbol{\sigma}}_k = [\hat{\boldsymbol{\sigma}}_k^{\boldsymbol{\omega}}, \hat{\boldsymbol{\sigma}}_k^{\boldsymbol{\alpha}}]^T$ denotes the learned corrections and $\hat{\boldsymbol{\eta}}_k = [\hat{\boldsymbol{\eta}}_k^{\boldsymbol{\omega}}, \hat{\boldsymbol{\eta}}_k^{\boldsymbol{\alpha}}]^T$ denotes the learned uncertainties for angular velocity and acceleration, respectively. Here, $\hat{\boldsymbol{\sigma}}_k^{\boldsymbol{\omega}}, \hat{\boldsymbol{\sigma}}_k^{\boldsymbol{\alpha}}, \hat{\boldsymbol{\eta}}_k^{\boldsymbol{\omega}}, \hat{\boldsymbol{\eta}}_k^{\boldsymbol{\alpha}} \in \mathbb{R}^3$, resulting in $\hat{\boldsymbol{\sigma}}_k, \hat{\boldsymbol{\eta}}_k \in \mathbb{R}^6$. Through the learned corrections, we obtain the corrected measurements as follows:

$$\tilde{\mathbf{m}}_k = \mathbf{m}_k + \hat{\boldsymbol{\sigma}}_k. \tag{2}$$

Following the preintegration framework [1], we perform integration consistent with this temporal correspondence to compute the relative transformation between consecutive frames. The corrected measurements $\tilde{\mathbf{m}}_k$ are then integrated to obtain the preintegrated quantities—rotation $\Delta\mathbf{R}_{i,i+1}^{\text{IMU}} \in SO(3)$, velocity $\Delta\mathbf{v}_{i,i+1}^{\text{IMU}} \in \mathbb{R}^3$, and position $\Delta\mathbf{p}_{i,i+1}^{\text{IMU}} \in \mathbb{R}^3$ as follows:

$$\Delta\mathbf{R}_{i,i+1}^{\text{IMU}} = \prod_{k=1}^{K} \operatorname{Exp}(\tilde{\boldsymbol{\omega}}_k \Delta t), \tag{3a}$$

$$\Delta\mathbf{v}_{i,i+1}^{\text{IMU}} = \sum_{k=1}^{K} \Delta\mathbf{R}_{i,k}^{\text{IMU}} \tilde{\boldsymbol{\alpha}}_k \Delta t, \tag{3b}$$

$$\Delta\mathbf{p}_{i,i+1}^{\text{IMU}} = \sum_{k=1}^{K} \left[\Delta\mathbf{v}_{i,k}^{\text{IMU}} \Delta t + \frac{1}{2} \Delta\mathbf{R}_{i,k}^{\text{IMU}} \tilde{\boldsymbol{\alpha}}_k \Delta t^2 \right], \tag{3c}$$

where $\operatorname{Exp}(\cdot)$ denotes the matrix exponential for rotation, Δt is the time step between consecutive IMU measurements, and $\Delta\mathbf{R}_{i,k}^{\text{IMU}}$ and $\Delta\mathbf{v}_{i,k}^{\text{IMU}}$ represent the intermediate rotation and velocity from time t_i to t_k. To propagate the IMU state forward in time, we transform the preintegrated measurements to the world frame at time t_{i+1}, which is estimated as follows:

$$\hat{\mathbf{R}}_{i+1}^{W} = \mathbf{R}_i^{W} \Delta\mathbf{R}_{i,i+1}^{\text{IMU}}, \tag{4a}$$

$$\hat{\mathbf{v}}_{i+1}^{W} = \mathbf{v}_i^{W} + \mathbf{g}^{W} \Delta t + \mathbf{R}_i^{W} \Delta\mathbf{v}_{i,i+1}^{\text{IMU}}, \tag{4b}$$

$$\hat{\mathbf{p}}_{i+1}^{W} = \mathbf{p}_i^{W} + \mathbf{v}_i^{W} \Delta t + \frac{1}{2}\mathbf{g}^{W} \Delta t^2 + \mathbf{R}_i^{W} \Delta\mathbf{p}_{i,i+1}^{\text{IMU}}, \tag{4c}$$

where the superscript W denotes quantities in the world frame, $\mathbf{R}_i^W$, $\mathbf{v}_i^W$, and $\mathbf{p}_i^W$ represent the rotation, velocity, and position at the previous time t_i in the world frame, $\hat{\mathbf{R}}_{i+1}^W$, $\hat{\mathbf{v}}_{i+1}^W$, and $\hat{\mathbf{p}}_{i+1}^W$ are the estimated states at time t_{i+1}, $\mathbf{g}^W \in \mathbb{R}^3$ is the gravity vector, and $\Delta t = t_{i+1} - t_i$.

A distinguishing feature of our approach is the propagation of learned uncertainties from (1) through the integration process.

Following [8], the state covariance evolves iteratively for each measurement step as follows:

$$\boldsymbol{\Sigma}_{k+1} = \mathbf{A}_k \boldsymbol{\Sigma}_k \mathbf{A}_k^T + \mathbf{B}_{\boldsymbol{\omega},k}\, \mathrm{diag}(\hat{\boldsymbol{\eta}}_k^{\boldsymbol{\omega}})\, \mathbf{B}_{\boldsymbol{\omega},k}^T + \mathbf{B}_{\boldsymbol{\alpha},k}\, \mathrm{diag}(\hat{\boldsymbol{\eta}}_k^{\boldsymbol{\alpha}})\, \mathbf{B}_{\boldsymbol{\alpha},k}^T, \quad (5)$$

where $\mathbf{A}_k$ is the state transition matrix and $\mathbf{B}_{\boldsymbol{\omega},k}$, $\mathbf{B}_{\boldsymbol{\alpha},k}$ are the noise propagation matrices that map measurement uncertainties to the state space. These matrices follow the standard IMU error-state formulation detailed in [1, 8]. The covariance propagation starts with initial covariance $\boldsymbol{\Sigma}_0$ and iteratively accumulates the learned uncertainties $\hat{\boldsymbol{\eta}}_k^{\boldsymbol{\omega}}$ and $\hat{\boldsymbol{\eta}}_k^{\boldsymbol{\alpha}}$ for each corrected measurement $\tilde{\mathbf{m}}_k$ from (2). After processing all $k \in \{1, ..., K\}$ measurements within the interval $[t_i, t_{i+1}]$, we obtain the corrected propagation covariance $\boldsymbol{\Sigma}_{i,i+1} = \boldsymbol{\Sigma}_K$.

This learned and corrected uncertainty $\boldsymbol{\Sigma}_{i,i+1}$ plays two critical roles in our framework. During training, it enables uncertainty-aware learning (for *stable* IO in Section III-C). During inference, it determines the confidence of IMU constraints in PGO (for *strong* IO in Section III-D).

B. Selective Pseudo-Labels for Self-Supervised Learning

Our self-supervised framework eliminates ground truth dependency by generating reliable pseudo-labels from the ICP or its variants [15, 16]. To ensure training stability, we dynamically select between ICP and PGO pseudo-labels based on geometric consistency.

First, we obtain the relative transformation $\Delta\mathbf{P}_{i,i+1}^{\text{ICP}} \in \mathrm{SE}(3)$ between consecutive LiDAR scans at times t_i and t_{i+1}. While ICP provides robust constraints, we enhance system robustness by addressing (i) potential degradation in geometrically degenerate environments and (ii) local optimization limitations that may affect global trajectory consistency. To achieve more reliable pseudo-labels, we formulate a PGO that jointly considers both LiDAR and IMU constraints over N_{node} LiDAR frames as follows:

$$\mathcal{C}_{\text{ICP}}^{\Delta\mathbf{P}} \coloneqq \sum_{i=1}^{N_{\text{node}}-1} \mathrm{Log}\left(\Delta\mathbf{P}_{i,i+1}^{\text{ICP}} \boxminus \Delta\mathbf{P}_{i,i+1}^{\text{PGO}}\right), \quad (6a)$$

$$\mathcal{C}_{\text{IMU}}^{\Delta\mathbf{R}} \coloneqq \sum_{i=1}^{N_{\text{node}}-1} \mathrm{Log}\left(\Delta\mathbf{R}_{i,i+1}^{\text{IMU}} \boxminus \Delta\mathbf{R}_{i,i+1}^{\text{PGO}}\right), \quad (6b)$$

$$\mathcal{C}_{\text{IMU}}^{\Delta\mathbf{v}} \coloneqq \sum_{i=1}^{N_{\text{node}}-1} \left\|\Delta\mathbf{v}_{i,i+1}^{\text{IMU}} - \Delta\mathbf{v}_{i,i+1}^{\text{PGO}}\right\|_2, \quad (6c)$$

$$\mathcal{C}_{\text{IMU}}^{\Delta\mathbf{p}} \coloneqq \sum_{i=1}^{N_{\text{node}}-1} \left\|\Delta\mathbf{p}_{i,i+1}^{\text{IMU}} - \Delta\mathbf{p}_{i,i+1}^{\text{PGO}}\right\|_2, \quad (6d)$$

where $\Delta\mathbf{P}_{i,i+1}^{\text{ICP}}$, $\Delta\mathbf{P}_{i,i+1}^{\text{PGO}} \in \mathrm{SE}(3)$ represent relative poses from ICP and PGO, $\Delta\mathbf{R}_{i,i+1}^{\text{IMU}}, \Delta\mathbf{v}_{i,i+1}^{\text{IMU}}, \Delta\mathbf{p}_{i,i+1}^{\text{IMU}}$ are preintegrated IMU quantities from (3), and $\boxminus$, $\mathrm{Log}(\,\cdot\,)$ denote standard manifold operations. Finally, we solve the total optimization objective using the Levenberg-Marquardt (LM) algorithm in PyPose [17], which combines all constraints as follows:

$$\mathcal{R}_{\text{training}} = w_1\mathcal{C}_{\text{ICP}}^{\Delta\mathbf{P}} + w_2\mathcal{C}_{\text{IMU}}^{\Delta\mathbf{R}} + w_3\mathcal{C}_{\text{IMU}}^{\Delta\mathbf{v}} + w_4\mathcal{C}_{\text{IMU}}^{\Delta\mathbf{p}}, \quad (7)$$

where $\mathbf{w} = [w_1, w_2, w_3, w_4]$ are fixed weights that balance the contribution of each constraint type.

When conditions (i) and (ii) are not met, ICP provides sufficiently reliable pseudo-labels. To ensure optimal reliability and select the most trustworthy measurements across all scenarios, we evaluate both ICP and PGO estimates based on their geometric consistency. For this selective fusion, we assess the quality of each estimate using the symmetric overlap score from [18] as follows:

$$s_{\text{overlap}}(\Delta\mathbf{P}) = 0.5\left[\mathbf{O}(\mathcal{P}_i, \mathcal{P}_{i+1}') + \mathbf{O}(\mathcal{P}_{i+1}, \mathcal{P}_i')\right], \quad (8)$$

where $\mathbf{O}(\,\cdot\,,\,\cdot\,)$ measures the overlap ratio based on nearest neighbor distances, $\mathcal{P}_{i+1}'$ and $\mathcal{P}_i'$ represent the transformed point clouds from $\Delta\mathbf{P}$. In this case, we evaluate two candidates: $\Delta\mathbf{P}^{\text{ICP}}$ and $\Delta\mathbf{P}^{\text{PGO}}$, selecting the transformation that yields the higher overlap score. Fig. 2 (a). 4 demonstrates this selection process, where the higher overlap score indicates better geometric alignment between consecutive point clouds, providing a more trustworthy reference pose for supervision. Moreover, this selective mechanism helps prevent the network from reinforcing its own errors by ensuring supervision comes from the most geometrically consistent source.

We evaluate both $s_{\text{ICP}} = s_{\text{overlap}}(\Delta\mathbf{P}_{i,i+1}^{\text{ICP}})$ and $s_{\text{PGO}} = s_{\text{overlap}}(\Delta\mathbf{P}_{i,i+1}^{\text{PGO}})$, then select the transformation with the higher overlap score as our pseudo-label. This selection process is applied to all consecutive frames, providing selective and reliable pseudo-labels for self-supervised learning without ground truth. The selected pseudo-label $\Delta\mathbf{P}_{i,i+1} = \{\Delta\mathbf{R}_{i,i+1}^{\text{pseudo}}, \Delta\mathbf{p}_{i,i+1}^{\text{pseudo}}\}$ represents the relative transformation between consecutive frames. To derive the supervisory states for training, we first propagate the pose: given $\mathbf{P}_i^W = \{\mathbf{R}_i^W, \mathbf{p}_i^W\}$ at time t_i, the pose at time t_{i+1} is computed as $\mathbf{P}_{i+1}^W = \mathbf{P}_i^W \cdot \Delta\mathbf{P}_{i,i+1}$. The velocity in the world frame is then derived from the position difference: $\mathbf{v}_{i+1}^W = (\mathbf{p}_{i+1}^W - \mathbf{p}_i^W)/\Delta t$, where Δt is in (4). These pseudo-label states $\{\mathbf{R}_{i+1}^W, \mathbf{v}_{i+1}^W, \mathbf{p}_{i+1}^W\}$ serve as supervision for the IMU-predicted states in our loss functions (Section III-C), enabling the network to learn accurate IO without ground truth.

C. Loss Functions and GMM-Based Training Strategy

1) Loss Functions: We first define the pose-level loss functions for rotation, velocity, and position errors between the predicted states $\{\hat{\mathbf{R}}_{i+1}^W, \hat{\mathbf{v}}_{i+1}^W, \hat{\mathbf{p}}_{i+1}^W\}$ and pseudo-labels $\{\mathbf{R}_{i+1}^W, \mathbf{v}_{i+1}^W, \mathbf{p}_{i+1}^W\}$ as follows:

$$\mathcal{L}_r = \left\|\log(\mathbf{R}_{i+1}^W \boxminus \hat{\mathbf{R}}_{i+1}^W)\right\|_2, \quad (9a)$$

$$\mathcal{L}_v = \left\|\mathbf{v}_{i+1}^W - \hat{\mathbf{v}}_{i+1}^W\right\|_2, \quad (9b)$$

$$\mathcal{L}_p = \left\|\mathbf{p}_{i+1}^W - \hat{\mathbf{p}}_{i+1}^W\right\|_2. \quad (9c)$$

Since each pose estimation inherently contains uncertainty, we additionally define uncertainty-aware loss functions that incorporate the learned uncertainties, derived from (5), to handle prediction reliability better, following the formulations in [8, 19]:

$$\mathcal{L}_r^{\text{cov}} = \frac{1}{2}\left(\left\|\log(\mathbf{R}_{i+1}^W \boxminus \hat{\mathbf{R}}_{i+1}^W)\right\|_{\boldsymbol{\Sigma}_{i,i+1}^r}^2 + \ln(\det(\boldsymbol{\Sigma}_{i,i+1}^r))\right), \quad (10a)$$

$$\mathcal{L}_v^{\text{cov}} = \frac{1}{2}\left(\|\mathbf{v}_{i+1}^W - \hat{\mathbf{v}}_{i+1}^W\|^2_{\boldsymbol{\Sigma}^v_{i,i+1}} + \ln(\det(\boldsymbol{\Sigma}^v_{i,i+1}))\right), \tag{10b}$$

$$\mathcal{L}_p^{\text{cov}} = \frac{1}{2}\left(\|\mathbf{p}_{i+1}^W - \hat{\mathbf{p}}_{i+1}^W\|^2_{\boldsymbol{\Sigma}^p_{i,i+1}} + \ln(\det(\boldsymbol{\Sigma}^p_{i,i+1}))\right), \tag{10c}$$

where $\|\cdot\|^2_{\boldsymbol{\Sigma}}$ denotes the Mahalanobis distance, and $\boldsymbol{\Sigma}^r_{i,i+1}$, $\boldsymbol{\Sigma}^v_{i,i+1}$, $\boldsymbol{\Sigma}^p_{i,i+1}$ represent the rotation, velocity, and position blocks of the propagated covariance matrix. Then, the final loss function combines all components as follows:

$$\mathcal{L}_{\text{total}} = \underbrace{\mathcal{L}_r + \mathcal{L}_v + \mathcal{L}_p}_{\text{pose-level loss}} + \underbrace{\varepsilon(\mathcal{L}_r^{\text{cov}} + \mathcal{L}_v^{\text{cov}} + \mathcal{L}_p^{\text{cov}})}_{\text{uncertainty-aware loss}}, \tag{11}$$

where ε is a scaling factor.

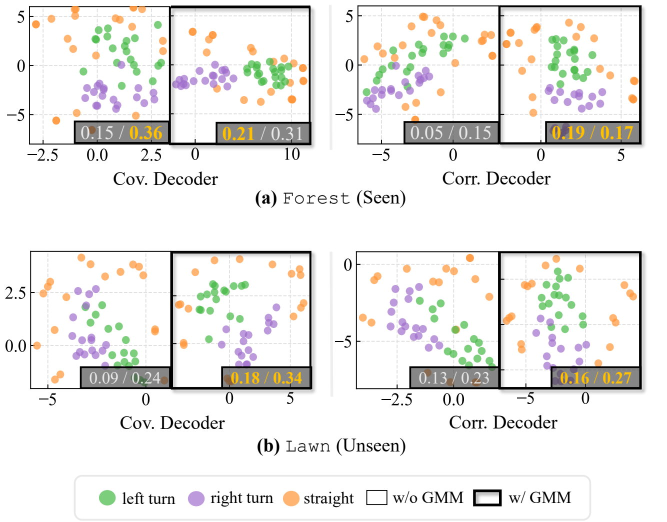
This comprehensive loss formulation allows the network to simultaneously learn accurate pose estimation and reliable uncertainty quantification, yield robust state representation that adapts to varying motion conditions and measurement quality.

2) GMM-Based Motion Decomposition: While (11) functions effectively incorporate uncertainty, they alone cannot address the fundamental challenge of motion distribution imbalance in sequential datasets. Sequential datasets typically contain uneven distributions of motion patterns, with dominant behaviors (e.g., straight-line motion in driving datasets) vastly outnumbering rare but critical maneuvers (e.g., sharp yaw rotations or sudden accelerations). Therefore, our goal is to achieve robust IMU learning that ensures diverse motion patterns receive balanced attention during training. This prevents overfitting to common motions while maintaining performance on rare but essential maneuvers as shown in Fig. 2 (b).

To achieve comprehensive motion coverage, we first define IMU windows of duration Δt_w (typically 0.2 sec) and extract motion descriptors for each window. For each window, we extract motion features including statistics of angular velocity, magnitudes of acceleration, and their temporal variations. These features are standardized to form a motion descriptor $\mathbf{z}_n \in \mathbb{R}^n$. We then analyze the motion distribution of the entire training dataset using these descriptors with GMM similar to [12]. We fit a GMM to the collection of motion descriptors from all training sequences, $p(\mathbf{z}_n) = \sum_{g=1}^{G} \pi_g \mathcal{N}(\mathbf{z}_n \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$, where G is the number of components, and π_g, $\boldsymbol{\mu}_g$, $\boldsymbol{\Sigma}_g$ represent the mixture weights, means, and covariances respectively. Following [20], we select the optimal number of components G using the Bayesian information criterion (BIC) to balance model complexity and fit quality.

3) Motion-Aware Reweighting Strategy: Inspired by the class-balanced weighting strategy [21], we extend this concept to address motion imbalance in IMU data. While they tackle long-tailed distributions in visual recognition, we adapt their re-weighting principle to balance diverse motion patterns in IO.

Let $w_g = \frac{1-\beta}{1-\beta^{N_g}}$ denote the raw reweighting factor for component g, where $\beta \in (0, 1)$ controls the re-weighting strength. Here, $N_g = \sum_{g=1}^{N} \gamma_g(\mathbf{z}_n)$ represents the frequency of each component across the training set, and $\gamma_g(\mathbf{z}_n) = \frac{\pi_g \mathcal{N}(\mathbf{z}_n \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)}{\sum_{g=1}^{G} \pi_g \mathcal{N}(\mathbf{z}_n \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)}$ denotes the posterior probability of component assignment. The normalized weight is then computed



(a) `Forest` (Seen)

(b) `Lawn` (Unseen)

Fig. 4: t-SNE visualization of motion pattern clustering with quantitative evaluation using silhouette score [22] / adjusted rand index [23]. We randomly sample motion data from each GMM-determined label across three motion patterns (left turn, right turn, straight), with motion types identified through ground truth velocity analysis, in seen (`Forest`) and unseen (`LAWN`) environments. Without GMM balance, baseline clustering performance shows standard feature separation. Our GMM-based balanced training achieves improved clustering metrics, demonstrating enhanced inter-class separation and intra-class compactness. The quantitative improvements demonstrate that motion-aware reweighting produces better generalization performance in embedding vector representations.

as $\tilde{w}_g = \frac{w_g}{\frac{1}{G}\sum_{g=1}^{G} w_g}$. Components with lower frequency N_g receive higher weights $\tilde{w}_g$, ensuring rare motion patterns are adequately represented during training.

During training, for each IMU window with motion descriptor $\mathbf{z}_n$, we compute its motion-aware weight as follows:

$$w_n^{\text{GMM}} = \sum_{g=1}^{G} \gamma_g(\mathbf{z}_n)\tilde{w}_g. \tag{12}$$

This weight reflects how much each sample should contribute to the loss based on its motion component assignment and the rarity of that component. Extending (11), the final training loss integrates these motion-aware weights as follows:

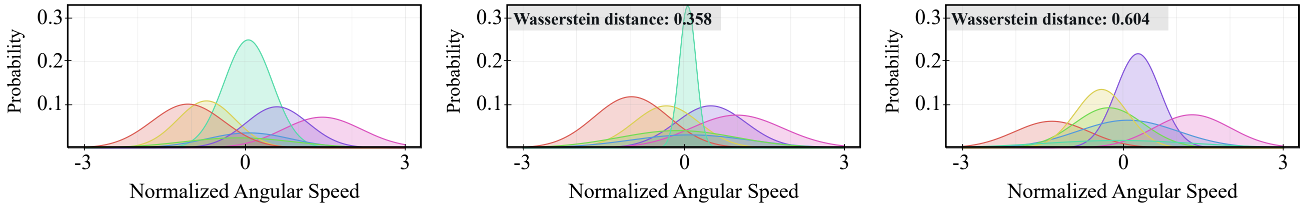
$$\mathcal{L}^{\star}_{\text{total}} = \frac{1}{|\mathcal{B}|}\sum_{n\in\mathcal{B}} w_n^{\text{GMM}} \cdot \mathcal{L}_{\text{total}}, \tag{13}$$

where $\mathcal{B}$ is the mini-batch and $|\cdot|$ denotes the cardinality of the set. The per-sample loss combines both motion-aware pose-level and motion- and uncertainty-aware terms as follows:

$$\begin{aligned}\mathcal{L}^{\star}_{\text{total}} = & \underbrace{\lambda_r \mathcal{L}_{r,n} + \lambda_v \mathcal{L}_{v,n} + \lambda_p \mathcal{L}_{p,n}}_{\text{motion-aware pose-level loss}} \\ & + \underbrace{\lambda_r \mathcal{L}^{\text{cov}}_{r,n} + \lambda_v \mathcal{L}^{\text{cov}}_{v,n} + \lambda_p \mathcal{L}^{\text{cov}}_{p,n}}_{\text{motion- and uncertainty-aware loss}},\end{aligned} \tag{14}$$

where $\lambda_{r,v,p} = w_n^{\text{GMM}}$ for motion-aware standard terms and $\lambda_{r,v,p} = \varepsilon \cdot w_n^{\text{GMM}}$ for motion- and uncertainty-aware terms.

This motion-aware training strategy ensures balanced learning across diverse motion patterns, emphasizing motion diversity over quantity to prevent the network from overfitting to



(a) `Forest` **(b)** `LAWN` **(c)** `PARK`

Fig. 5: Motion pattern analysis using GMM across `DiTer++` sequences. Using Bayesian information criterion [20], we determine the optimal number of components as G=7. GMM distributions of normalized angular speed reveal motion characteristics for (a) `Forest`, (b) `LAWN`, and (c) `PARK` sequences. Lower Wasserstein distances between `Forest` and `LAWN` compared to `PARK` indicate better motion similarity.

dominant behaviors while neglecting critical but rare maneuvers. By integrating motion-balanced weighting with uncertainty-aware losses, our approach achieves stable IO through comprehensive motion coverage rather than simply increasing training data volume. As illustrated in Fig. 4, our strategy demonstrates improved feature clustering and separation compared to the baseline, indicating enhanced generalization performance across diverse motion patterns.

D. Sensor Confidence-Aware Adaptive Weights for PGO

To achieve strong IO during inference, we enhance PGO through adaptive constraint weighting based on sensor confidence. While training employs fixed weights for strong IO, inference benefits from dynamic weighting that adapts to the varying reliability levels of measurements, maintaining robust performance across different motion scenarios.

For adaptive LiDAR constraints, the symmetric overlap score s_{overlap} from (8) inherently quantifies registration confidence—higher overlap indicates more reliable geometric alignment. For adaptive IMU constraints, the propagated covariance $\mathbf{\Sigma}_{i,i+1}$ from (5) represents measurement uncertainty, which we convert to confidence through the information matrix $\mathbf{\Omega}_{i,i+1} = \mathbf{\Sigma}_{i,i+1}^{-1}$. We incorporate these confidence measures into the PGO framework. In sum, the adaptive PGO cost function becomes as follows:

$$\begin{aligned}\mathcal{R}_{\text{inference}} &= s_{\text{overlap}} \cdot (\kappa_r \|\mathcal{C}_{\text{ICP}}^{\Delta \mathbf{R}}\|^2 + \kappa_p \|\mathcal{C}_{\text{ICP}}^{\Delta \mathbf{p}}\|^2) \\ &\quad + \tau_R \|\mathcal{C}_{\text{IMU}}^{\Delta \mathbf{R}}\|_{\mathbf{\Omega}^{\Delta \mathbf{R}}}^2 + \tau_v \|\mathcal{C}_{\text{IMU}}^{\Delta \mathbf{v}}\|_{\mathbf{\Omega}^{\Delta \mathbf{v}}}^2 + \tau_p \|\mathcal{C}_{\text{IMU}}^{\Delta \mathbf{p}}\|_{\mathbf{\Omega}^{\Delta \mathbf{p}}}^2,\end{aligned} \tag{15}$$

where κ_r, κ_p are LiDAR sensor's scaling factors and τ_R, τ_v, τ_p are IMU sensor's scaling factors. Finally, by solving this adaptive cost function with the LM optimizer, we achieve strong IO that dynamically adapts to varying sensor reliability and motion conditions.

IV. Experimental Results

A. Experimental Setup

We conducted experiments on a desktop equipped with an AMD EPYC 7513 32-core processor (3.3 GHz) and an NVIDIA RTX 3090 GPU. All experiments were performed using PyTorch with CUDA acceleration for neural network training and inference.

1) Datasets: We evaluated our method on `Botanic Garden` [24], `DiTer++` [25], and an `In-House` dataset. `Botanic Garden` provided natural environments captured with a wheeled robot equipped with Velodyne VLP-16 LiDAR and Xsens MTI-680G IMU (9 DoF), which presented primarily environmental challenges. `DiTer++` featured a Unitree GO2 quadrupedal robot with diverse motion patterns across various terrains captured using Ouster OS-1 64 / 128 LiDAR and sensor-integrated IMU (6 DoF), which combined moderate environmental and motion complexities. Our `In-House` dataset used a larger Unitree B2 quadrupedal robot in crater-filled planetary-analogous terrain captured with Ouster OS-1 32 LiDAR and sensor-integrated IMU (6 DoF), exhibiting extreme motion variations and harsh environmental conditions representing the highest complexity level. The complexity of each dataset was denoted by stars (★) in our evaluation tables.

2) Evaluation Metrics: We evaluated our method using two trajectory accuracy metrics: absolute pose error (APE) and relative pose error (RPE). In particular, to focus on motion errors rather than accumulated drift errors, we followed a similar RPE evaluation protocol from [8] as follows: We divided trajectories into intervals of fixed duration $\Delta t = 0.2$ sec, assigning identical initial conditions (i.e., initial pose and velocity) to all methods at the beginning of each interval, and evaluated the cumulative error over that time period as follows:

- APE = $\frac{1}{N}\sum_{i=1}^{N} \left\|\mathbf{p}_i^{\text{est}} - \mathbf{p}_i^{\text{gt}}\right\|_2$
- RPE = $\frac{1}{N}\sum_{i=1}^{N} \left\|\mathbf{p}_{i+\Delta t} - \mathbf{p}_i - \mathbf{R}_i \mathbf{R}_{\text{GT}}^T (\mathbf{p}_{i+\Delta t} - \mathbf{p}_i)\right\|_2$.

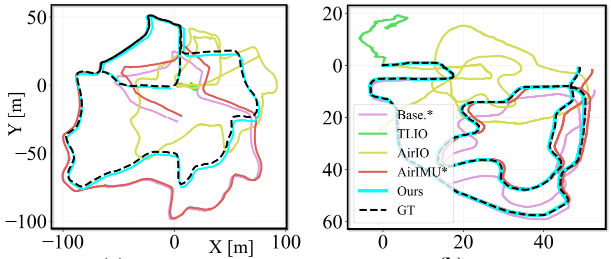
Here, for APE, $\mathbf{p}_i^{\text{est}}$ and $\mathbf{p}_i^{\text{gt}}$ represent estimated and ground truth positions. For RPE, $\mathbf{p}_i$ and $\mathbf{R}_i$ represent the estimated position and rotation at the interval start t_i, $\mathbf{R}_{\text{GT}}$ denotes the ground truth rotation, and N is the number of intervals. Both metrics report errors in meters. In the result tables, cells were color-coded to indicate performance rankings: **1st**, 2nd, 3rd.

3) Comparative Methods: We compared our approach against four methods: a Baseline (i.e., dead-reckoning IMU preintegration) and three state-of-the-art learning inertial odometry methods: TLIO [4], AirIMU [8], and AirIO [9]. For fair comparison, we augmented methods without EKF-based frameworks (e.g., AirIMU and Baseline) by adding our LiDAR-based PGO module to address their lack of global drift correction, denoting these augmented variants with an asterisk (*).

To evaluate our motion-aware training strategy's robustness, we deliberately created challenging conditions by limiting training to a single sequence with only 30 epochs, which could potentially lead to overfitting. These challenging conditions were designed to create potential overfitting scenarios to test our approach. Under these constrained settings, all learning-based methods were trained equally, except TLIO, which required 100 epochs to achieve meaningful results for comparison. All

TABLE I: Quantitative comparison on Botanic Garden and DiTer++ datasets with varying training data percentages. Training is performed only on seen sequences (👁: 1005-01, Forest), while evaluation includes both seen and unseen sequences (⊘: 1006-01, 1008-03, LAWN, PARK). Results show that our approach consistently achieves stable performance across different data scales and generalizes better to unseen environments. See Table II footer for details of Ours‡, Ours†, and Ours.

	Dataset	Botanic Garden (★★)						DiTer++ (★★★)					
Train	Seq.	1005-01(👁)		1006-01(⊘)		1008-03(⊘)		Forest(👁)		LAWN(⊘)		PARK(⊘)	
	Eval.	RPE [m]	APE [m]	RPE [m]	APE [m]	RPE [m]	APE [m]	RPE [m]	APE [m]	RPE [m]	APE [m]	RPE [m]	APE [m]
	Baseline*	0.720	15.907	1.180	30.454	0.848	34.450	1.095	6.102	0.817	9.049	1.160	76.380
100 %	TLIO	**0.372**	21.164	2.590	66.180	2.956	60.767	**0.263**	33.731	0.700	34.900	**0.752**	N/A
	AirIO	0.685	23.357	1.188	44.769	**0.798**	19.443	0.916	12.685	0.652	14.811	1.205	70.026
	AirIMU*	0.685	15.884	1.188	41.243	**0.798**	41.619	0.916	8.099	0.652	5.014	1.205	32.982
	Ours‡	0.712	4.991	**1.180**	40.306	0.839	25.807	0.880	1.805	0.644	1.328	1.174	85.252
	Ours†	0.716	4.277	**1.180**	37.618	0.839	25.448	0.836	1.893	**0.638**	1.204	1.159	82.396
	Ours	0.716	**2.531**	**1.180**	**2.495**	0.839	**7.988**	0.836	**1.114**	**0.638**	**0.965**	1.159	**9.943**
60 %	TLIO	2.507	73.939	2.510	60.374	2.911	67.076	**0.302**	9.112	0.681	52.309	**0.693**	N/A
	AirIO	**0.661**	6.531	1.179	12.450	**0.796**	44.696	0.922	33.523	0.683	29.224	1.208	116.882
	AirIMU*	**0.661**	14.369	1.179	29.044	**0.796**	43.023	0.922	13.047	0.683	6.478	1.208	31.905
	Ours‡	0.707	5.164	1.178	41.046	0.836	26.080	0.994	**1.024**	0.725	2.408	1.156	59.049
	Ours†	0.702	5.542	**1.177**	42.815	0.835	26.291	0.861	1.716	**0.632**	1.074	1.134	80.294
	Ours	0.702	**2.523**	**1.177**	**2.712**	0.835	**8.379**	0.861	1.052	**0.632**	**0.464**	1.134	**6.356**
20 %	TLIO	2.480	68.257	2.510	58.767	2.917	71.386	**0.635**	29.428	**0.625**	45.947	**0.718**	N/A
	AirIO	0.915	25.834	1.228	45.406	1.024	41.396	1.043	16.317	0.735	26.045	1.347	90.908
	AirIMU*	0.915	13.746	1.228	28.960	1.024	34.841	1.043	10.775	0.735	3.526	1.347	28.269
	Ours‡	0.712	5.326	1.180	42.582	0.860	25.944	0.997	**0.900**	0.726	2.033	1.162	64.372
	Ours†	**0.695**	5.202	**1.178**	42.688	**0.828**	26.257	0.945	1.235	0.680	2.074	1.139	69.714
	Ours	**0.695**	**2.516**	**1.178**	**2.716**	**0.828**	**8.269**	0.945	1.059	0.680	**0.493**	1.139	**1.387**



(a) 1006-01 **(b)** LAWN

Fig. 6: Trajectory comparison on unseen sequences (a) 1006-01 and (b) LAWN showing our method versus comparative approaches.

methods employed their publicly available implementations with recommended hyperparameters. We evaluated three variants of our method (i.e., Ours‡, Ours†, and Ours; see Table II footer for details).

Note that AirIO / AirIMU and Ours† / Ours share identical RPE values within each pair, as they use the same IMU measurement correction networks but differ only in post-processing (i.e., EKF filtering) and inference (i.e., adaptive weighting).

B. Performance on Environmental Challenges

Table I shows quantitative comparison results in Botanic Garden dataset and training data percentages. Most existing methods, including our method, sometimes maintain lower robustness than the enhanced baseline (Baseline*) on unseen sequences, revealing their tendency to overfit to specific motion patterns and environmental conditions. Among them, our approach maintains comparatively robust performance across environments through motion-aware training. In particular, our approach with adaptive weighting (i.e., Ours) achieves *strong* results with only 20 % training data while demonstrating consistent generalization to unseen sequences. TLIO exemplifies the overfitting problem, showing robust performance on seen sequences with full training data but sharp degradation on unseen sequences as training data decreases.

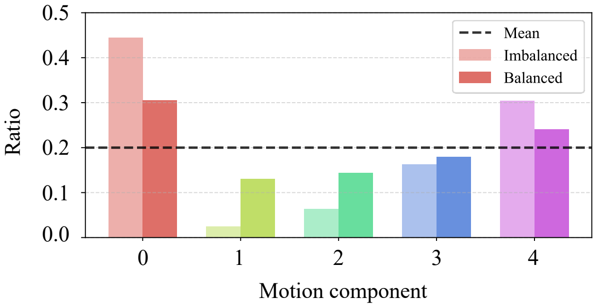


Fig. 7: Motion component weighting before (imbalanced) and after (balanced) GMM balancing on Forest's 20 % training sequence, demonstrating transition from biased to uniform motion coverage.

C. Performance on Motion and Environmental Complexity

Table I shows that both Ours and its variants consistently maintain robust performance, ranking between **1st** and 3rd across almost all metrics. Notably, the improvement from Ours‡ to Ours† demonstrates consistent RPE reduction across all sequences, validating the *stable* performance of GMM-based motion analysis. As shown in Fig. 7, our approach maintains consistent performance using only 20% training data, showing that motion diversity is more important than data volume for generalization. TLIO shows divergent behavior on PARK due to motion distribution differences. Fig. 5 shows Forest has a Wasserstein distance of $d_{\text{FtoL}} = 0.358$ with LAWN, while PARK shows $d_{\text{FtoP}} = 0.604$, indicating different motion distributions that lead to performance degradation across most methods.

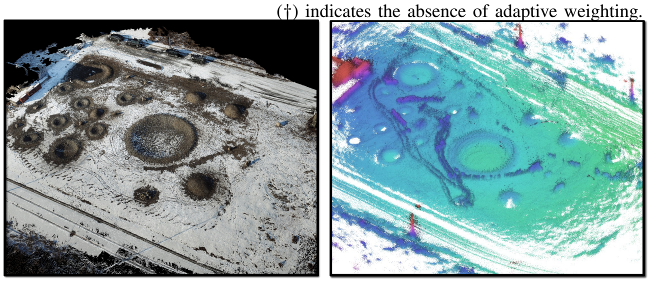
D. Performance on Extreme Conditions

To demonstrate the practical advantages of our self-supervised approach, we conducted a feasibility analysis on our In-House dataset, which represents the most challenging conditions. As shown in Table II, the complex terrain renders traditional supervised methods infeasible due to the inability to acquire ground truth data. Motion capture systems cannot be

TABLE II: Feasibility analysis on `In-House` dataset. The highly complex motions and featureless environment prevent the acquisition of ground truth, making supervised methods (✗) infeasible, while our self-supervised approach (✓) remains deployable.

Dataset	`In-House` (★★★★★)					
Method	TLIO	AirIO	AirIMU*	Ours†	Ours‡	Ours
Feasibility	✗	✗	✗	✓	✓	✓

(‡) indicates the absence of adaptive weighting and GMM-based motion analysis.
(†) indicates the absence of adaptive weighting.



(a) Drone View **(b)** Ours

Fig. 8: Mapping results comparison on `In-House` dataset. (a) Aerial view of crater-filled planetary terrain with Unitree B2 quadruped robot. (b) Our mapping results. The proposed method maintains consistent mapping quality despite challenging environmental and motion conditions.

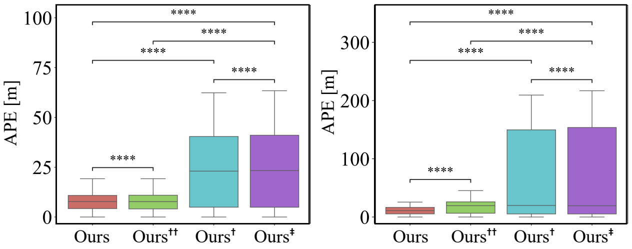
deployed in such outdoor environments, and the highly dynamic quadruped motions prevent maintaining the centimeter-level accuracy required for supervised training. In contrast, our *self-supervised* approach remains fully deployable, requiring only inertial and point cloud data without external ground truth. As shown in Fig. 8, our method maintains consistent mapping even under these extreme conditions. This highlights the practical value of self-supervised learning for real-world applications where supervised methods face deployment constraints.

E. Ablation studies

Finally, we conducted ablation studies across diverse robotic platforms and motion patterns to evaluate the contribution of each component within our framework. Fig. 9 demonstrates statistically significant performance improvements across all framework components (GMM balancing and adaptive weighting). This validates the generalizability and effectiveness of our integrated approach through comprehensive statistical analysis. Notably, Ours†† (without GMM but with adaptive weighting) shows the 2nd performance benefits of adaptive weighting alone, with detailed trajectory comparisons illustrated in Fig. 1.

V. Conclusion & Discussion

We propose a novel self-supervised inertial odometry (IO) framework, called *KISS-IMU*, that eliminates ground truth dependency through selective LiDAR registration or PGO and GMM-based motion analysis. Our approach achieves *stable* IO through balanced motion learning and *strong* IO through adaptive weighting based on uncertainty-aware inference. Experimental results demonstrate that our method maintains robust performance even as complexity increases, ranging from environmental challenges in natural settings to highly dynamic quadruped robot motions in challenging terrains Notably, our framework demonstrates extensibility by successfully integrating with other IO methods, showing its potential for broader applicability beyond the tested implementations. To the best of our knowledge, this is the first framework that learns only IO in a self-supervised manner, distinguishing it from existing approaches that require either ground truth supervision or learnable supervision. Our motion analysis techniques open promising directions for future work, particularly in online learning and test-time adaptation.



(a) `1008-03` **(b)** `PARK`

Fig. 9: Absolute pose error (APE) box plots on unseen sequences: (a) `1008-03` and (b) `PARK`. Statistical significance ($****$: p-value $< 10^{-4}$ after a paired t-tests) confirms our method's consistent performance improvements.

References

[1] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," *Proc. Robot.: Science & Sys. Conf.*, 2015.

[2] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," in *Proc. AAAI National Conf. on Art. Intell.*, vol. 32, no. 1, 2018.

[3] S. Herath, H. Yan, and Y. Furukawa, "Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods," in *Proc. IEEE Intl. Conf. on Robot. and Automat.* IEEE, 2020, pp. 3146–3152.

[4] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis *et al.*, "Tlio: Tight learned inertial odometry," *IEEE Robot. and Automat. Lett.*, vol. 5, no. 4, pp. 5653–5660, 2020.

[5] Y. Wang, J. Kuang, X. Niu, and J. Liu, "Llio: Lightweight learned inertial odometer," *IEEE Internet of Things Journal*, vol. 10, no. 3, pp. 2508–2518, 2022.

[6] M. Brossard, A. Barrau, and S. Bonnabel, "Ai-imu dead-reckoning," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.

[7] G. Cioffi, L. Bauersfeld, E. Kaufmann, and D. Scaramuzza, "Learned inertial odometry for autonomous drone racing," *IEEE Robot. and Automat. Lett.*, vol. 8, no. 5, pp. 2684–2691, 2023.

[8] Y. Qiu, C. Wang, C. Xu, Y. Chen, X. Zhou *et al.*, "Airimu: Learning uncertainty propagation for inertial odometry," *arXiv preprint arXiv:2310.04874*, 2023.

[9] Y. Qiu, C. Xu, Y. Chen, S. Zhao, J. Geng *et al.*, "Airio: Learning inertial odometry with enhanced imu feature observability," *arXiv preprint arXiv:2501.15659*, 2025.

[10] T. Fu, S. Su, Y. Lu, and C. Wang, "islam: Imperative slam," *IEEE Robot. and Automat. Lett.*, vol. 9, no. 5, pp. 4607–4614, 2024.

[11] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon, "Learning inertial odometry for dynamic legged robot state estimation," in *Conference on robot learning*. PMLR, 2022, pp. 1575–1584.

[12] S. Zhao, S. Zhou, R. Blanchard, Y. Qiu, W. Wang *et al.*, "Tartan imu: A light foundation model for inertial positioning in robotics," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 22 520–22 529.

[13] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[14] D. He, W. Xu, N. Chen, F. Kong, C. Yuan *et al.*, "Point-lio: robust high-bandwidth light detection and ranging inertial odometry," *Advanced Intelligent Systems*, vol. 5, no. 7, p. 2200459, 2023.

[15] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley *et al.*, "Kiss-icp: In defense of point-to-point icp–simple, accurate, and robust registration if done the right way," *IEEE Robot. and Automat. Lett.*, vol. 8, no. 2, pp. 1029–1036, 2023.

[16] K. Koide, "small_gicp: Efficient and parallel algorithms for point cloud registration," *Journal of Open Source Software*, vol. 9, no. 100, p. 6948, Aug. 2024.

[17] C. Wang, D. Gao, K. Xu, J. Geng, Y. Hu *et al.*, "Pypose: A library for robot learning with physics-based optimization," in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2023, pp. 22 024–22 034.

[18] M. Jung, S. Jung, H. Gil, and A. Kim, "Helios: Heterogeneous lidar place recognition via overlap-based learning and local spherical transformer," *arXiv preprint arXiv:2501.18943*, 2025.

[19] R. L. Russell and C. Reale, "Multivariate uncertainty in deep learning," *IEEE Trans. Neural Networks and Learning Sys.*, vol. 33, no. 12, pp. 7937–7943, 2021.

[20] H. Wan, H. Wang, B. Scotney, and J. Liu, "A novel gaussian mixture model

for classification," in *2019 IEEE international conference on systems, man and cybernetics (SMC)*. IEEE, 2019, pp. 3298–3303.

[21] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2019, pp. 9268–9277.

[22] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[23] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.

[24] Y. Liu, Y. Fu, M. Qin, Y. Xu, B. Xu *et al.*, "Botanicgarden: A high-quality dataset for robot navigation in unstructured natural environments," *IEEE Robot. and Automat. Lett.*, vol. 9, no. 3, pp. 2798–2805, 2024.

[25] J. Kim, H. Kim, S. Jeong, Y. Shin, and Y. Cho, "Diter++: Diverse terrain and multi-modal dataset for multi-robot slam in multi-session environments," in *Proc. IEEE Intl. Conf. on Robot. and Automat.* IEEE, 2025, pp. 12 187–12 193.