GNSS/Multisensor Fusion Using Continuous-Time Factor Graph Optimization for Robust Localization

Zhang, Haoming; Chen, Chih Chun; Vallery, Heike; Barfoot, Timothy D.

DOI
10.1109/TRO.2024.3443699

Publication date
2024

Document Version
Final published version

Published in
IEEE Transactions on Robotics

Citation (APA)
Zhang, H., Chen, C. C., Vallery, H., & Barfoot, T. D. (2024). GNSS/Multisensor Fusion Using Continuous-Time Factor Graph Optimization for Robust Localization. *IEEE Transactions on Robotics*, *40*, 4003-4023. https://doi.org/10.1109/TRO.2024.3443699

GNSS/Multisensor Fusion Using Continuous-Time Factor Graph Optimization for Robust Localization

Haoming Zhang, *Member, IEEE*, Chih-Chun Chen, *Graduate Student Member, IEEE*, Heike Vallery, *Member, IEEE*, and Timothy D. Barfoot, *Fellow, IEEE*

***Abstract*—Accurate and robust vehicle localization in highly urbanized areas is challenging. Sensors are often corrupted in those complicated and large-scale environments. This article introduces `gnssFGO`, a global and online trajectory estimator that fuses global navigation satellite systems (GNSS) observations alongside multiple sensor measurements for robust vehicle localization. In `gnssFGO`, we fuse asynchronous sensor measurements into the graph with a continuous-time trajectory representation using Gaussian process (GP) regression. This enables querying states at arbitrary timestamps without strict state and measurement synchronization. Thus, the proposed method presents a generalized factor graph for multisensor fusion. To evaluate and study different GNSS fusion strategies, we fuse GNSS measurements in loose and tight coupling with a speed sensor, inertial measurement unit, and LiDAR-odometry. We employed datasets from measurement campaigns in Aachen, Düsseldorf, and Cologne and presented comprehensive discussions on sensor observations, smoother types, and hyperparameter tuning. Our results show that the proposed approach enables robust trajectory estimation in dense urban areas where a classic multisensor fusion method fails due to sensor degradation. In a test sequence containing a 17-km route through Aachen, the proposed method results in a mean 2-D positioning error 0.48 m while fusing raw GNSS observations with LiDAR odometry in a tight coupling**

***Index Terms*—Autonomous vehicle navigation, factor graph optimization (FGO), global navigation satellite systems (GNSS), localization, sensor fusion.**

Manuscript received 19 February 2024; revised 9 July 2024; accepted 31 July 2024. Date of publication 15 August 2024; date of current version 5 September 2024. This work was supported by AFR-Aachen Research Association for Control Engineering e.V. This article was recommended for publication by Associate Editor S. Weiss and Editor J. Civera upon evaluation of the reviewers' comments. *(Corresponding author: Haoming Zhang.)*

Haoming Zhang and Chih-Chun Chen are with the Institute of Automatic Control, Faculty of Mechanical Engineering, RWTH Aachen University, 52062 Aachen, Germany (e-mail: haoming.zhang@rwth-aachen.de; c.chen@irt.rwth-aachen.de).

Heike Vallery is with the Institute of Automatic Control, Faculty of Mechanical Engineering, RWTH Aachen University, 52062 Aachen, Germany, and with the Department of BioMechanical Engineering, Delft University of Technology, 2628 CD Delft, Netherlands, and also with the Department for Rehabilitation Medicine, Erasmus MC, 3015, GD Rotterdam, Netherlands (e-mail: h.vallery@irt.rwth-aachen.de).

Timothy D. Barfoot is with the University of Toronto Robotics Institute, Toronto, ON M5S 0C9, Canada (e-mail: tim.barfoot@utoronto.ca).

This article has supplementary downloadable multimedia material available at https://ieeexplore.ieee.org provided by the authors. This includes a video. The uploaded multimedia presents brief introduction of the problem statement and method in this manuscript. It also shows supplementary experiment results for: 1) robust and accurate vehicle localization in challenging areas by benchmarking with a LiDAR-centric approach (LIO-SAM); 2) vehicle localization in high-speed scenario; 3) robust vehicle localizaton and consistent LiDAR-map building in a large-scale environment. High-resolution available at: https://youtu.be/JhxJc1NFN7g Full figures in the manuscript: https://github.com/rwth-irt/gnssFGO/tree/ros2/online_fgo/plots_tro

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2024.3443699, provided by the authors.

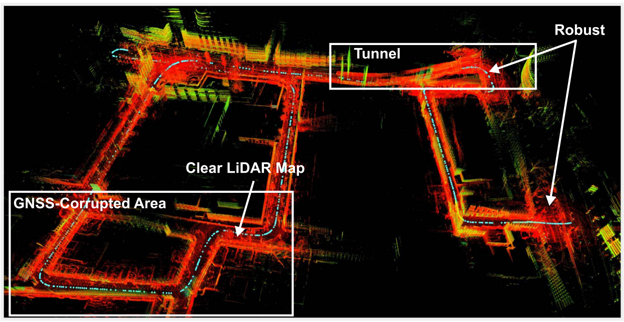
Digital Object Identifier 10.1109/TRO.2024.3443699

I. Introduction

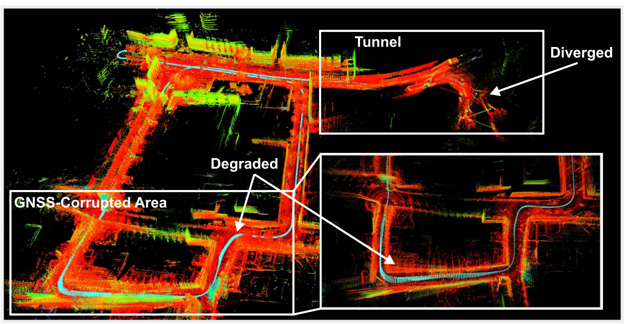
SAFE and reliable autonomous driving operations in urban areas require accurate and consistent vehicle localization that infers a smooth trajectory estimate for planning and control tasks. Autonomous vehicles may use global navigation satellite systems (GNSS) to achieve global positioning in large-scale environments. However, the performance of GNSS is highly degraded when a vehicle passes through tunnels or urban canyons, where GNSS signal loss can be expected, greatly penalizing positioning availability. Moreover, the error dynamics of GNSS observations grow increasingly complex due to multipath and nonline-of-sight effects, resulting in inconsistent error models used in state estimation [1].

Many previous works fuse information from local optical sensors (e.g., LiDARs or cameras) for vehicle localization. They can typically be categorized into pose retrieval using a given map [3] and simultaneous location and mapping (SLAM) [4]. Generally, landmarks in sensor frames are extracted and associated to acquire either frame-to-map global pose constraints or frame-to-frame local motion increments. Lacking high-quality maps for vehicle pose retrieval in many areas, approaches relying on local sensors can often only achieve satisfactory localization if the ground is even and sufficient loop-closure constraints help eliminate drift. However, these requirements cannot always be met for long-term autonomous operations in large-scale environments [5].

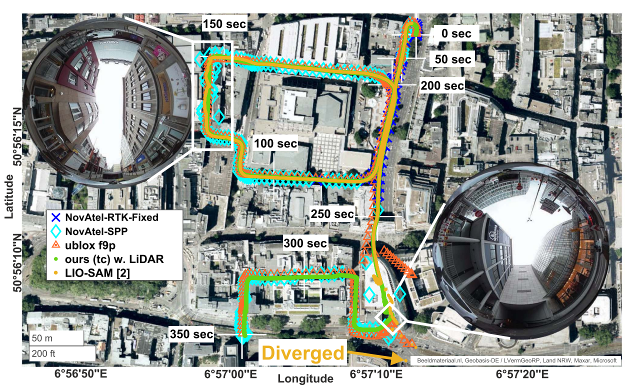
In recent years, combining local sensors with GNSS has been investigated as a robust way to enable accurate and precise vehicle location in challenging areas. Incremental batch estimation implemented as factor graph optimization (FGO) is often superior to classic filtering-based algorithms in terms of localization performance and consistency [6], [7]. Unlike Bayesian filters, a factor graph fuses prior information and sensor measurements associated with the to-be-estimated state variables into probabilistic representations. A maximum-a-posterior problem (MAP) can be formulated from the factor graph and solved in a batch configuration using iterative Gauss–Newton-like algorithms [8]. In general, this optimization procedure is activated only if new sensor observations are available. Thus, previous works that use FGO generally rely on a primary sensor that schedules the optimization procedure [2], [9], [10], [11], [12].



(a)



(b)



(c)

Fig. 1. Demonstration of multisensor fusion for vehicle localization of test sequence C02 in the city of Cologne. (a) LiDAR map and trajectory using tightly coupled `gnssFGO` (ours). The proposed method provides robust trajectory estimation and clear LiDAR maps in GNSS-corrupted areas. (b) LiDAR map and trajectory using LIO-SAM [2]. This approach fused with GNSS positioning failed due to faulty scan registrations while crossing a tunnel and degrades dramatically with corrupted GNSS measurements. (c) Measured and estimated trajectories. Fisheye images show challenging areas where GNSS observations are blocked or strongly corrupted.

By this means, the primary sensor that is expected to effectively constrain the vehicle states coordinates the creation of new to-be-estimated state variables and initiates the solver for the MAP problem. To fuse additional sensor modalities, asynchronous measurements must be synchronized with the primary sensor, leading to information loss and inefficient fusion mechanisms. Furthermore, classic FGO approaches degrade if the primary sensor is compromised or fails, which is likely in challenging environments. In this case, state variables cannot be effectively constrained by other sensor observations if the graph is not constructed in time. Fig. 1 exemplifies this problem, where a state-of-the-art LiDAR-centric SLAM approach diverges due to scan registration failures while driving in a tunnel.[1] In fact, as discussed in [13], [14], and [15], commonly used sensors in localization deteriorate under challenging environmental conditions, complicating robust and long-term vehicle localization.

In this work, we address the degradation problem of GNSS-based localization approaches by translating classic FGO for multisensor fusion into an approach where the graph associated with all to-be-estimated state variables is constructed deterministically based on a priori chosen timestamps. It thus presents a time-centric factor graph construction that is independent of any particular reference sensor (e.g., GNSS). To achieve this, we represent the vehicle trajectory in continuous time using a GP. This approach incorporates a motion prior using the white-noise-on-jerk (WNOJ) motion model, as originally proposed in [16]. The algorithm feeds new observations from each sensor independently into the factor graph without measurement-to-state synchronization. If a measurement cannot be temporally aligned with any state variable, we query a GP-interpolated state corresponding to the measurement used for the error evaluation.

To retrieve a robust global trajectory estimation while the GNSS measurements are strongly corrupted, we implemented the time-centric factor graph to fuse GNSS observations with measurements of an inertial measurement unit (IMU), optical speed sensor, and LiDAR for vehicle localization in challenging urban scenarios. We propose two factor graph structures for both loosely and tightly coupled fusion of GNSS observations alongside other local sensor measurements, demonstrating the flexibility of the proposed `gnssFGO`. For the graph that considers the GNSS positioning solution of a low-grade GNSS receiver in the loose coupling, we fuse the preintegrated IMU measurements, 2-D velocity measurements, and LiDAR odometry. In tightly coupled fusion, we replace GNSS solution factors with GNSS pseudorange and Doppler-shift factors, which are expected to provide more effective constraints compared to inconsistent GNSS positioning in urban areas [6].

We used raw data from measurement campaigns in the cities of Aachen, Düsseldorf, and Cologne to evaluate the proposed approach by benchmarking with a well-known LiDAR-centric SLAM approach [2], [17]. This LiDAR-centric SLAM has been shown to perform best for vehicle localization tasks in large-scale environments and can be configured to fuse GNSS measurements [18], which presents an equivalent fusion mechanism as our loosely coupled `gnssFGO`.

In contrast to our previous study [7], which focused only on trajectory smoothness using an offline FGO, we now address online multisensor fusion for vehicle localization.

The contributions of this work are summarized as follows.

1) We propose a flexible, online, continuous-time FGO framework that can accommodate common multisensor fusion problems. The flexibility comes from the fact that

[1] Same noise models and smoother were used while benchmarking the LiDAR-centric approach with tightly coupled `gnssFGO`. The same illustration settings (e.g., point size) were used in Fig. 1(a) and (b).

a) we can accommodate asynchronous measurements and b) we choose estimation timestamps independent of any particular sensor frequency. This latter feature, as well as the smoothing effect of a motion prior, provides robustness in the presence of any particular sensor dropout.

2) We implement the proposed method for vehicle localization in challenging scenarios and conduct comprehensive studies on loosely coupled and tightly coupled fusion mechanisms to fuse GNSS measurements with other local sensors to present extensive evaluations and discussions on accuracy, robustness, and run-time efficiency.
3) We evaluate the GP motion prior, which is implemented using the white-noise-on-acceleration (WNOA) and WNOJ models, to study the accuracy of the interpolated states.
4) We introduce a scalable and modular estimation framework `gnssFGO`[2] that can be extended for arbitrary robot localization using continuous-time FGO.

The rest of this article is organized as follows. Section II presents a comprehensive literature review on multisensor fusion. Section III introduces the proposed continuous-time FGO in detail. The mathematical background for factor formulations is presented in Section IV, whereas the graph implementations are introduced in Section V. We verify our method in Section VII and conduct further experiments on the precision and consistency of estimated trajectories. Finally, Section VIII concludes this article. We release our code and raw data in our experiments. A demonstration video is also available.[3]

II. Related Work

A. Graph Optimization for GNSS-Based Vehicle Localization

In recent years, fusing GNSS observations using FGO for robust vehicle localization has drawn great attention. Compared with filtering-based approaches, FGO conducts batch optimization, where all measurement models are relinearized and reevaluated iteratively, resulting in a more robust state estimation even with measurement outliers. Previous work demonstrated robust localization in urban areas only by factoring pseudoranges with robust error models [19], [20]. Later, Wen et al., [6] and Zhang et al., [7] showed that FGO generally outperforms Kalman filters with respect to the precision and smoothness of the estimated trajectory.

GNSS data can be integrated into the graph using a loosely or tightly coupled schema [21]. While the loosely coupled fusion incorporates GNSS positioning solution into the graph, preprocessed raw GNSS observations, such as code or carrier-phase measurements can be fed into the estimator in a tight coupling as state constraints. As the to-be-estimated state variables can be directly observed in GNSS solutions, fusing GNSS data in a loose coupling enables quick convergence and elevated accuracy if high-quality real-time-kinematic (RTK)-fixed GNSS solutions are available.

[2][Online]. Available: https://github.com/rwth-irt/gnssFGO
[3][Online]. Available: https://youtu.be/JhxJc1NFN7g

In contrast, the integration of raw GNSS observations contributes to multiple state constraints associated with received satellites, which has been shown to be more robust than loose coupling [6], [22], [23]. Wen et al. [24] included double-differenced pseudorange (DDPR) and double-differenced carrier-phase measurements (DDCP) in FGO, resulting in performance improvement. Later, this work was extended to efficiently model carrier-phase constraints between multiple satellite measurement epochs within a time window [25]. In [26], time-differenced carrier-phase (TDCP) was integrated with the cycle-slip estimation, which achieved accurate localization while presenting substantial availability compared to DDCP if satellites can be continuously tracked. Congram and Barfoot [27] also proposed a global positioning system (GPS) odometry using TDCP with more prominent cycle slip detection and showed an effective drift reduction compared to visual odometry. However, since carrier-phase observations are also disturbed in deep urban areas, the robustness of the state estimation cannot yet be guaranteed.

As FGO presents a convenient tool for robust error modeling [28], several works employ m-estimators to reject faulty GNSS observations [7], [20], [26], [29]. Recently, FGO has been explored in the context of vehicle location based on GNSS for noise distribution identification or adaptive rejection of outliers [30], [31], showing a positive impact on consistent trajectory estimation using FGO.

B. Graph Optimization for Multisensor Fusion

While the aforementioned works have particularly explored graph optimization for GNSS observations, they may still suffer from performance degeneration in complex scenarios if GNSS measurements are lost or present outliers. Therefore, another research domain focuses on fusing more sensor modalities (more than two) alongside GNSS observations into the graph, with applications predominantly in SLAM.

A pose graph that fuses GPS position measurements and LiDAR odometry with loop-closure constraints for outdoor scenarios improved both runtime efficiency and performance compared to LiDAR-only approaches [9]. In [2], feature-based LiDAR odometry and loop-closure constraints were merged into a factor graph with synchronized GPS position measurements to achieve a drift-free pose estimate, which was forwarded to another graph optimization with preintegrated IMU measurements for high-frequency and real-time state estimation. In addition to integrating feature-based LiDAR odometry into FGO, the LiDAR map can also be used for GNSS visibility assessment [11]. Some works also introduce camera-centric sensor fusion, where other sensor observations are synchronized with camera data and fused on the graph [10], [12], [32]. In [33], camera, LiDAR, and wheel odometers were fused into the graph along with the GNSS positioning solution and IMU measurements, presenting consistent localization in featureless environments for long-term runs. Similar works also conduct multisensor fusion without GNSS and propose a carefully managed fusion architecture [34], [35]. However, these works still require well-handled data synchronization and careful graph construction to fuse heterogeneous sensor measurements.

Many recent approaches introduce multigraph structures to achieve flexible and compact sensor fusion. In [36], IMU, GNSS, and LiDAR observations were separately integrated into multiple graphs in parallel with a switching mechanism. When the GNSS receiver lost its signal, the LiDAR-centric graph was activated. Another work aimed to confederate loosely and tightly coupled fusion schemes to ensure estimation performance [37]. Each sensor modality is associated with a separate graph and proposes odometry factors to the IMU-centric graph that provides final estimated states in real time. However, incorporating multigraph structures introduces redundant and complex system architectures that may require well-managed engineering work. Moreover, these works did not address challenging environments where sensor observations can be highly corrupted with inconsistent noise distributions.

Other works exploit high-frequency IMU measurements to coordinate multisensor fusion. In [38] and [39], asynchronous global pose measurements (e.g., GNSS measurements) are propagated into timestamps of visual-inertial factors using preintegrated IMU measurements. The same concept has been extended to a forward–backward IMU preintegration mechanism in order to precisely associate asynchronous measurements with keyframes [40]. Nevertheless, these methods still depend on the noisy IMU sensor, which introduces uncertainty.

C. Continuous-Time Trajectory Representation

One essential requirement for flexible graph-based multisensor fusion is the ability to query the states associated with the observations within the iterative optimization process. This requirement can be fulfilled if the trajectory is represented in continuous time. In [41], B-splines were proposed as a parametric approach to represent the trajectory in continuous time. This method was later used to propose stereo-inertial odometry [42]. Another approach utilizes exactly sparse GP regression by assuming that system dynamics follows a linear time-varying stochastic differential equation (LTV-SDE) [43]. The system dynamics is typically modeled as WNOA. This approach was verified in [44], [45], and[46], where the reliability of this proposed surrogate dynamics model was demonstrated. Recently, Tang et al. [16] proposed an improved system dynamics model, which assumed a WNOJ model in LTV-SDE. They showed that the WNOJ could model the vehicle dynamics more accurately, and thus, was appropriate for systems with more complicated dynamics. As discussed in [47], continuous-time trajectory using GPs should be used if the measurement times match the estimation times. Thus, we follow this aspect and adapt the GP-WNOJ model proposed in [16] as between-state motion constraints and state interpolator to fuse asynchronous measurements. Although continuous-time trajectory representation is studied for localization and mapping problems by extending incremental smoothing using sparse GP interpolation to reduce computation time [45], fusing GNSS observations with multiple heterogeneous sensor measurements for online vehicle localization has not yet been presented or discussed.

D. Modular Estimation Framework

As the aforementioned approaches share similar procedures to solve estimation problems, the idea of a modular estimation framework that unifies the system design for different applications has emerged. Labbé and Michaud [48] originally proposed a real-time mapping framework to manage the memory of the internal map for loop closure detection. This framework was continuously extended by the same authors in [49] to enable multisensor fusion and benchmarking. Due to its modularity and scalability, many works based on this framework can be performed [50], [51]. A similar framework for a plug-and-play SLAM system was presented in [52]. Recently, Solà et al. [53] inherited this design with a tree-based estimation framework, which formulates all the necessary robot entities in different branches, including hardware, trajectory, and map management. Sensor measurements and prior information are fused using a decentralized strategy in which primary sensors are selected in the configuration file to actively create new keyframes (aka state variables). However, none of the modular frameworks mentioned above represents the trajectory in continuous time, which still requires measurement alignment to keyframes, making a loss of sensor observations inescapable.

Inspired by abovementioned works, we address the problem of multisensor fusion for GNSS-based vehicle localization using continuous-time trajectory representation, which enables a fusion of asynchronous sensor observations in a single factor graph. Our hypotheses of the contributions presented above are: 1) factor graph construction in continuous time generalizes multisensor fusion and enables consistent trajectory estimation that incorporates effective state constraints from multiple sensor modalities in challenging scenarios; 2) in this spirit, a natural and efficient modular estimation framework can be presented thanks to the ability of querying state at arbitrary times, in which different applications and experiments can be configured directly using configuration files; and 3) the GP-WNOJ motion model presents a larger capacity to represent complicated system dynamics, such as driving in urban areas.

III. Time-Centric Factor Graph Optimization

In this section, we introduce an implementation of continuous-time trajectory estimation, as proposed in [16] and [43]. Generally, fusing multiple heterogeneous sensor observations into a state estimator incorporates different timestamps due to asynchronous measurements and unpredictable delays. In this work, we assume that the state estimator and all measurements have the same clock. Compared to the estimated states in continuous time, all sensor observations are sampled and processed in asynchronous timestamps, as illustrated in Fig. 2. Here, we use the variable $\tau = t_{\text{meas.}} - t_{\boldsymbol{x}_i}$ to define the time offset between a nondelayed measurement and the last state variable $\boldsymbol{x}_i$ prior to this measurement. If a measurement is delayed with the timestamp $\check{t}_{\text{meas.}}$, we use the given time delay t_d to calculate the nondelayed measurement timestamp $t_{\text{meas.}} = \check{t}_{\text{meas.}} - t_d$.

We employ GP motion priors that enable a continuous-time trajectory representation. In this way, constructing a factor graph can be deterministic and time-centric, bypassing asynchronous

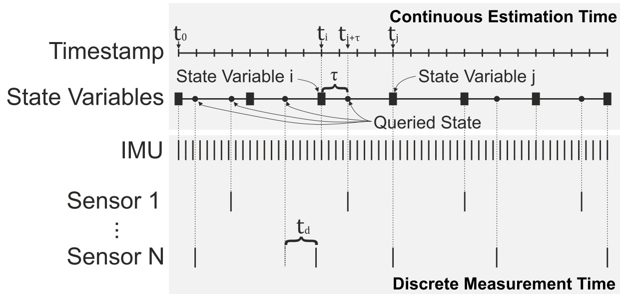


Fig. 2. Continuous-time state estimation with asynchronous measurements. A time offset τ can be calculated with respect to a former state variable $\boldsymbol{x}_i$ at timestamp t_i for each asynchronous measurement. The variable t_d denotes a measurement delay that is assumed to be given.

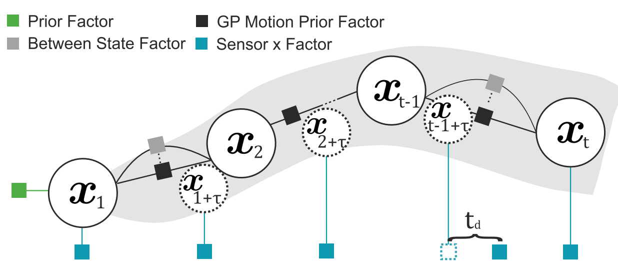


Fig. 3. General time-centric factor graph. The state variables $\boldsymbol{x}_t$ are created and constrained with GP motion prior factors on time while all asynchronous measurements are fused by querying a state with a time offset τ between the measurement and the former state variable. The queried states (in dashed circles) are thus not to-be-estimated state variables. We assume that the measurement delay t_d is known to correct the measurement timestamp for querying a state.

sensor frequencies and timing issues. We show the general structure of a time-centric factor graph in Fig. 3, where the to-be-estimated state variables $\boldsymbol{x}_t$ are presented in solid line circles on a continuous-time trajectory. Queried states in dashed line circles are not to-be-estimated state variables and, therefore, are only queried between two successive state variables using the time offset τ between the sensor timestamp and the previous state variable.

Algorithm 1 explains one optimization procedure from graph construction to iterative optimization. Assume that the time-centric factor graph is extended with n new to-be-estimated state variables in each procedure. We extend the graph with n state variables and create GP motion prior factors that constrain the relative state transitions between two successive state variables. In doing so, the timestamps of all state variables are chosen deterministically. While solving the iterative optimization problem, an initial prediction $\boldsymbol{x}_k^- \in \boldsymbol{\mathcal{X}}^-$ must be provided for each state variable. These predictions can be acquired using prior motion models (e.g., GP state extrapolation [54]). In this work, we utilize state propagation using IMU measurements to calculate the initial estimate of future states at high frequency.

As new sensor observations are received at different timestamps in parallel to estimation times, we retrieve the cached m observations from each sensor $s \in \mathcal{S}$ in a second loop. We define a time threshold t_{sync} for state-observation alignment to query the index of related state variables. If state variables can be associated with sensor observations within this threshold, normal sensor factors are added to the graph. Otherwise, we

Algorithm 1: Time-Centric Factor Graph Optimization.

Input : Last state id and timestamp pair $(x_{\text{id}}^-,\ x_{\text{ts}}^-)$
Propagated states $\boldsymbol{x}_k^- \in \boldsymbol{\mathcal{X}}^-,\ k = 1 \ldots n$
List of sensor measurements $s \in \boldsymbol{\mathcal{S}}$
Output: Current state id and timestamp pair $(x_{\text{id}}^+,\ x_{\text{ts}}^+)$
Optimized state $\boldsymbol{x}_k^+$ and uncertainties $\boldsymbol{P}_k^+$

$\boldsymbol{\mathcal{G}} \leftarrow$ `initGraph`$(\boldsymbol{x}_0^-,\ \boldsymbol{P}_0^-)$;
List of state id and timestamp pairs $\boldsymbol{\mathcal{P}} = \emptyset$;
$x_{\text{id}} = x_{\text{id}}^-$;
for $k = 1 : n$ **do**
  $x_{\text{id}}^+ = x_{\text{id}} + 1$;
  $x_{\text{ts}}^+ =$ `updateTimestamp`(x_{ts}^-);
  $\boldsymbol{\mathcal{G}} \leftarrow$ `NewStateVariable`$(x_{\text{id}},\ x_{\text{ts}},\ \boldsymbol{\mathcal{X}})$;
  $\boldsymbol{\mathcal{G}} \leftarrow$ `GPMotionFactor`$(x_{\text{id}-1},\ x_{\text{id}})$;
  $\boldsymbol{\mathcal{P}} \leftarrow (x_{\text{id}},\ x_{\text{ts}})$;
end for
for *Each sensor* $s \in \boldsymbol{\mathcal{S}}$ **do**
  for *Each observation* $k = 1 : m$ **do**
    $(x_i^{\text{id}},\ x_i^{\text{ts}},\ \tau,\ type)$
      $\leftarrow$ `queryStateInfo`(timestamp$_k^{\boldsymbol{s}}$, $\boldsymbol{\mathcal{P}}$);
    if *type is* `DROPPED` **then**
      /* *Measurements in the past.* */;
      `discardMeasurement`$(\boldsymbol{o}_k^s)$;
    else if *type is* `SYNCHRONIZED` **then**
      $\boldsymbol{\mathcal{G}} \leftarrow$ `SensorFactor`$(x_{\text{id}},\ \boldsymbol{o}_k^s)$;
    else if *type is* `INTERPOLATED` **then**
      $\boldsymbol{\mathcal{G}} \leftarrow$
      `GPSensorFactor`$(x_i^{\text{id}},\ x_{i+1}^{\text{id}},\ \tau,\ \boldsymbol{o}_k^{\boldsymbol{s}})$;
    else if *type is* `CACHED` **then**
      /* *Measurements in the future.* */
      `cacheMeasurement`$(\boldsymbol{o}_k^s)$;
  end for
end for
$\{\boldsymbol{x}_k^+,\ \boldsymbol{P}_k^+\} \leftarrow$
  `doOptimizationAndMarginalization`$(\boldsymbol{\mathcal{G}})$;
return $\{(x_{\text{id}}^+,\ x_{\text{ts}}^+),\ \boldsymbol{x}_k^+,\ \boldsymbol{P}_k^+\}$

construct the measurement factors by querying a GP interpolated state aligned with the measurement timestamp. In this case, two successive state variables $\boldsymbol{x}_i$ and $\boldsymbol{x}_j, j = i + 1$ are obtained with a time offset τ between the measurement and the former state $\boldsymbol{x}_i$.

After graph construction, we employ a Gauss–Newton-like optimizer to solve the MAP problem [55]. The optimized state $\boldsymbol{x}_k^+$ and marginalized uncertainties $\boldsymbol{P}_k^+$ are returned for further state propagation, as introduced in Section V-F.

IV. Mathematical Background

A. Frame and Frame Transformation

Starting with GPS in 1987, GNSS typically provides a vehicle's position in the World Geodetic System (WGS84) frame using geodetic ellipsoidal (aka geodetic) coordinates (latitude φ, longitude λ, and height h, LLH) [56], as shown in Fig. 4. Geodetic coordinates can be transformed into the Earth-centered, Earth-fixed (ECEF) frame that is defined at the center of the

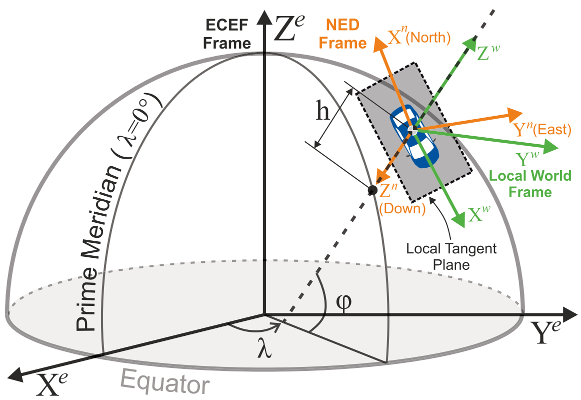


Fig. 4. Coordinate frames used in this work.

Earth's mass, denoted as $(\cdot)^e$. As many estimation and control approaches require a Cartesian frame in a local tangent plane, the North-East-Down (NED) frame and the East-North-Up (ENU) frame are commonly introduced as navigation frames $(\cdot)^n$ to present the vehicle's velocity and orientation. In this work, we present the pose and velocity of the vehicle in the ECEF frame. A transformation from frame e to frame n is used to formulate factors and calculate error metrics. We also introduce a local-world frame $(\cdot)^w$ following [10] that rotates by the initial yaw angle of the vehicle in frame n. The body frame is denoted as $(\cdot)^b$. In the following, we briefly introduce the related frame transformations.

1) Transform Between Geodetic and Cartesian Coordinates: A geodetic coordinate $\boldsymbol{p}^{\mathrm{LLH}} = [\varphi \;\; \lambda \;\; h]^T$ can be transformed to the ECEF frame using [21]

$$x_b^e = (R_e + h)\cos\varphi\cos\lambda \tag{1}$$

$$y_b^e = (R_e + h)\cos\varphi\sin\lambda \tag{2}$$

$$z_b^e = [R_e(1 - \mathrm{ecc}^2) + h]\sin\varphi \tag{3}$$

where the constant $\mathrm{ecc} = 0.08181919$ is the eccentricity[4] of the ellipsoid. The transverse radius of curvature given the latitude φ is calculated as $R_e(\varphi) = \frac{a}{\sqrt{1-\mathrm{ecc}^2\sin^2(\varphi)}}$, where the scalar $a = 6378137\,\mathrm{m}$ denotes the equatorial radius[4] of the Earth.

However, transforming a Cartesian coordinate in frame e back to LLH using the inverse of (1)–(3) can only be solved iteratively due to the nonlinearity [21]. Although many closed-form alternatives for this transformation are available, we use Heikkinen's solution [57] in this work considering its high precision [58]. Due to limited space, see [57] or our code[2] for more details on the implementation.

2) Transform Between Frame e and Frame n: Given a point in the e frame $\boldsymbol{p}_0^e$ as origin, a fixed navigation frame (aka local tangent plane) can be determined by two plane rotations associated with longitude λ_0 and latitude φ_0 of $\boldsymbol{p}_0^{\mathrm{LLH}}$. Fig. 4 illustrates this transformation from the frame e to the NED frame using the direction cosine matrix (DCM)

$$\boldsymbol{R}_e^{\mathrm{ned}}(\boldsymbol{p}_0^{\mathrm{LLH}}) = \begin{bmatrix} -\sin\varphi_0\cos\lambda_0 & -\sin\varphi_0\sin\lambda_0 & \cos\varphi_0 \\ -\sin\lambda_0 & \cos\lambda_0 & 0 \\ -\cos\varphi_0\cos\varphi_0 & -\cos\varphi_0\sin\lambda_0 & -\sin\varphi_0 \end{bmatrix}. \tag{4}$$

The DCM from frame e to the ENU frame is given as

$$\boldsymbol{R}_e^{\mathrm{enu}}(\boldsymbol{p}_0^{\mathrm{LLH}}) = \begin{bmatrix} -\sin\lambda_0 & \cos\lambda_0 & 0 \\ -\cos\lambda_0\sin\varphi_0 & -\sin\lambda_0\sin\varphi_0 & \cos\varphi_0 \\ \cos\lambda_0\cos\varphi_0 & \sin\lambda_0\cos\varphi_0 & \sin\varphi_0 \end{bmatrix}. \tag{5}$$

B. Notation

To present the state variables in different frames, we use $\boldsymbol{R}_b^e \in \mathbb{R}^{3\times 3}$ and $\boldsymbol{p}_b^e \in \mathbb{R}^3$ to denote the rotation matrix and position vector of frame b relative to frame e. This notation is extended to $\boldsymbol{R}_{b,t}^e$ to represent the states with respect to time t. For motion increments in the same frame, we simplify the notation as $\Delta\boldsymbol{p}_{ij}$ to represent the translational offset of two timestamps i and j. We follow the pose representation $\boldsymbol{T}_b^e = [\begin{smallmatrix}\boldsymbol{R}_b^e & \boldsymbol{p}_b^e \\ \boldsymbol{0} & 1\end{smallmatrix}] \in SE(3)$ to calculate the motion increment [60]. For high-dimensional transition matrices in GP motion models (e.g., $\boldsymbol{\Lambda}(\tau) \in \mathbb{R}^{18\times 18}$), we denote the subblocks $\boldsymbol{\Lambda}_{mn} \in \mathbb{R}^{6\times 6}$ associated with different state components for linear state querying in (21).

C. Continuous-Time Trajectory Representation Using GP

Barfoot et al. [43], [61] originally proposed a continuous-time trajectory representation using GP regression, which presents an exactly sparse kernel by assuming that the system dynamics follow a linear time-invariant SDE (LTI-SDE):

$$\begin{aligned} \dot{\boldsymbol{\gamma}}(t) &= \boldsymbol{A}\boldsymbol{\gamma}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{F}\boldsymbol{w}(t) \\ \boldsymbol{w}(t) &\sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{Q}_c \cdot \delta(t - t')) \end{aligned} \tag{6}$$

where the vector $\boldsymbol{\gamma}(t)$ represents a local state variable. The time-varying system matrices are denoted as $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{F}$, respectively. The input vector $\boldsymbol{u}(t)$ is set to $\boldsymbol{0}$. The process noise $\boldsymbol{w}(t)$ is given as a zero-mean GP with the kernel function formulated with the power spectral density matrix $\boldsymbol{Q}_c \in \mathbb{R}^{6\times 6}$ and the Dirac delta function δ [60].

In discrete time following [61], this state-space model can be furthermore interpreted to interpolate an arbitrary state $\boldsymbol{\gamma}(t_{i+\tau})$ at timestamp $t_{i+\tau} = t_i + \tau$ between two successive local states $\boldsymbol{\gamma}(t_i)$ and $\boldsymbol{\gamma}(t_j)$, where the state timestamps $t_i < t_{i+\tau} < t_j$, using

$$\boldsymbol{\gamma}(t_{i+\tau}) = \boldsymbol{\Lambda}(t_{i+\tau})\boldsymbol{\gamma}(t_i) + \boldsymbol{\Omega}(t_{i+\tau})\boldsymbol{\gamma}(t_j) \tag{7}$$

where

$$\boldsymbol{\Lambda}(t_{i+\tau}) = \boldsymbol{\Phi}(\tau) - \boldsymbol{\Omega}(t_{i+\tau})\boldsymbol{\Phi}(t_j - t_{i+\tau}) \tag{8}$$

$$\boldsymbol{\Omega}(t_{i+\tau}) = \boldsymbol{Q}(\tau)\boldsymbol{\Phi}(t_j - t_{i+\tau})^T\boldsymbol{Q}^{-1}(\tau). \tag{9}$$

The system transition matrix $\boldsymbol{\Phi}$ in (8) and (9) can be defined using a WNOA, aka constant-velocity prior, as demonstrated in earlier works [43], [61]. Later, Tang et al. [16] introduced a

[4]The eccentricity and the Earth's equatorial radius in different satellite systems vary slightly depending on the ellipsoid and the satellite geometry. In this work, we follow the parameter definitions from WGS84 [59].

WNOJ prior that presents third-order system dynamics with the system transition function

$$\boldsymbol{\Phi}(\Delta t) = \begin{bmatrix} \mathbf{1} & \Delta t\mathbf{1} & \frac{1}{2}\Delta t^2\mathbf{1} \\ \mathbf{0} & \mathbf{1} & \Delta t\mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}. \tag{10}$$

The time-varying covariance matrix $\boldsymbol{Q}(\Delta t) \in \mathbb{R}^{18\times 18}$ and its precision matrix $\boldsymbol{Q}^{-1}(\Delta t)$ are computed as

$$\boldsymbol{Q}(\Delta t) = \begin{bmatrix} \frac{1}{20}\Delta t^5\boldsymbol{Q}_c & \frac{1}{8}\Delta t^4\boldsymbol{Q}_c & \frac{1}{6}\Delta t^3\boldsymbol{Q}_c \\ \frac{1}{8}\Delta t^4\boldsymbol{Q}_c & \frac{1}{3}\Delta t^3\boldsymbol{Q}_c & \frac{1}{2}\Delta t^2\boldsymbol{Q}_c \\ \frac{1}{6}\Delta t^3\boldsymbol{Q}_c & \frac{1}{2}\Delta t^2\boldsymbol{Q}_c & \Delta t\boldsymbol{Q}_c \end{bmatrix} \tag{11}$$

$$\boldsymbol{Q}^{-1}(\Delta t) = \begin{bmatrix} 720\Delta t^{-5}\boldsymbol{Q}_c^{-1} & -360\Delta t^{-4}\boldsymbol{Q}_c^{-1} & 60\Delta t^{-3}\boldsymbol{Q}_c^{-1} \\ -360\Delta t^{-4}\boldsymbol{Q}_c^{-1} & 192\Delta t^{-3}\boldsymbol{Q}_c^{-1} & -36\Delta t^{-2}\boldsymbol{Q}_c^{-1} \\ 60\Delta t^{-3}\boldsymbol{Q}_c^{-1} & -36\Delta t^{-2}\boldsymbol{Q}_c^{-1} & 9\Delta t^{-1}\boldsymbol{Q}_c^{-1} \end{bmatrix}. \tag{12}$$

Compared to other approaches, trajectory representation (interpolation) using GP regression effectively incorporates physics-driven models to retrieve realistic vehicle motion by scaling the transition function with the time-varying covariance matrix $\boldsymbol{Q}$. As the hyper-parameter $\boldsymbol{Q}_c$ can be tuned for different applications [62], this approach can be extended for nonlinear problems (see Section IV-D) and enables more accurate state interpolation [7], [46].

D. GP-WNOJ Motion Prior Model

Following the approach in [16], a GP motion prior for SE(3) can be defined as

$$\begin{aligned} \dot{\boldsymbol{T}}(t) &= \boldsymbol{\varpi}(t)^\wedge \boldsymbol{T}(t) \\ \dot{\boldsymbol{\varpi}}(t) &= \boldsymbol{w}(t) \end{aligned} \tag{13}$$

where the vehicle pose in the global frame is denoted as $\boldsymbol{T}(t)$, which can be calculated as $\boldsymbol{T}(t) = \exp\left(\boldsymbol{\xi}(t)^\wedge\right)$ with local pose $\boldsymbol{\xi}(t) = [\boldsymbol{\rho}(t)^T \;\; \boldsymbol{\phi}(t)^T]^T \in \mathbb{R}^6$. The vectors $\boldsymbol{\rho}(t)$ and $\boldsymbol{\phi}(t)$ represent the position and orientation of a local pose (e.g., in the body frame) [63].

A local pose can be converted to $\mathfrak{se}(3)$ by applying the operator $(\cdot)^\wedge$. The operator $(\cdot)^\vee$ is the inverse of $(\cdot)^\wedge$ [60]. The vector $\boldsymbol{\varpi}(t) = [\boldsymbol{\nu}(t)^T \;\; \boldsymbol{\omega}(t)^T]^T \in \mathbb{R}^6$ represents the body-centric velocity. With this motion prior, the state of the GP motion model in a global frame is given as

$$\boldsymbol{x}(t) = \{\boldsymbol{T}(t) \;\; \boldsymbol{\varpi}(t) \;\; \dot{\boldsymbol{\varpi}}(t)\} \in SE(3) \times \mathbb{R}^{12}. \tag{14}$$

However, the GP motion prior in (13) cannot be implemented directly using (6) due to nonlinearity of the system dynamics. To address this problem, Anderson and Barfoot [61] showed that a local linear GP prior can be defined between each state-timestamp pair t_i and t_{i+1} by transforming the global pose $\boldsymbol{T}(t)$ into the local tangent frame, where a local pose $\boldsymbol{\xi}(t)$ can be calculated as

$$\boldsymbol{\xi}_i(t) = \ln(\boldsymbol{T}(t)\boldsymbol{T}_{t_i}^{-1})^\vee, \quad t_i \leq t \leq t_{i+1} \tag{15}$$

where we consider the pose $\boldsymbol{T}_{t_i}$ at the timestamp t_i as a fixed parameter while formulating the local pose $\boldsymbol{\xi}_i(t)$ for an arbitrary pose $\boldsymbol{T}(t)$ for $t > t_i$.

Because the motion between state-timestamp pairs, which are usually associated with high-frequent measurement timestamps (e.g., LiDAR at 10 Hz), is generally small, this local GP prior approximately represents an LTI SDE, which can be driven from (6) by assuming the system matrices remain constant. Thus, a local state variable of GP-WNOJ prior for SE(3) can be defined as

$$\boldsymbol{\gamma}(t) = [\boldsymbol{\xi}(t)^T \;\; \dot{\boldsymbol{\xi}}(t)^T \;\; \ddot{\boldsymbol{\xi}}(t)^T]^T \tag{16}$$

and propagated using (7)–(9). The time derivatives of the local pose can be calculated as

$$\dot{\boldsymbol{\xi}}(t) = \boldsymbol{\mathcal{J}}(\boldsymbol{\xi}_i(t))^{-1}\boldsymbol{\varpi}(t) \tag{17}$$

$$\ddot{\boldsymbol{\xi}}(t) = -\frac{1}{2}(\boldsymbol{\mathcal{J}}(\boldsymbol{\xi}(t))^{-1}\boldsymbol{\varpi}(t))^\curlywedge\boldsymbol{\varpi}(t) + \boldsymbol{\mathcal{J}}(\boldsymbol{\xi}(t))^{-1}\dot{\boldsymbol{\varpi}}(t) \tag{18}$$

where the matrix $\boldsymbol{\mathcal{J}}$ is the left Jacobian of SE(3) [60]. To calculate $\frac{d(\boldsymbol{\mathcal{J}}^{-1})}{dt}$ in closed form for (18), we approximately formulate $\boldsymbol{\mathcal{J}}^{-1} \approx \mathbf{1} - \frac{1}{2}\boldsymbol{\xi}^\curlywedge$ [16].

The operator $(\boldsymbol{\xi})^\curlywedge$ represents the adjoint of $\boldsymbol{\xi}^\wedge \in \mathfrak{se}(3)$ [60], which can be calculated as

$$\boldsymbol{\xi}^\curlywedge = \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\phi} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\phi}^\wedge & \boldsymbol{\rho}^\wedge \\ \mathbf{0} & \boldsymbol{\phi}^\wedge \end{bmatrix}. \tag{19}$$

Because the left Jacobian requires several matrix calculations, it can be approximated as an identity matrix $\mathbf{1}$ over small intervals to improve the computation efficiency [64].[5]

Given a local state variable that represents the origin system state for each state-timestamp pair, we can retrieve the WNOJ motion model for two successive local state variables in the local frame as

$$\boldsymbol{\gamma}_i(t_i) = [\mathbf{0}\boldsymbol{\varpi}(t)^T \;\; \dot{\boldsymbol{\varpi}}(t)^T]^T$$

$$\boldsymbol{\gamma}_i(t_{i+1}) = \begin{bmatrix} \ln(\boldsymbol{T}_{i+i,i})^\vee \\ \boldsymbol{\mathcal{J}}_{i+1}^{-1}\boldsymbol{\varpi}_{i+1} \\ -\frac{1}{2}(\boldsymbol{\mathcal{J}}_{i+1}^{-1}\boldsymbol{\varpi}_{i+1})^\curlywedge\boldsymbol{\varpi}_{i+1} + \boldsymbol{\mathcal{J}}_{i+1}^{-1}\dot{\boldsymbol{\varpi}}_{i+1} \end{bmatrix}. \tag{20}$$

Using the GP-WNOJ prior, a state at an arbitrary time $\tau \in (i, i+1)$ can be queried as

$$\begin{aligned} \boldsymbol{T}_\tau = \exp\Bigg\{\Bigg[&\boldsymbol{\Lambda}_{12}(\tau)\boldsymbol{\varpi}_i + \boldsymbol{\Omega}_{13}(\tau)\dot{\boldsymbol{\varpi}}_i + \boldsymbol{\Sigma}_{11}(\tau)\ln(\boldsymbol{T}_{i+1,i})^\vee \\ &+ \boldsymbol{\Omega}_{12}(\tau)\boldsymbol{\mathcal{J}}_{i+1}^{-1}\boldsymbol{\varpi}_{i+1} \\ &+ \boldsymbol{\Omega}_{13}(\tau)\left(-\frac{1}{2}(\boldsymbol{\mathcal{J}}_{i+1}^{-1}\boldsymbol{\varpi}_{i+1})^\curlywedge\boldsymbol{\varpi}_{i+1} + \boldsymbol{\mathcal{J}}_{i+1}^{-1}\dot{\boldsymbol{\varpi}}_{i+1}\right)\Bigg]^\wedge\Bigg\}\boldsymbol{T}_i \end{aligned} \tag{21}$$

where $\boldsymbol{\Lambda}$ and $\boldsymbol{\Omega}$ are vehicle transition matrices obtained from (7)–(11).

[5]We implement this trick as a configuration in the proposed `gnssFGO`.

Remark 1: Hyperparameter of GP-WNOJ model: As discussed in [16], representing a realistic system transition using the GP motion priors requires proper tuning of the power spectral density matrix $\boldsymbol{Q}_c$. In this work, we assume that $\boldsymbol{Q}_c$ is a constant diagonal matrix defined as $\boldsymbol{Q}_c = \mathrm{diag}(\boldsymbol{q}_c)$ with a 6-D hyperparameter $\boldsymbol{q}_c$. The $\boldsymbol{q}_c$ was manually tuned for the experiments. For more details, see Section VII-E.

E. Measurement Models

1) GNSS Observations: Generally, a single antenna GNSS receiver can provide both position, velocity, and time (PVT) solutions and raw observations. A GNSS receiver equipped with multiple antennas or an inertial sensor can also produce a position, velocity, and altitude (PVA) solution. In this work, we only use the GNSS-PVT solution from a low cost in the loosely coupled fusion because it does not require additional hardware components. The altitude in the GNSS-PVA solution is taken as a reference. As the pose and velocity of the GNSS solution can be directly associated with the state variables in the FGO, we only present the measurement models for the raw GNSS observations: pseudorange ρ and Doppler-shift Δf_k in this section.

In localization approaches that tightly fuse the GNSS observations, pseudorange and Doppler shift are commonly used and well studied [21]. The pseudorange ρ represents a geometric distance between the phase center of the GNSS antenna and the associated satellite, which contains several range delays due to satellite orbit bias and atmospheric delays. The pseudorange can be modeled with respect to the antenna position as

$$\rho_k = \underbrace{\left\|\boldsymbol{p}^e_{\text{ant}} - \boldsymbol{p}^e_{\text{sat},k}\right\|}_{\text{1-D geometric range}} + c_b - c_{b,\text{sat}} + T + I + M + w_{\rho,k} \quad (22)$$

where the vectors $\boldsymbol{p}^e_a$ and $\boldsymbol{p}^e_{\text{sat},k}$ represent the positions of the GNSS antenna and the kth satellite in ECEF frame,[6] respectively. The variables c_b and $c_{b,\text{sat}}$ represent the bias due to receiver clock delay and satellite clock delay. The tropospheric, ionospheric, and multipath delays are denoted as $T(t)$, $I(t)$, and M, respectively. The pseudorange noise is w_ρ.

The Doppler-shift[7] Δf_k measures the difference in frequency between the original and received carrier signal of a satellite, which is usually obtained in the carrier-phase tracking loop [21]. With this observation, the vehicle velocity related to the satellite velocity can be represented as

$$-\lambda_c \Delta f_k = \underbrace{(\boldsymbol{u}^{\text{sat}}_{\text{ant}})^T (\boldsymbol{v}^e_{\text{ant}} - \boldsymbol{v}^e_{\text{sat},k})}_{\text{1-D range rate}} + c_d - c_{d,\text{sat}} + w_{\Delta f_k}. \quad (23)$$

In (23), the constant λ_c is the wavelength of the GNSS signal. The unit vector $\boldsymbol{u}^{\text{sat}}_{\text{ant}}$ represents the direction from the antenna to the kth satellite. We denote the satellite and antenna velocities in the ECEF frame[6] by $\boldsymbol{v}^e_{\text{sat},k}$ and $\boldsymbol{v}^e_{\text{ant}}$, respectively. The receiver clock drift and the satellite clock drift are given as c_d and $c_{d,\text{sat}}$. We use the scalar $w_{\Delta f_k}$ to denote the noise of the measured Doppler shift.

To formulate the pseudorange and Doppler-shift factors, we assume that the satellite clock delay $c_{b,\text{sat}}$ and drift $c_{d,\text{sat}}$ are eliminated using the received navigation messages in a GNSS preprocessing process [7]. We used well-calibrated correction data from a reference station to cancel the tropospheric and ionospheric interference. The multipath error M is not explicitly modeled in this work as m-estimators are used to reject measurement outliers, see Section V-E. We filter out all GNSS observations from satellites with an elevation angle less than 15°.[8]

2) IMU Preintegration: In graph-optimization-based state estimation approaches, the IMU preintegration, introduced in [66], is generally utilized to integrate high-frequency IMU measurements as between-state factors for the optimization procedures running at a lower rate. The preintegrated IMU measurements represent the relative motion increments on manifold. These relative motion increments can be assumed unchanged while relinearizing the consecutive state variables in the optimization iterations, resulting in efficient computation. Due to limited space, see [66] for more details. We use this IMU mechanism to formulate the IMU factor, as presented in Section V-B1.

3) LiDAR Odometry: We adapt the feature extraction and matching methods from a feature-based LiDAR odometry [2], [17] to obtain the relative motion increments between two laser keyframes. The coordinates of raw LiDAR points acquired in different timestamps are recalibrated using the IMU measurement to the original timestamp of the LiDAR scan. We classify the calibrated points into edge and planar features, $\boldsymbol{F}_t = \{\boldsymbol{F}^e_t \;\; \boldsymbol{F}^p_t\}$, based on the smoothness metric shown in [17] and [67]. In scan registration, all k features in $\boldsymbol{F}_{t+1}$ of the current scan are associated with pose priors $\boldsymbol{T}^w_{t+1,1:k}$ and used to find the best transformation $\Delta\boldsymbol{T}^w_{t,t+1}$ from the last laser scan by solving an optimization problem that takes the distance between the corresponding features in $\boldsymbol{F}_t$ using a Gauss–Newton algorithm.

In [2], a LiDAR-centric SLAM approach is presented that optionally fuses the GNSS positioning solution. This approach can only present accurate state estimates if the scan registration converges and sufficient global references (e.g., GNSS position or loop closure) are available. In contrast to [2], we query the vehicle states at scan timestamps from a previously built time-centric graph and integrate the transformation $\Delta\tilde{\boldsymbol{T}}^w_{t,t+1}$ as between-pose constraints, which is used to formulate the between-pose factor (see Section V-B2). After the graph optimization, we query the optimized states again using the GP motion model and update LiDAR keyframe poses in frame w using the following transformation:

$$\boldsymbol{T}^w_{l,t} = \boldsymbol{T}^{e,-1}_{l,\text{anc}} \boldsymbol{T}^e_{l,t} \quad (24)$$

where the transformation matrices $\boldsymbol{T}^e_{l,t}$ and $\boldsymbol{T}^w_{l,t}$ denote LiDAR poses in frame e and frame w, respectively. As LiDAR odometry

[6] We represent the vehicle and satellite position in the ECEF frame instead of the Earth-centered inertial frame for clarity by assuming that the GNSS preprocessing has calibrated the earth rotation during GNSS signal propagation.

[7] In some literature, the Doppler-shift is also formulated as deltarange or pseudorange rate measurement [21], [65]. A positive Doppler shift denotes that the receiver is approaching the tracked satellite.

[8] This is an ad-hoc choice, generally used in ground vehicle navigation approaches and GNSS receivers.

requires a state-space representation in a local-world (aka local-tangent) frame w, where the z-axis is gravity aligned, we query an anchor pose $\boldsymbol{T}^e_{l,\mathrm{anc}} = [\begin{smallmatrix}\boldsymbol{R}^e_{l,\mathrm{anc}} & \boldsymbol{p}^e_{l,\mathrm{anc}} \\ \mathbf{0} & 1\end{smallmatrix}]$ of the LiDAR sensor on first scan and initialize a local-world frame of the LiDAR odometry by setting the anchor pose as its origin. In contrast to [10], a coarse orientation estimate is unnecessary in our work to align the local-world frame and the navigation frame because a prior vehicle heading is provided by the dual-antenna GNSS receiver.

4) Optical Speed Sensor: We employ a high-grade vehicle optical speed sensor that provides unbiased 2-D velocity observations $\bar{\boldsymbol{v}}^b_t$ in the body frame at 100 Hz. The 2-D velocity observations can be associated with the vehicle velocity in the state vector using

$$\tilde{\boldsymbol{v}}^b_t = \begin{bmatrix}\tilde{v}^b_{t,x} \\ \tilde{v}^b_{t,y}\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 \\ 0 & 1 & 0\end{bmatrix} \cdot \boldsymbol{R}^b_e \boldsymbol{v}^e_b \tag{25}$$

where the vector $\tilde{\boldsymbol{v}}^b_t$ represents the observed 2-D velocity components in frame b, which can be evaluated with the vehicle velocity variable $\boldsymbol{v}^e_b$ transformed with the inverse rotation matrix $\boldsymbol{R}^b_e$ back to frame b.

V. FGO for Vehicle Localization

This section presents our implementation of the proposed `gnssFGO` for two sensor fusion schemes. In loosely coupled fusion, we fuse the PVT solution from a low-cost GNSS receiver with the IMU measurements, the observed 2-D vehicle velocity from a high-grade speed sensor, and the LiDAR odometry. To defend the superiority of fusing raw GNSS observations for vehicle localization, we propose a tightly coupled fusion of raw GNSS observations with IMU measurements and LiDAR odometry, which is evaluated with the baseline trajectory. In this section, we introduce all probabilistic factor formulations and the proposed factor graph structures.

A. State Variables

The state variable at timestamp t in this work is defined as

$$\boldsymbol{x}_t \triangleq \{\boldsymbol{T}^e_{b,t} \;\; \boldsymbol{v}^e_{b,t} \;\; \boldsymbol{b}^{\mathrm{a}}_{b,t} \;\; \boldsymbol{b}^{\mathrm{g}}_{b,t} \;\; \boldsymbol{c}^{\mathrm{r}}_t\}. \tag{26}$$

We estimate the vehicle pose $\boldsymbol{T}^e_{b,t} \in SE(3)$ and 6-D velocity $\boldsymbol{v}^e_b$ in frame e. The vectors $\boldsymbol{b}^{\mathrm{a}}_{b,t}$ and $\boldsymbol{b}^{\mathrm{g}}_{b,t}$ denote the 3-D biases of the accelerometer and gyroscope, respectively. The 2-D vector $\boldsymbol{c}^{\mathrm{r}}_t = [c_{b,t} \;\; c_{d,t}]^T$ represents the GNSS receiver clock bias $c_{b,t}$ and drift $c_{d,t}$, which is only estimated by the tightly coupled fusion of raw GNSS observations.

Remark 2: Acceleration of GP-WNOJ: Unlike [16], we do not estimate 6-D accelerations in GP motion models to reduce the dimension of the state vector. Instead, we consider the vehicle accelerations measured by the IMU as inputs to the WNOJ model.

B. Factor Formulations

1) Preintegrated IMU Factor: Following [66], we define the error function of the IMU factor between two consecutive state variables at timestamps t_i, t_j as

$$\left\|\boldsymbol{e}^{\mathrm{imu}}_{ij}\right\|^2 = \left\|\left[\boldsymbol{r}^T_{\Delta \boldsymbol{R}_{ij}} \;\; \boldsymbol{r}^T_{\Delta \boldsymbol{v}_{ij}} \;\; \boldsymbol{r}^T_{\Delta \boldsymbol{p}_{ij}}\right]^T\right\|^2_{\boldsymbol{\Sigma}^{\mathrm{imu}}} \tag{27}$$

where

$$\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}} = \mathrm{Log}(\Delta \tilde{\boldsymbol{R}}_{ij}(\boldsymbol{b}^{\mathrm{g}}_i))\boldsymbol{R}^T_i \boldsymbol{R}_i \tag{28}$$

$$\boldsymbol{r}_{\Delta \boldsymbol{v}_{ij}} = \boldsymbol{R}^T_i(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g}\Delta t_{ij}) - \Delta \tilde{\boldsymbol{v}}_{ij}(\boldsymbol{b}^{\mathrm{g}}_i, \boldsymbol{b}^{\mathrm{a}}_i) \tag{29}$$

$$\boldsymbol{r}_{\Delta \boldsymbol{p}_{ij}} = \boldsymbol{R}^T_i\left(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i\Delta t_{ij} - \frac{1}{2}\boldsymbol{g}\Delta t^2_{ij}\right) - \Delta \tilde{\boldsymbol{p}}_{ij}(\boldsymbol{b}^{\mathrm{g}}_i, \boldsymbol{b}^{\mathrm{a}}_i). \tag{30}$$

In (28)–(30), we omit the bias derivatives that can be ignored between two state variables. The motion increments $\{\Delta \tilde{\boldsymbol{R}}_{ij} \;\; \Delta \tilde{\boldsymbol{v}}_{ij} \;\; \Delta \tilde{\boldsymbol{p}}_{ij}\}$ are provided by the IMU preintegration with

$$\Delta \tilde{\boldsymbol{R}}_{ij} = \prod_{k=i}^{j-1} \mathrm{Exp}((\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}^{\mathrm{g}}_i - \boldsymbol{\eta}^{\mathrm{g}}_k)\Delta t) \tag{31}$$

$$\Delta \tilde{\boldsymbol{v}}_{ij} = \sum_{k=i}^{j-1} \Delta \tilde{\boldsymbol{R}}_{ik}(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}^{\mathrm{a}}_i - \boldsymbol{\eta}^{\mathrm{a}}_k)\Delta t \tag{32}$$

$$\Delta \tilde{\boldsymbol{p}}_{ij} = \sum_{k=i}^{j-1} \left[\Delta \boldsymbol{v}_{ik}\Delta t + \frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}^{\mathrm{a}}_i - \boldsymbol{\eta}^{\mathrm{a}}_k)\Delta t^2\right] \tag{33}$$

where the raw vehicle acceleration $\tilde{\boldsymbol{a}}$ and rotation rate $\tilde{\boldsymbol{\omega}}$ from the IMU are integrated. The predefined noise parameters $\{\boldsymbol{\eta}_{\mathrm{a}} \;\; \boldsymbol{\eta}_{\mathrm{g}}\}$ are propagated to acquire the covariance matrix $\boldsymbol{\Sigma}^{\mathrm{imu}}$ [66]. The gravity vector is updated according to the current position in the e frame for each preintegration.

As in [66], we estimate the accelerometer and gyroscope biases with the Brownian motion model by formulating the bias error function as

$$\left\|\boldsymbol{e}^b_{ij}\right\|^2 = \left\|\boldsymbol{b}^{\mathrm{a}}_j - \boldsymbol{b}^{\mathrm{a}}_i\right\|^2_{\boldsymbol{\Sigma}^{\mathrm{ba}}} + \left\|\boldsymbol{b}^{\mathrm{g}}_j - \boldsymbol{b}^{\mathrm{g}}_i\right\|^2_{\boldsymbol{\Sigma}^{\mathrm{bg}}}. \tag{34}$$

2) Between-Pose Factor: For the relative odometry observations $\Delta \tilde{\boldsymbol{T}}^e_{i,j} = \{\Delta \tilde{\boldsymbol{R}}^e_{i,j} \;\; \Delta \tilde{\boldsymbol{p}}^e_{i,j}\}$, we follow the original implementation in [68] and formulate the between pose factor represented as

$$\left\|\boldsymbol{e}^{\mathrm{bp}}_{i,j}\right\|^2 = \left\|\ln(\boldsymbol{T}^{e,-1}_i \boldsymbol{T}^e_j \Delta \tilde{\boldsymbol{T}}^e_{i,j})^\vee\right\|^2_{\boldsymbol{\Sigma}^{\mathrm{bp}}} \tag{35}$$

where the pose $\boldsymbol{T}^e_i$ and $\boldsymbol{T}^e_j$ are queried using timestamps associated with two successive LiDAR scans.

3) Velocity Factor: We use the 2-D observations $\tilde{\boldsymbol{v}}^b_t$ to formulate the navigation velocity factor. As the measured velocity can be directly associated with the velocity in state variables, as denoted in (25), we formulate the error function for the velocity observations considering the lever arm $\boldsymbol{l}^{b,\mathrm{vel}}$ from the body frame to the sensor center as

$$\left\|\boldsymbol{e}^{\mathrm{vel}}_i\right\|^2 = \left\|\begin{bmatrix}1 & 0 & 0 \\ 0 & 1 & 0\end{bmatrix} \cdot (\boldsymbol{R}^b_{e,i}\boldsymbol{v}^e_{b,i} + \boldsymbol{\omega}^{b\wedge}_i \boldsymbol{l}^{b,\mathrm{vel}}) - \tilde{\boldsymbol{v}}^b_i\right\|^2_{\boldsymbol{\Sigma}^{\mathrm{vel}}}. \tag{36}$$

4) GNSS-PVT Factor: We propose a generalized implementation of the GNSS-PVT factor for the observed antenna position $\tilde{\boldsymbol{p}}^e_{\text{ant}}$ and the velocity $\tilde{\boldsymbol{v}}^n_{\text{ant}}$. Taking into account the lever arm $\boldsymbol{l}^b_{\text{ant}}$ from the IMU center to the phase center of the GNSS antenna, we calculate the antenna position at timestamp t_i as $\boldsymbol{p}^e_{\text{ant},i} = \boldsymbol{p}^e_{b,i} + \boldsymbol{R}^e_{b,i}\boldsymbol{l}^b_{\text{ant}}$ and velocity as $\boldsymbol{v}^e_{\text{ant},i} = \boldsymbol{v}^e_{b,i} + \boldsymbol{R}^e_{b,i}(\boldsymbol{\omega}^b_i)^\wedge \boldsymbol{l}^b_{\text{ant}}$. Thus, the error function can be derived as

$$\left\|\boldsymbol{e}^{\text{pvt}}_i\right\|^2 = \left\|[\boldsymbol{r}^T_{\boldsymbol{p}_i} \quad \boldsymbol{r}^T_{\boldsymbol{v}_i}]^T\right\|^2_{\boldsymbol{\Sigma}^{\text{pvt}}} \tag{37}$$

with

$$\boldsymbol{r}_{\boldsymbol{p}_i} = \boldsymbol{p}^e_{\text{ant},i} - \tilde{\boldsymbol{p}}^e_{\text{ant},i} \tag{38}$$

$$\boldsymbol{r}_{\boldsymbol{v}_i} = \boldsymbol{R}^n_{e,i}\boldsymbol{v}^e_{\text{ant},i} - \tilde{\boldsymbol{v}}^n_{\text{ant},i} \tag{39}$$

where the rotation matrix $\boldsymbol{R}^n_{e,i}$ is given in (4) by substituting the geodetic coordinate of the main antenna $\boldsymbol{p}^{\text{LLH}}_{\text{ant,i}}$ as the origin. We use the measured standard deviations in the GNSS solutions to formulate the covariance matrix $\boldsymbol{\Sigma}^{\text{pvt}}$.

5) Pseudorange and Doppler-Shift (PrDo) Factor: We derive the error function for the preprocessed pseudorange and Doppler-shift observations with (22) and (23) as

$$\left\|\boldsymbol{e}^{\text{PrDo}}_i\right\|^2 = \left\|[r^{\text{Pr}}_i \quad r^{\text{Do}}_i]^T\right\|^2_{\boldsymbol{\Sigma}^{\text{PrDo}}} \tag{40}$$

where

$$r^{\text{Pr}}_i = \left\|\boldsymbol{p}^e_{\text{ant},i} - \boldsymbol{p}^e_{\text{sat},k,i}\right\| + c_{b,i} - \rho_{k,i} \tag{41}$$

$$r^{\text{Do}}_i = (\boldsymbol{u}^{\text{sat}}_{\text{ant},i})^T\left(\boldsymbol{v}^e_{\text{ant},i} - \boldsymbol{v}^e_{\text{sat},k,i}\right) + c_{d,i} + \lambda_c \Delta f_{k,i}. \tag{42}$$

We consider a scaled carrier-to-noise ratio (C/N_0) with hyperparameters λ_ρ and $\lambda_{\Delta f_k}$ to represent the variance of pseudorange and Doppler-shift observations, which is denoted as

$$\eta^2_\rho = \lambda_\rho 10^{-\frac{\text{C/N}_0}{10}} \text{ and } \eta^2_{\Delta f} = \lambda_{\Delta f} 10^{-\frac{\text{C/N}_0}{10}}. \tag{43}$$

6) GNSS Receiver Clock Error Factor: In the tight coupling of the raw GNSS observations, the unknown receiver clock bias and drift (cbd) are estimated in the state variable by assuming a constant drifting model, which can be fused as

$$\left\|\boldsymbol{e}^{\text{cbd}}_i\right\|^2 = \left\|\begin{bmatrix}1 & \Delta t\\ 0 & 1\end{bmatrix}\begin{bmatrix}c_{b,i-1}\\ c_{d,i-1}\end{bmatrix} - \begin{bmatrix}c_{b,i}\\ c_{d,i}\end{bmatrix}\right\|^2_{\boldsymbol{\Sigma}^{\text{cbd}}}. \tag{44}$$

7) GP-WNOJ Motion Prior Factor: We implement the GP-WNOJ motion model as between-state factors, similar to [45]. The error function was originally given in [16] using (20). We summarize this error function for convenience as

$$\left\|\boldsymbol{e}^{\text{gp}}_{ij}\right\|^2 = \left\|[\boldsymbol{r}^T_{\Delta\boldsymbol{\gamma}_{ij}} \quad \boldsymbol{r}^T_{\Delta\boldsymbol{\varpi}_{ij}}]^T\right\|^2_{\boldsymbol{\Sigma}^{\text{gp}}} \tag{45}$$

where

$$\boldsymbol{r}_{\Delta\boldsymbol{\gamma}_{ij}} = \ln(\boldsymbol{T}_{j,i})^\vee - (t_j - t_i)\boldsymbol{\varpi}_i - \frac{1}{2}(t_j - t_i)^2\dot{\boldsymbol{\varpi}}_i \tag{46}$$

$$\boldsymbol{r}_{\Delta\boldsymbol{\varpi}_{ij}} = \boldsymbol{\mathcal{J}}^{-1}_{j,i}\boldsymbol{\varpi}_j - \boldsymbol{\varpi}_i - (t_j - t_i)\dot{\boldsymbol{\varpi}}_i. \tag{47}$$

As introduced in Section V-A, we used the measured accelerations of the IMU in our GP motion models. Thus, only the 6-D pose and the 6-D velocity are evaluated in GP-WNOJ motion factors, so that $\boldsymbol{e}^{\text{gp}}_{ij} \in \mathbb{R}^{12}$. The analytical Jacobians of the GP motion models can be found in [16] and [69].

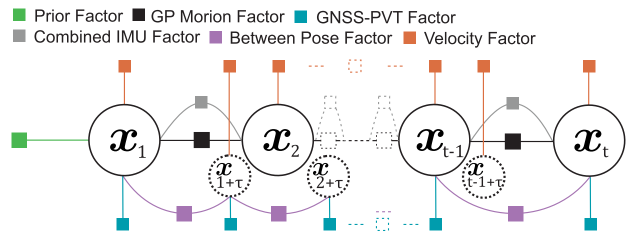


Fig. 5. General graph of loose coupling in `gnssFGO`.

C. Loosely Coupled FGO

Although the loosely coupled fusion with GNSS and IMU measurements has been shown to be less performant compared to tight coupling [6], we implemented a loosely coupled fusion of sensor observations, including the 2-D speed sensor and the LiDAR odometry, to 1) study the performance gain by fusing multiple sensor observations; 2) evaluate the loosely and tightly coupled fusion for GNSS-based vehicle localization in challenging areas; and 3) demonstrate the flexibility and scalability of the proposed method.

The proposed factor graph is shown in Fig. 5. The states $\boldsymbol{x}_{1:t}$ are created deterministically on the graph independently of any measurement. If a measurement cannot be associated with any state variable, a state $\hat{\boldsymbol{x}}_{i+\tau}$ between two state variables $\hat{\boldsymbol{x}}_i$ and $\hat{\boldsymbol{x}}_{i+1}$ (where $t_i < \tau < t_{i+1}$) is queried for the error evaluation.

The optimization problem can then be formulated as

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\text{argmin}}\left(\left\|\boldsymbol{e}^0\right\|^2_{\boldsymbol{\Sigma}_0} + \sum_{i=1}^{M}\left\|\boldsymbol{e}^{\text{imu}}_i\right\|^2_{\boldsymbol{\Sigma}^{\text{imu}}} + \sum_{i=1}^{M}\left\|\boldsymbol{e}^{\text{gp}}_i\right\|^2_{\boldsymbol{\Sigma}^{\text{gp}}} + \right.$$
$$\left. + \sum_{i=1}^{N}\left\|\boldsymbol{e}^{\text{vel}}_i\right\|^2_{\boldsymbol{\Sigma}^{\text{vel}}} + \sum_{i=1}^{K}\left\|\boldsymbol{e}^{\text{pvt}}_i\right\|^2_{\boldsymbol{\Sigma}^{\text{pvt}}} + \sum_{i=1}^{J}\left\|\boldsymbol{e}^{\text{bp}}_i\right\|^2_{\boldsymbol{\Sigma}^{\text{bp}}}\right) \tag{48}$$

where the error term $\boldsymbol{e}^0$ represents the prior factor obtained at initialization or from marginalization. Because sensor observations are received asynchronously other than estimation timestamps M, we use different index notations N, K, and J to indicate the number of sensor observations in (48).

D. Tightly Coupled FGO

In contrast to the loosely coupled fusion approach, a tightly coupled fusion of raw GNSS observations contributes more constraints with multiple observed satellites to state variables, as illustrated in Fig. 6. Unlike Fig. 5, we include the pseudorange and Doppler-shift factors in the graph, providing redundant constraints to each state variable. To improve the robustness while GNSS observations are degraded or lost in challenging areas, we include LiDAR odometry as between-state constraints to improve the consistency of the estimated trajectory. The receiver clock error factor is also added to the graph. In this fusion mechanism, we do not fuse the measurements from the 2-D velocity, which is not commonly used in vehicle localization approaches, aiming to highlight the robustness of the tightly coupled fusion [see discussion in (Section VII-B2) and (VII-B3)].

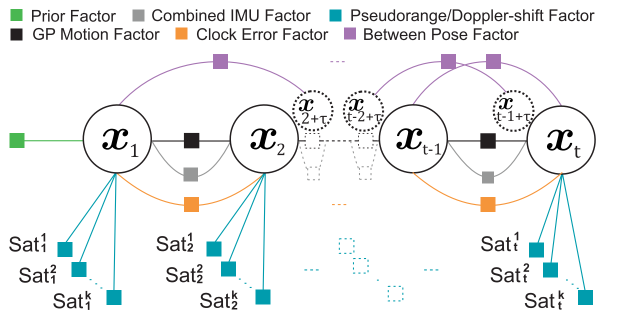


Fig. 6. General graph of tight coupling in `gnssFGO` with k raw satellite observations at each timestamp t denoted as Sat_t^k.

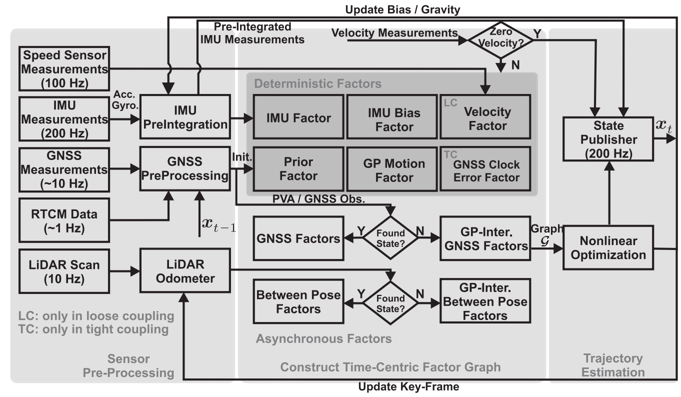


Fig. 7. System overview showing all data interfaces and factor types by implementing Algorithm 1.

The optimization problem with sensor observations from different time domains becomes

$$\hat{\boldsymbol{X}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \Bigg(\left\|\boldsymbol{e}_i^0\right\|_{\boldsymbol{\Sigma}_0}^2 + \sum_{i=1}^{M} \left\|\boldsymbol{e}_i^{\mathrm{IMU}}\right\|_{\boldsymbol{\Sigma}^{\mathrm{IMU}}}^2 + \sum_{i=1}^{M} \left\|\boldsymbol{e}_i^{\mathrm{gp}}\right\|_{\boldsymbol{\Sigma}^{\mathrm{gp}}}^2 + \sum_{i=1}^{N} \left\|\boldsymbol{e}_i^{\mathrm{bp}}\right\|_{\boldsymbol{\Sigma}^{\mathrm{bp}}}^2 + \sum_{i=1}^{M} \left\|\boldsymbol{e}_i^{\mathrm{cbd}}\right\|_{\boldsymbol{\Sigma}^{\mathrm{cbd}}}^2 + \sum_{i=1}^{J} \sum_{s=1}^{K} \left\|\boldsymbol{e}_{s,i}^{\mathrm{PrDo}}\right\|_{\boldsymbol{\Sigma}^{\mathrm{PrDo}}}^2 \Bigg). \tag{49}$$

E. Noise Models

In this work, we formulate the covariance matrices of the GNSS-related factors using noise values provided in GNSS observations, as presented in Section V-B4 and V-B5. The IMU noise is characterized by the Allan noise parameters,[9] which is used to calculate $\boldsymbol{\Sigma}^{\mathrm{imu}} \in \mathbb{R}^{6\times 6}$, $\boldsymbol{\Sigma}^{\mathrm{ba}} \in \mathbb{R}^{3\times 3}$, and $\boldsymbol{\Sigma}^{\mathrm{bg}} \in \mathbb{R}^{3\times 3}$. Because there are no noise indicators for LiDAR odometry, speed sensor, and receiver clock errors, we formulate the covariance matrices $\boldsymbol{\Sigma}^{\mathrm{bp}} \in \mathbb{R}^{6\times 6}$, $\boldsymbol{\Sigma}^{\mathrm{vel}} \in \mathbb{R}^{2\times 2}$, and $\boldsymbol{\Sigma}^{\mathrm{cbd}} \in \mathbb{R}^{2\times 2}$ manually as diagonal matrices using ad-hoc parameters. Noise models of different factors can be configured to use m-estimators [60]. In our experiments, we use the m-estimator with *Cauchy* loss [70] in the factors, such as GNSS-PVT factors, which may be affected by outlier measurement due to strong corruption in urban areas.

F. System Overview

The system overview with the implementation of Algorithm 1 and all data interfaces is shown in Fig. 7. The sensor data are received and preprocessed in separate processes. We construct the time-centric factor graph in a two-stage process, as introduced in Algorithm 1. The first stage (line 4–10 of Algorithm 1) includes between-state factors and delay-free IMU factors to build a deterministic graph on time. Subsequently, asynchronous sensor observations are fused into the deterministic graph by aligning the timestamps between the measurement and the state variables (line 11–24 of Algorithm 1). For measurements that cannot be aligned with any state, two successive state variables are queried to construct a GP-interpolated state for measurement evaluation in optimization procedures. The time-centric graph can be optimized using a fixed-lag batch optimizer [71] or a fixed-lag incremental smoother iSAM2 [72] at a lower frequency. In the experimental results, the estimated trajectories in the error metrics are optimized using iSAM2. We also evaluate both smoothers with respect to both estimator performance and computation efficiency, as presented in Section VII-D. After each optimization procedure, we forward the optimized state variables to a state publisher and sensor preprocessing modules. The state publisher is associated with the IMU sensor and provides high-frequent state estimates at 200 Hz.

Remark 3: Near-Zero-Velocity Detection: While the vehicle is stationary, the state estimation exhibits random pose drift. This is a known problem in vehicle localization using inertial measurements [73]. In this case, the state observability degrades dramatically due to insufficient IMU excitation, leading to unbounded error accumulation. Thus, we follow the idea proposed in [73] to detect near-zero velocity motion by voting through multiple sensors that provide velocity information. If the vehicle is voted to be stationary, we temporally pause the graph optimization and state propagation.

Remark 4. Optimization Frequency: By default, we extend and optimize the time-centric graph at 10 Hz in our experiments to achieve a good balance of accuracy and runtime efficiency. Although these frequencies can be flexibly configured in the proposed estimation framework, we found that optimizing the graph at 5 Hz is a threshold to avoid discontinuities (jumps) of the estimated trajectory in our application. For applications restricted by low-performance computing devices, choosing a higher frequency to extend the graph and a lower frequency for optimization can be considered.

G. Implementation

We implemented our approach in C++ using Robot Operating System `ROS2`.[10] The open-source software library `GTSAM`[11] was extended to implement the graph and factor formulations.

[9][Online]. Available: https://github.com/ori-drs/allan_variance_ros

[10][Online]. Available: https://docs.ros.org/en/humble/index.html

[11][Online]. Available: https://gtsam.org

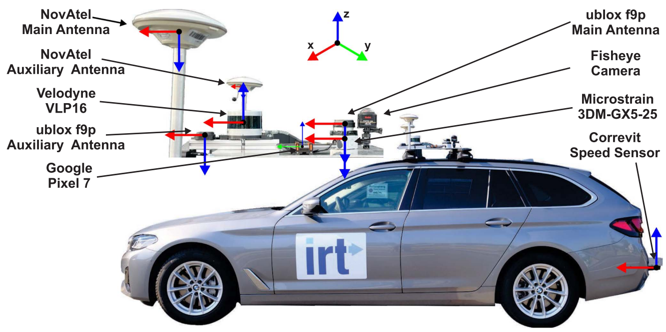


Fig. 8. Sensor setup and frames on the test vehicle.

TABLE I
TEST SEQUENCES DEFINITION

Seq.	Leng. (km)	Tunnel (m)	Dura. (s)	$\bar{v}$ (km/h)	$\bar{n}^{\mathrm{sat}}$.	$R^{\mathrm{RTK}}_{\mathrm{fixed}}$ (%)	R^{NS} (%)
AC	17.0	270	2477	27.25	11	76.51	1.7
DUS	5.25	-	1350	13.48	8	52.06	0.9
C01	0.81	276	160	17.89	7	60.8	31.78
C02	1.45	145	390	13.36	7	37.74	11.56
HS	10.6	-	300	124.82	14	94.9	1.47

We denote the test sequences in Aachen, Dusseldorf, Cologne, and high-speed tracks with "AC", "DUS", "C", and "HS", respectively. The variable $\bar{v}$ represents the average speed and the scalar $\bar{n}^{\mathrm{sat}}$. is the average number of satellites used for a GNSS-PVA solution. We calculated the ratio of RTKfixed solution and No-Solution due to insufficient GNSS observations denoted by $R^{\mathrm{RTK}}_{\mathrm{fixed}}$ and R^{NS}, correspondingly.

We adopted the software solution for LiDAR odometry from `LIO-SAM`,[12] where only the front-end feature extraction and association were adapted in our work. We used the positioning and orientation estimation solution from a dual-antenna GNSS setup to initialize the state variable $\boldsymbol{x}_0$. In this work, we used a laptop with an Intel i9-9900K, 16 cores at maximum 4.7 GHz and 64 GB memory for sensor preprocessing and graph optimization in experimental studies.

VI. Measurement Setup and Test Sequences

A. Measurement Setup

In the measurement campaigns, we recorded sensor data of long-range routes in different areas of Aachen, Düsseldorf, and Cologne. Our sensor setup included two GNSS receivers, a Microstrain 3DM-GX5 IMU, and a Velodyne VLP-16 LiDAR. Both the high-grade GNSS receiver (NovAtel PwrPak7D-E1) and the low-grade GNSS receiver (ublox f9p) were equipped with dual GNSS antennas and served with RTK correction data received from a base station. A high-grade optical speed sensor, Correvit S-Motion DTI from Kistler, was mounted on the trailer hitch on the vehicle's rear side. The sensor-equipped test vehicle is shown in Fig. 8. We have manually calibrated the static transformations between different sensors mounted on the roof rack of the test vehicle. Static transformation of the speed sensor to other sensors was measured using a Leica total station. These static transformations are assumed to be constant in all experiments. For more details, see our code.[2]

The IMU data were acquired at 200 Hz, while LiDAR point-clouds were recorded at 10 Hz. GNSS observations from the NovAtel and ublox receiver were recorded at 10 and 5 Hz, respectively. We used the high-grade GNSS receiver Novatel PwrPark7D-E1 with a dual constellation of GPS and Galileo satellite systems as a reference source. In addition to the sensor data, we received the pulse-per-second signal (1PPS) from the NovAtel receiver at 1 Hz to calculate the measurement delays. The RTCMv3 (RTK) correction data from the German satellite positioning service *SAPOS*[13] was also stored at about 1 Hz for GNSS preprocessing.

B. Test Sequences

Our dataset contains different driving scenarios: open-sky, semi-/dense-urban, and high-speed track. For a clear evaluation, we define different test sequences throughout multiple measurement campaigns and analyze the driving conditions for each sequence, as shown in Table I. The test sequences include lengthy runs with a maximum 17-km route, aiming to evaluate the estimation performance for long-term operations. For test sequences in urban areas, we chose data from scenarios with different urbanization rates containing tunnel and bridge crossings to evaluate the limitations of the proposed fusion approaches. In addition, we also considered open-sky areas on the high-speed track, where a maximum vehicle speed of 170 km/h was reached, creating significant motion distortion in the LiDAR point clouds.

C. Reference Trajectory and Metrics

To evaluate the proposed fusion strategies, we employ the RTK-fixed GNSS-PVA solution associated with low uncertainties ($\sigma_{\mathrm{pos}} < 0.05\,\mathrm{m}$ and $\sigma_{\mathrm{rot}} < 1^\circ$) to calculate the absolute root mean square error. Besides the error metrics, we employ the Pythagoras' theorem implemented in the Open Motion Planning Library[14] (OMPL) to calculate the trajectory smoothness (contrary to trajectory roughness) for all test sequences, aiming to provide a relative performance metrics. The smoothness is given as the sum of angles between all path segments in the local-world frame, as denoted in (50), where the variables a_i, b_i, and c_i are the length of the trajectory segments containing three successive vehicle positions in the Euclidean frame. For the same test sequence with k vehicle positions, a smaller s shows a high smoothness of the trajectory. In this work, we used the propagated states from the state publisher at a high frequency to calculate the smoothness s

$$s = \sum_{i=2}^{k-1} \left(\frac{2\left(\pi - \arccos \frac{a_i^2 + b_i^2 - c_i^2}{2a_i b_i}\right)}{a_i + b_i} \right)^2. \tag{50}$$

[12][Online]. Available: https://github.com/TixiaoShan/LIO-SAM

[13][Online]. Available: http://www.sapos.nrw.de

[14][Online]. Available: https://ompl.kavrakilab.org/

VII. Experiments and Results

A. Experiment Design

To evaluate the proposed `gnssFGO`, we first benchmark the loosely coupled fusion of the GNSS solution with the IMU, 2-D speed sensor, and a LiDAR-centric SLAM approach `LIO-SAM` [2], aiming to evaluate the robustness of the proposed method. Compared to other multisensor fusion approaches, `LIO-SAM` represents a classical multisensor fusion framework performing well in outdoor scenarios where LiDAR odometry is the primary sensor. For a fair evaluation, we have adapted the `LIO-SAM` implementation[6] using the same robust error models and parameterizations as in our method. We also enable the loop-closure detection in `LIO-SAM` to maximize state estimation performance. We follow the implementation in [2] to eliminate motion distortions in LiDAR points using the IMU measurements.

Furthermore, we evaluate two fusion mechanisms with different sensor modalities. In the loosely coupled fusion, we conduct fusion configurations of the IMU and the GNSS-PVT solution with and without multisensor including a 2-D speed sensor and LiDAR odometry (w. and w/o. MultiSensor). Later, we propose similar experiments by fusing raw GNSS observations and IMU measurements with and without LiDAR odometry in a tight coupling (w. and w/o. LiDAR), which is expected to present a more robust trajectory estimation in challenging areas compared to the loose coupling.

Finally, we discuss the smoother type and computation time using different lag sizes. We also evaluate the GP-WNOJ prior and the GP-WNOA prior.

B. General Error Metrics

With predefined test sequences in Table I, we present the general error metrics for all experiments in Table II by taking the RTK-fixed GNSS-PVA solution as the ground truth. Because an RTK-fixed solution is not available in challenging areas, we denote the solution rate used as a ground-truth reference to calculate the error metrics of each test sequence as a percentage in the column "Seq." Due to limited space, figures cannot be presented on a full scale; we thus upload all interactive figures in `GitHub`.[15]

1) LiDAR-Centric Fusion: As shown in Table II, the LiDAR-centric SLAM approach `LIO-SAM` failed in several test sequences even when the same factors with robust error modeling were used and loop-closure detection was enabled (see video demonstration[2]). The most frequent reason is that the scan registration fails due to an invalid feature association, which can be observed in all failed test sequences. Another possible reason for the failure can be associated with corrupted GNSS observations that show inconsistent noise values, resulting in a divergence in optimization. Fig. 1(b) demonstrates this result, where the estimate diverged and cannot be recovered after the vehicle entered a tunnel. In Seq. HS, the LiDAR-centric approach cannot even be properly initialized while the vehicle moves very fast, which was not observed in the proposed `gnssFGO`. Furthermore, the estimated velocities and orientations using `LIO-SAM` show large variation and therefore less robustness compared to both fusion mechanisms in `gnssFGO` [see Fig. 10(c) and (d)].

Discussion: We observe that because graph construction triggered by the LiDAR odometer in [2] requires strict timestamp synchronization of GNSS measurements with LiDAR timestamps, asynchronous GNSS measurements are dropped. Although LiDAR odometry and detected loop closures are still available, dropping GNSS measurements results in a loss of effective state constraints, so graph optimization becomes more sensitive to inaccurate scan registration. Therefore, the trajectory's smoothness and the estimate's accuracy are dramatically penalized (see Table II). This hypothesis is supported in the test sequences DUS and C01 (see Figs. 10 and 11), where the estimated height, orientation, and velocities were frequently diverted. Therefore, it can be observed that trajectory drift cannot be effectively eliminated using the classic sensor-centric localization approach `LIO-SAM`. Even worse, the robustness and reliability of sensor-centric approaches cannot be guaranteed in challenging areas once the primary sensor is compromised. As online applications raise computation time and resource requirements, sensor degradation due to, e.g., insufficient data processing becomes nontrivial. The proposed `gnssFGO` presents an effective workaround while fusing multiple sensors to eliminate the dependence on a single sensor, enabling the fusion of lossless information and improving the robustness of the estimate if sensor failure can be expected.

2) Loosely Coupled (LC) Fusion: It can be observed in Table II that the accuracy of the low-grade GNSS receiver, especially in the vertical dimension (height), is dramatically degraded compared to the high-grade GNSS receiver. Furthermore, this receiver does not characterize its noise values, so the standard deviations provided are strongly inconsistent with the real noise. Therefore, fusing the PVT solution from the low-grade GNSS receiver significantly downgrades the performance of the loosely coupled fusion in all test sequences, even when multiple sensor observations are fused. In a less challenging environment (Seq. AC) [see Fig. 9], the proposed loosely coupled fusion generally outperforms the original PVT solution. However, the same improvement cannot always be expected in challenging environments when comparing the error metrics of other test sequences. The primary reason for this result is caused by inconsistent noise values and highly inaccurate height measurements in the PVT observations. Another interesting phenomenon is that the loosely coupled fusion of the PVT solution with other sensors can present a degraded performance (see Seq. C01 and Seq. HS). Due to inaccurate height measurements that present high variations [see Figs. 10(b) and 11(b)], the pose of the LiDAR keyframes cannot be stably optimized, so the keyframes, which are used to calculate the relative motion of each scan, present different height values. In such cases, the estimated LiDAR odometry is associated with incorrect relative motion increments, which degrades graph optimization [see height in Fig. 11(b) and Vd in Fig. 11(d)].

To improve the robustness of the estimation by acquiring redundant state constraints, we used the high-grade speed sensor

[15][Online]. Available: https://github.com/rwth-irt/gnssFGO/tree/ros2/online_fgo/plots_tro

TABLE II
GENERAL TRAJECTORY ESTIMATION METRICS

Seq.	Configuration	Mean 2D Pos. Err. (m)	2D Pos. STD (m)	Max. 2D Pos. Err. (m)	Mean 3D Pos. Err. (m)	3D Pos. STD (m)	Max. 3D Pos. Err. (m)	Mean Yaw Err. (°)	Yaw STD (°)	Max . Yaw Err. (°)	Smooth. s
AC (52%)	ublox-f9p	0.889	±0.237	12.905	99.424	±0.293	263.24	6.651	±37.54	359.9	1.07e+6
	LIO-SAM [2]	Failed (diverged occasionally)									
	ours (lc) w/o. MultiSensor	0.732	±0.152	12.378	**0.918**	±0.217	12.619	42.984	±4.57	90.0	7301.52
	ours (lc) w. MultiSensor	0.770	±**0.149**	15.299	0.960	±**0.1964**	15.770	44.365	±2.14	89.99	1439.82
	ours (tc) w/o. LiDAR	0.936	±0.650	5.877	1.276	±1.392	6.127	2.76	±2.68	88.53	1706.63
	ours (tc) w. LiDAR	**0.476**	±0.738	**1.605**	1.057	±1.088	**2.343**	**1.18**	±**2.13**	**85.12**	**982.40**
DUS (18%)	ublox-f9p	3.441	±2.413	19.284	76.939	±3.121	276.001	1.956	±62.44	9.368	8250.5
	LIO-SAM [2]	**0.736**	-	9.129	**1.024**	-	17.391	1.758	-	80.44	2.15e+5
	ours (lc) w/o. MultiSensor	4.138	±1.228	19.544	9.256	±1.813	36.166	40.202	±5.41	89.745	1.02e+7
	ours (lc) w. MultiSensor	2.631	±0.462	9.402	7.668	±**0.950**	39.806	39.697	±1.33	89.980	1.19e+9
	ours (tc) w/o. LiDAR	1.123	±1.081	**4.415**	1.461	±2.524	7.141	**0.473**	±3.01	81.32	3595.72
	ours (tc) w. LiDAR	1.074	±**0.822**	4.570	2.170	±1.890	**4.789**	0.511	±**2.40**	**6.018**	**1510.48**
C01 (57%)	ublox-f9p	3.126	±52.01	21.583	106.731	±52.872	259.399	76.175	±40.81	258.040	1.580e+4
	LIO-SAM [2]	3.339	-	8.186	3.478	-	8.463	3.774	-	34.260	6.30e+5
	ours (lc) w/o. MultiSensor	4.290	±2.350	23.662	10.085	±3.307	41.421	82.340	±10.1	89.830	435.89
	ours (lc) w. MultiSensor	7.697	±1.258	17.487	12.221	±2.050	45.104	87.164	±**1.77**	88.384	1.20e+5
	ours (tc) w/o. LiDAR	0.727	±1.52	1.125	0.78	±3.194	1.179	0.935	±3.209	6.534	5.38e+3
	ours (tc) w. LiDAR	**0.402**	±**0.846**	**0.807**	**0.468**	±**1.892**	**0.841**	**0.888**	±2.76	**7.025**	**122.358**
C02 (27%)	ublox-f9p	3.632	±14.69	6.745	76.043	±16.053	256.092	2.4326	±91.67	357.521	1.12e+4
	LIO-SAM [2]	Failed (diverged occasionally)									
	ours (lc) w/o. MultiSensor	7.421	±1.350	33.417	25.919	±2.020	88.612	28.084	±7.86	40.242	562.92
	ours (lc) w. MultiSensor	4.029	±**0.479**	20.393	18.227	±**1.058**	53.803	10.673	±4.87	58.392	7.29e+6
	ours (tc) w/o. LiDAR	0.933	±1.111	1.933	1.188	±2.836	2.104	0.927	±4.15	**3.811**	**198.03**
	ours (tc) w. LiDAR	**0.264**	±1.039	**0.499**	**0.328**	±2.402	**0.565**	**0.496**	±**2.40**	9.716	218.42
HS (78%)	ublox-f9p	0.645	±**0.217**	**1.361**	33.486	±**0.273**	66.942	0.977	±**0.236**	**2.453**	**0.004**
	LIO-SAM [2]	Failed (diverged on initialization)									
	ours (lc) w/o. MultiSensor	1.354	±0.520	6.010	2.167	±0.729	11.873	87.287	±3.63	89.156	0.248
	ours (lc) w. MultiSensor	3.136	±1.299	15.571	8.051	±1.634	41.482	88.902	±0.98	89.954	0.300
	ours (tc) w/o. LiDAR	**0.404**	±0.509	2.265	**0.822**	±1.08	4.833	0.993	±1.69	10.05	0.139
	ours (tc) w. LiDAR	0.475	±0.672	3.121	1.400	±1.252	**3.609**	**0.699**	±0.94	3.778	0.060

Calculating the smoothness as a relative performance metric is independent of the ground-truth trajectory.
A test run is classified as failed if the algorithm diverges. We show the RTK-fixed PVA solution ratio in percent from the high-grade NovAtel GNSS receiver under each sequence's name in column Seq. (e.g., AC (52%)) that is used to calculate the error metrics.
Best results are shown in bold.

in this fusion approach, expecting it to effectively constrain the unobserved states once the GNSS solutions are compromised. However, our experiments indicate that the 2-D velocity measurements provided by the 2-D speed sensor cannot sufficiently constrain the state space, especially when vehicle orientation cannot be observed.

Discussion: Although the loosely coupled fusion using the proposed `gnssFGO` did not fail in our test sequences, we can observe the same conclusion, as shown in [6], that loosely coupled fusion cannot serve as a robust state estimator in challenging areas where GNSS solutions present inconsistent uncertainties. However, we indicate that this fusion mechanism can present fast estimation convergence as long as accurate GNSS-PVT solutions are available. This is because the state variables (position and velocity) can be directly observed in the GNSS-PVT solution, where the measurement model does not present high nonlinearity, and thus, effective state constraints are presented.

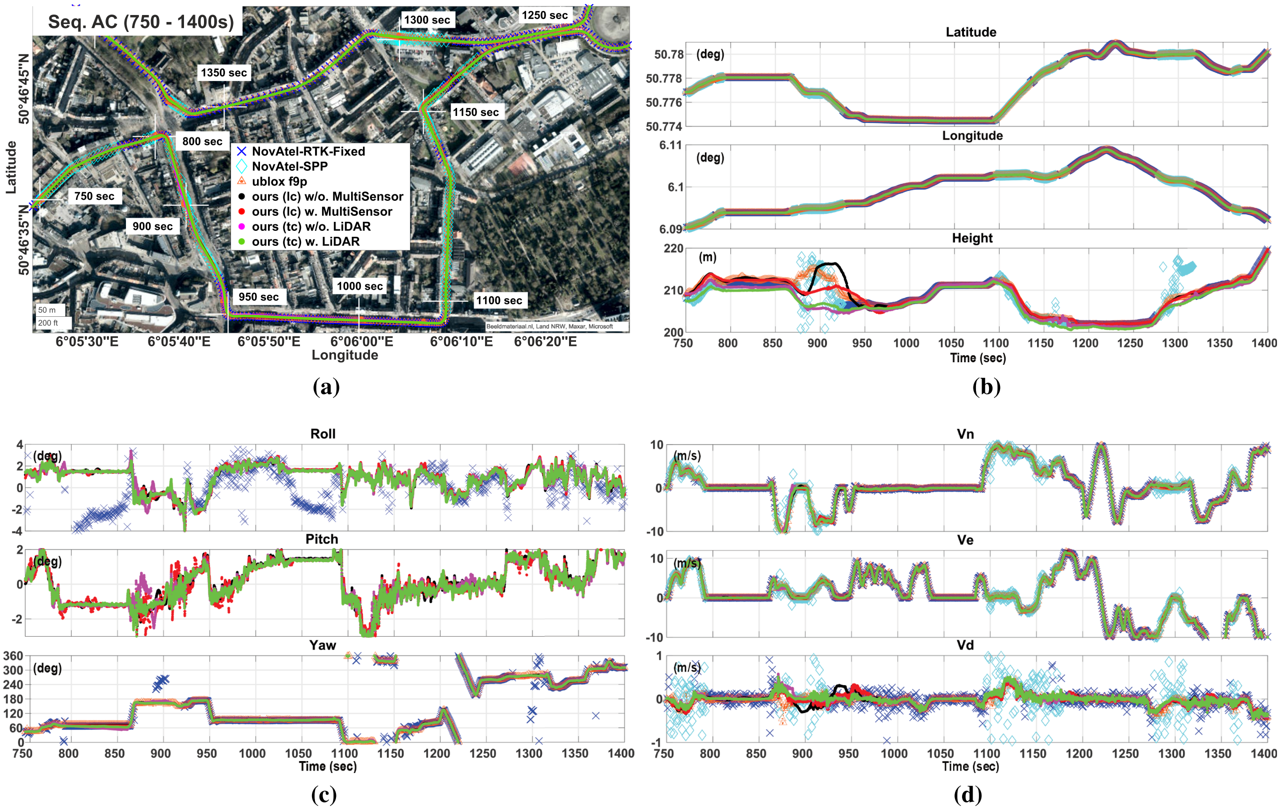


Fig. 9. Trajectory plot (700–1400 s) in urban areas in Aachen. We plot the GNSS single point position (SPP) if the RTK-fixed solution is unavailable. (a) Trajectories of Seq. AC. (b) Coordinates in the WGS84 frame. (c) Estimated rotation in the NED frame. (d) Estimated velocity in the NED frame.

3) Tightly Coupled (TC) Fusion: Compared to loosely coupled sensor fusion, fusing preprocessed GNSS observations in a tight coupling contributes to redundant state constraints. Thus, the tightly coupled fusion can generally present more robust trajectory estimations with a smaller maximum position error and larger trajectory smoothness in lengthy runs, except in the high-speed scenario (Seq. HS). In challenging urban areas, such as Seq. C01 and C02, fusing the LiDAR odometry as between-state constraints generally improves the estimation performance and trajectory smoothness. This conclusion can also be drawn when referring to Figs. 10(b) and 11(b), where more accurate height and velocity estimations can be observed by fusing LiDAR odometry in the graph. In high-speed scenarios, LiDAR scans suffer from serious motion distortion, and no sufficient features can be extracted compared to urban areas. Therefore, a limited performance improvement can be observed by fusing LiDAR odometry in the graph.

Discussion: Based on the experimental results presented above, a robust trajectory estimation can be achieved in challenging scenarios using the proposed approach by fusing multiple sensor measurements in a tight coupling, which supports our hypothesis proposed in Section II. In contrast to the loose coupling, the tightly coupled multisensor fusion presents a more robust trajectory estimation in our experimental studies. The same conclusion has also been shown in [6] and [10]. However, acceptable accuracy cannot be achieved, especially in dense urban scenarios. For instance, although all estimated trajectories using the proposed `gnssFGO` in Fig. 1 remain consistent, a large drift is presented using the proposed sensor integration. Possible reasons to explain this phenomenon can be traced back to LiDAR degradation, insufficient GNSS observations, and inconsistent sensor noise models due to the presence of outliers. As `gnssFGO` provides a flexible fusion mechanism, this problem can be addressed by integrating more effective state constraints into the graph.

C. Challenging Scenarios

In this part, we propose experimental studies regarding GNSS observations, LiDAR odometry, and solver settings. We also evaluated the GP-WNOA and GP-WNOJ priors and discussed the hyperparameter tuning for $\boldsymbol{Q}_c$.

1) Loss of GNSS Observation: Generally, losing GNSS observations in a short time interval does not lead to immediate divergence or trajectory drift if multiple state constraints, such as LiDAR odometry or motion prior factors are still presented. This conclusion can be drawn from our experiment in Seq. C01, where the vehicle crossed a large bridge at the central train station in Cologne, as shown in Fig. 11. It is also interesting to observe that the loss of GNSS observations is frequently accompanied by highly corrupted GNSS observations due to multipath effects. For example, fusion approaches can diverge when the vehicle approaches or leaves a tunnel. In this case, robustness can be significantly affected even if no local sensors are fused.

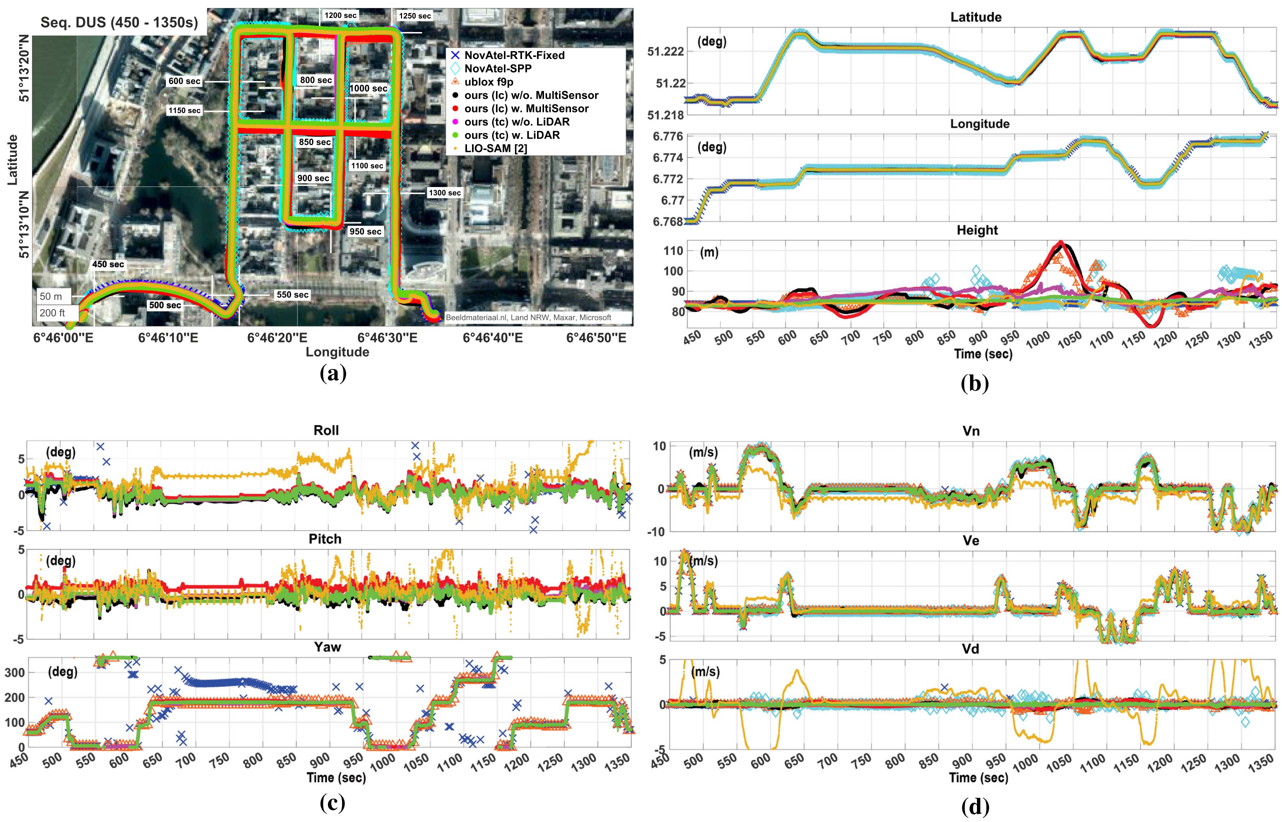


Fig. 10. Trajectory plot (450–1350 s) in challenging areas in Düsseldorf. (a) Trajectories of Seq. DUS. (b) Coordinates in the WGS84 frame. (c) Estimated rotation in the NED frame. (d) Estimated velocity in the NED frame.

Furthermore, fusing GNSS observations in a tight coupling extends the state variables with receiver cbd $\boldsymbol{c}^r = [c_b \ \ c_d]^T$, which become unobservable if less than four satellites are visible. Fig. 12 shows the estimated clock bias c_b with respect to the number of tracked satellites. The estimated clock bias drifts dramatically in the graph where only GNSS observations and IMU measurements are integrated. Even if the observability of the clock bias can be recovered, it takes some time until the state variable $\boldsymbol{c}^r$ converges (see Fig. 12), which downgrades the overall estimation performance. Similar results can be observed in other experiments where unobservable state variables can lead to estimation divergence and an ill-posed optimization problem. Fortunately, this problem can be eliminated in the graph fused with LiDAR odometry. Thanks to the between-state constraints that prevent other state variables (e.g., position and rotation) from divergence, robust trajectory estimation can be guaranteed. In addition, a large trajectory drift can be expected if the global reference (e.g., GNSS observations) is lost over a long time interval, such as when crossing a long tunnel.

2) Highly Corrupted GNSS Observations: Compared to the temporary loss of GNSS observations, we emphasize that including highly corrupted GNSS observations in the graph has a greater impact on estimation performance. This conclusion can be supported by Seq. C02, where the accuracy of our proposed fusion paradigms is significantly degraded in GNSS-corrupted areas, as shown in Fig. 1. In Fig. 13, we plot the estimated trajectories in this scenario by transforming the coordinates in the navigation frame (ENU). A large trajectory drift up to 25 m can be observed in tightly coupled fusion without LiDAR odometry (see Fig. 13). Although fusing relative motion constraints, such as odometry, can effectively constrain divergence, trajectory drifts cannot be eliminated until valid global references are acquired.

3) LiDAR Odometry Degradation: As discussed in [74], traditional LiDAR odometry algorithms suffer from dramatic degradation in unstructured environments and high-speed scenarios. This problem can also be observed in our experiments. Fig. 14 illustrates three scenarios in which the accuracy of the LiDAR odometry is penalized if the vehicle is driving in featureless areas or in high-speed mode with an average vehicle speed of 125 km/h. In low-speed driving mode and open-sky areas, LiDAR degradation does not reduce the estimation performance, while high-quality GNSS measurements are available. However, if the vehicle moves at high speed, the LiDAR odometry becomes inaccurate because of motion distortion. Therefore, including LiDAR odometry factors in the graph can decrease localization accuracy, as Table II of Seq presents HS. In scenarios with long tunnels, trajectory drifting can always be expected due to the loss of global reference. This presents the major limitation of classic LiDAR odometers that calculate only pose increments. Another crucial aspect to be mentioned is the uncertainty of LiDAR odometry used in fusion approaches. Because no covariance is provided by classic scan matching algorithms, the predefined noise parameters (see Section V-E) may be

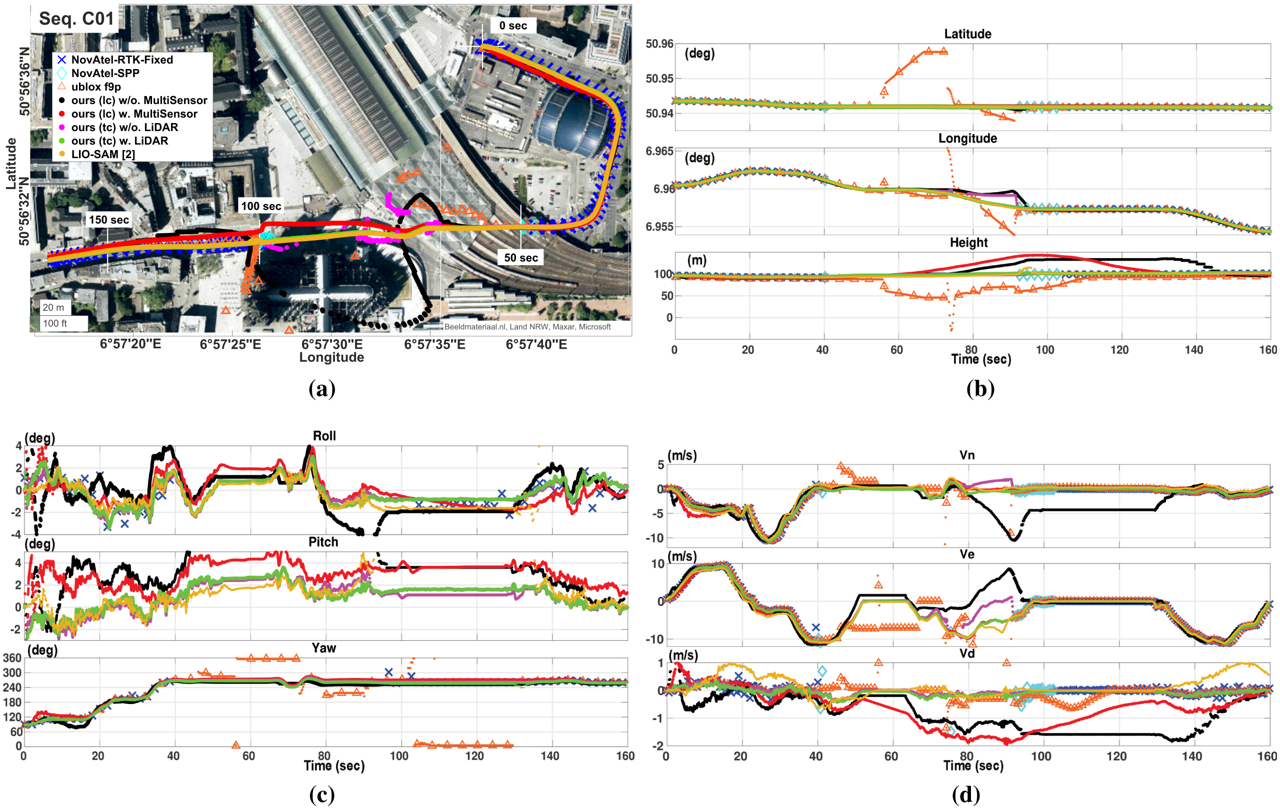


Fig. 11. Trajectory plot near the central station of cologne (Seq. C01). For the tightly coupled fusion, we omitted the near-zero velocity detection to present trajectory drifting while the receiver clock error is unobservable. (a) Trajectories of Seq. C01. (b) Coordinates in the WGS84 frame. (c) Estimated rotation in the NED frame. (d) Estimated velocity in the NED frame.

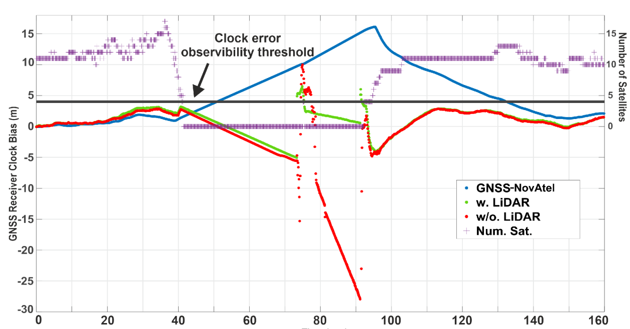


Fig. 12. Estimated GNSS receiver clock bias of Seq. C01 without near-zero-velocity detection. The receiver clock error is unobservable during tunnel crossing. In this scenario, fusing raw GNSS observations without LiDAR odometry cannot constrain the state estimation, resulting in trajectory drifting. The corresponding trajectories of both fusion approaches are presented in Fig. 11.

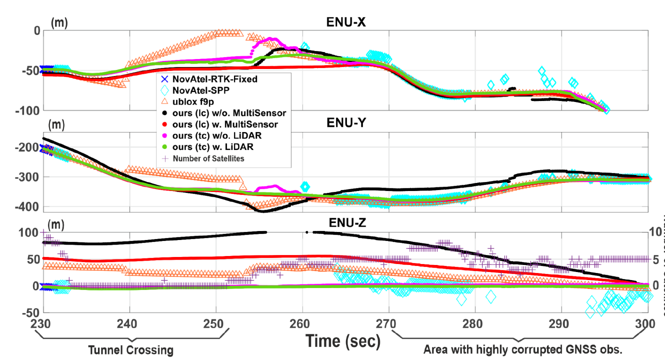


Fig. 13. Coordinates in the ENU frame of Seq. C02. The estimated trajectories become unsmooth if the GNSS observations are strongly corrupted in urban areas.

overconfident. Therefore, we emphasize the importance of acquiring realistic uncertainty quantification for LiDAR odometry in future work.

D. Smoother Type and Computation Time

To study the impact of different smoother types and lag sizes, we evaluated batch and incremental smoother `iSAM2` with different lag sizes for Seq. DUS. The performance metrics are presented in Table III. Compared to an incremental smoother, solving the optimization problem with a batch optimizer does not show a considerable improvement in accuracy. This happens because the graph structure becomes more similar to a Markov chain in large-scale localization applications with fewer loop-closure constraints. In this scenario, relinearizing all past state variables does not contribute more information that improves the accuracy. For loosely coupled fusion, the batch smoother presents a smoother trajectory. However, this advantage is absent with the incremental smoother when fusing GNSS observations in a tight coupling.

Furthermore, the batch optimizer requires more computational resources than the incremental smoother (see Fig. 15), especially in urban areas with more measurement outliers. In online applications, estimation accuracy and trajectory smoothness can be penalized once optimization takes longer. This conclusion

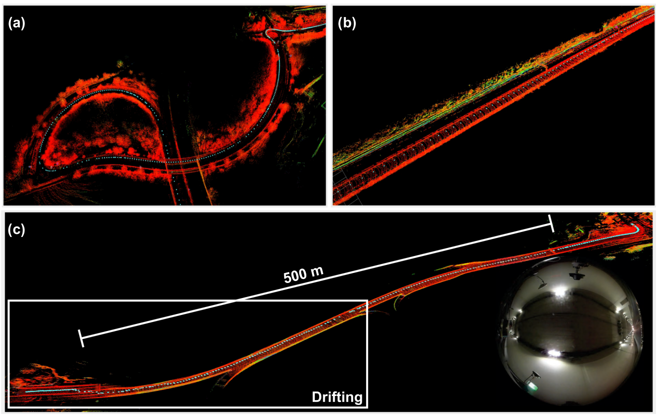


Fig. 14. Examples of LiDAR odometry degradation in three scenarios. (a) Unstructured feature-less area. (b) High-speed scenario. (c) Long tunnel (400 m).

TABLE III
ESTIMATION PERFORMANCE OF SEQ. DUS USING DIFFERENT SOLVER CONFIGURATIONS

Configuration	Mean 2-D Pos. Err. (m)	Mean 3-D Pos. Err. (m)	Mean Yaw Err. (°)	Smooth. s
lc-batch-3sec	0.324	0.839	0.208	1526.4
lc-isam2-3sec	0.397	1.146	0.216	5672.8
tc-batch-3sec	1.543	2.008	0.497	1649.3
tc-isam2-10sec	1.419	1.733	0.573	1537.9
tc-isam2-3sec	1.524	1.766	0.511	1510.5

Configuration: (fusion type)-(solver type)-(lag size), e.g., lc-batch-3sec

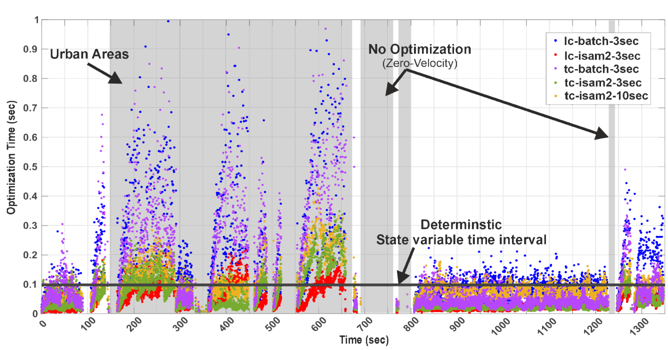


Fig. 15. Computation time with different configurations of Seq. DUS.

is supported by referring to the tightly coupled fusion in Table III. Similarly to the optimizer type, considering a large lag size does not contribute significantly. Moreover, even the incremental smoother with a large lag size frequently violates the desired optimization frequency, resulting in inefficient optimization procedures.

E. GP-WNOA/WNOJ Motion Model

In this section, we evaluated the continuous-time trajectory representation using the GP interpolation with both WNOJ and WNOA models. For a fair evaluation, the hyperparameter $\boldsymbol{q}_c$ was manually tuned by penalizing the vehicle pose in $\boldsymbol{Q}_c$ with the same parameterization for pose weighting, aiming to evaluate the effect between the third-order and second-order dynamics models.

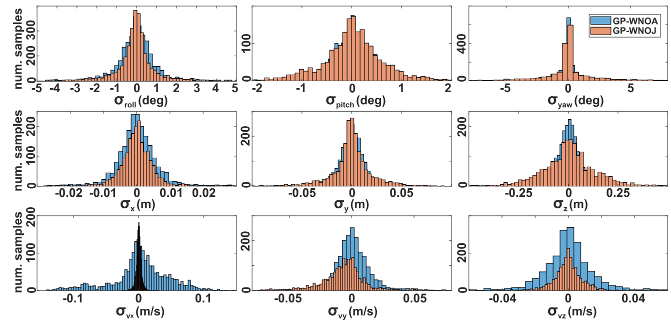


Fig. 16. Histogram of whitened state errors of GP models in an open-sky area.

Compared to the GP-WNOJ model, a GP-WNOA model assumes that the system transition follows a constant velocity model [7], [43]. As discussed in [16], representing vehicle trajectories with an approximately constant-velocity model may be insufficient in urban driving scenarios where the vehicle accelerates and brakes frequently. To evaluate the performance of both GP models, we chose a part of Seq. AC containing 200 s test run in open-sky areas, where the PVT solution from the high-grade GNSS receiver presents the ground-truth trajectory. We calculate the whitened error of the vehicle pose and the linear velocity in the body frame and plot the results on the histogram in Fig. 16. Because the GP-WNOJ model represents second-order system dynamics, it shows smaller errors in all linear velocity components. Both models perform similarly in position estimation, where the GP-WNOJ is more accurate in the main motion direction $x-$axis. For rotation, the GP-WNOJ does not present considerable improvements compared to the GP-WNOA. One possible reason supporting this result can be traced back to rotational acceleration that cannot be observed directly using the IMU (see Section V-A).

We have validated that the GP motion model formulates a valid continuous-time trajectory representation. However, tuning the power spectral matrix $\boldsymbol{Q}_c$ that scales the system transition in the GP kernel has a large effect on numerical stability and estimation performance [16]. Although the GP-WNOJ model presents reliable velocity estimates compared to the GP-WNOA model, it shows higher sensitivity on the power spectral matrix $\boldsymbol{Q}_c$ [16], which therefore requires more careful parameter tuning when incorporating accelerations in state propagation.

Remark 5. Tuning of $\boldsymbol{Q}_c$: In this work, we did not explicitly investigate parameter tuning for the power spectral matrix $\boldsymbol{Q}_c$. As discussed in the original works [16] and [61], this hyperparameter can be calibrated using supervised-learning approaches. Recent works also explored this idea and showed the possibility of learning this parameter without ground-truth labels using variational Bayesian approaches [62], [75]. However, vehicle dynamics presents a high variation in real-world driving scenarios, the parameter $\boldsymbol{Q}_c$ should be dynamically tuned in an online process, which remains our future work.

VIII. Conclusion

This article proposes an online factor graph optimization that generalizes multisensor fusion for robust trajectory estimation focusing on GNSS. The vehicle trajectory is represented in continuous time using a GP motion prior that enables arbitrary state querying, presenting a sensor-independent graph optimization. We successfully fused asynchronous sensor measurements into the proposed method for robust vehicle localization in challenging environments. The experimental studies show that the proposed method is robust, flexible, accurate, and works online with multiple datasets collected from challenging scenarios. All our FGO configurations succeed in all test sequences, whereas the classic state-of-the-art LiDAR-centric method [2] failed in some situations. Observed from the experimental results, the GP-WNOJ motion prior enables accurate trajectory representations in continuous time with properly tuned hyperparameters.

In this work, we did not fully exploit the GNSS observations, such as carrier-phase, which requires complicated techniques to resolve the ambiguities and detect the satellite cycle slips. Our framework can also utilize advanced techniques to exclude multipath and non-line-of-sight GNSS observations. We also neglected the hyperparameter tuning of the GP-WNOJ model and sensor noise identification, which can be solved online using learning-based methods. In addition, additional sensor modalities, such as visual odometry, can be utilized in the fusion framework to improve performance. In summary, these research objectives formulate our future work.

Acknowledgment

The authors would like to thank Robin Taborsky from the Institute of Automatic Control at RWTH Aachen University and the public order office in Aachen, Düsseldorf, and Cologne for their great support in measurement campaigns. The authors also thank David Yoon and Keenan Burnett from the Autonomous Space Robotics Laboratory at the University of Toronto for their discussions and support in this work.

References

[1] N. Zhu, J. Marais, D. Bétaille, and M. Berbineau, "GNSS position integrity in urban environments: A review of literature," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 2762–2778, Sep. 2018.

[2] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.

[3] W. Burgard, O. Brock, and C. Stachniss, *Map-Based Precision Vehicle Localization in Urban Environments*. Cambridge, MA, USA: MIT Press, 2008.

[4] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 2, no. 3, pp. 194–220, Sep. 2017.

[5] C. Sung, S. Jeon, H. Lim, and H. Myung, "What if there was no revisit? Large-scale graph-based SLAM with traffic sign detection in an HD map using LiDAR inertial odometry," *Intell. Service Robot.*, vol. 15, pp. 161–170, 2022.

[6] W. Wen, T. Pfeifer, X. Bai, and L.-T. Hsu, "Factor graph optimization for GNSS/INS integration: A comparison with the extended Kalman filter," *NAVIGATION: J. Inst. Navigation*, vol. 68, no. 2, pp. 315–331, 2021.

[7] H. Zhang, X. Xia, M. Nitsch, and D. Abel, "Continuous-time factor graph optimization for trajectory smoothness of GNSS/INS navigation in temporarily GNSS-denied environments," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 9115–9122, Oct. 2022.

[8] F. Dellaert, "Factor graphs: Exploiting structure in robotics," *Annu. Rev. Control, Robot., Automat. Syst.*, vol. 4, no. 1, pp. 141–166, 2021.

[9] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional LiDAR-based system for long-term and wide-area people behavior measurement," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 2, 2019, Art. no. 1729881419841532.

[10] S. Cao, X. Lu, and S. Shen, "GVINS: Tightly coupled GNSS-visual-inertial fusion for smooth and consistent state estimation," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2004–2021, Aug. 2022.

[11] J. Zhang, W. Wen, H. Feng, X. Chen, and L.-T. Hsu, "Continuous GNSS-RTK aided by LiDAR/Inertial odometry with intelligent GNSS selection in urban canyons," in *Proc. 34th Int. Tech. Meeting Satell. Division Inst. Navigation*, Sep. 2021, pp. 4198–4207.

[12] W. Wen, X. Bai, and L.-T. Hsu, "3D vision aided GNSS real-time kinematic positioning for autonomous systems in urban canyons," *NAVIGATION: J. Inst. Navigation*, vol. 70, no. 3, 2023, Art. no. 590.

[13] L.-T. Hsu, "Analysis and modeling GPS NLOS effect in highly urbanized area," *GPS Solutions*, vol. 22, Nov. 2017, Art. no. 7.

[14] K. Burnett, Y. Wu, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "Are we ready for radar to replace LiDAR in all-weather mapping and localization?," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 10328–10335, 2022.

[15] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex urban dataset with multi-level sensors from highly diverse urban environments," *Int. J. Robot. Res.*, vol. 38, no. 6, pp. 642–657, 2019.

[16] T. Y. Tang, D. J. Yoon, and T. D. Barfoot, "A White-Noise-on-Jerk motion prior for continuous-time trajectory estimation on SE(3)," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 594–601, Apr. 2019.

[17] J. Zhang and S. Singh, "Low-drift and real-time LiDAR odometry and mapping," *Auton. Robots*, vol. 41, pp. 401–416, Feb. 2017.

[18] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[19] N. Sünderhauf and P. Protzel, "Towards robust graphical models for GNSS-based localization in urban environments," in *Proc. Int. Multi-Conf. Syst., Signals, Devices*, 2012, pp. 1–6.

[20] R. Watson and J. Gross, "Robust navigation in GNSS degraded environment using graph optimization," in *Proc. 30th Int. Tech. Meeting Satell. Division Inst. Navigation*, Sep. 2017, pp. 2906–2918.

[21] P. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd ed. Norwood, MA, USA: Artech House, 2013.

[22] W. Wen, Y. C. Kan, and L.-T. Hsu, "Performance comparison of GNSS/INS integrations based on EKF and factor graph optimization," in *Proc. 32th Int. Tech. Meeting Satell. Division Inst. Navigation*, Sep. 2019, pp. 3019–3032.

[23] W. Wen, X. Bai, Y. C. Kan, and L.-T. Hsu, "Tightly coupled GNSS/INS integration via factor graph and aided by fish-eye camera," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 10651–10662, Nov. 2019.

[24] W. Wen and L.-T. Hsu, "Towards robust GNSS positioning and real-time kinematic using factor graph optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5884–5890.

[25] X. Bai, W. Wen, and L.-T. Hsu, "Time-correlated window-carrier-phase-aided GNSS positioning using factor graph optimization for urban positioning," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, no. 4, pp. 3370–3384, Aug. 2022.

[26] T. Suzuki, "GNSS Odometry: Precise trajectory estimation based on carrier phase cycle slip estimation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7319–7326, Jul. 2022.

[27] B. Congram and T. D. Barfoot, "Field testing and evaluation of single-receiver GPS odometry for use in robotic navigation," *Field Robot.*, vol. 2, pp. 1849–1873, 2022.

[28] T. Pfeifer, P. Weissig, S. Lange, and P. Protzel, "Robust factor graph optimization: A comparison for sensor fusion applications," in *Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Automat.*, 2016, pp. 1–4.

[29] T. Suzuki, "1st place winner of the smartphone decimeter challenge: Two-step optimization of velocity and position using smartphone's carrier phase observations," in *Proc. 35th Int. Tech. Meeting Satell. Division Inst. Navigation*, Sep. 2022, pp. 2276–2286.

[30] T. Pfeifer and P. Protzel, "Incrementally learned mixture models for GNSS localization," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 1131–1138.

[31] W. Wen, G. Zhang, and L.-T. Hsu, "GNSS outlier mitigation via graduated non-convexity factor graph optimization," *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 297–310, Jan. 2022.

[32] Z. Gong et al., "Graph-based adaptive fusion of GNSS and VIO under intermittent GNSS-degraded environment," *IEEE Trans. Instrum. Meas.*, vol. 70, 2021, Art. no. 8501116.

[33] Y. Wang et al., "Rail vehicle localization and mapping with LiDAR-vision-inertial-GNSS fusion," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 9818–9825, Oct. 2022.

[34] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled LiDAR-Visual-Inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1004–1011, Apr. 2021.

[35] W. Shao, S. Vijayarangan, C. Li, and G. Kantor, "Stereo visual inertial LiDAR simultaneous localization and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 370–377.

[36] J. Nubert, S. Khattak, and M. Hutter, "Graph-based multi-sensor fusion for consistent localization of autonomous construction robots," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 10048–10054.

[37] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super odometry: IMU-centric LiDAR-Visual-Inertial estimator for challenging environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 8729–8736.

[38] G. Cioffi and D. Scaramuzza, "Tightly-coupled fusion of global positional measurements in optimization-based visual-inertial odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5089–5095.

[39] S. Boche, X. Zuo, S. Schaefer, and S. Leutenegger, "Visual-inertial SLAM with tightly-coupled dropout-tolerant GPS fusion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 7020–7027.

[40] C. Hu, S. Zhu, Y. Liang, and W. Song, "Tightly-coupled visual-inertial-pressure fusion using forward and backward IMU preintegration," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6790–6797, Jul. 2022.

[41] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 2088–2095.

[42] D. Hug, P. Bänninger, I. Alzugaray, and M. Chli, "Continuous-time stereo-inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6455–6462, Jul. 2022.

[43] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch continuous-time trajectory estimation as exactly sparse Gaussian process regression," in *Proc. Conf. Robot.: Sci. Syst.*, 2014, pp. 1–9.

[44] M. Mukadam, J. Dong, F. Dellaert, and B. Boots, "STEAP: Simultaneous trajectory estimation and planning," *Auton. Robots*, vol. 43, pp. 415–434, 2019.

[45] X. Yan, V. Indelman, and B. Boots, "Incremental sparse GP regression for continuous-time trajectory estimation and mapping," *Robot. Auton. Syst.*, vol. 87, pp. 120–132, 2017.

[46] J. Dong, J. G. Burnham, B. Boots, G. C. Rains, and F. Dellaert, "4-D crop monitoring: Spatio-temporal reconstruction for agriculture," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 3878–3885.

[47] J. Johnson, J. Mangelson, T. D. Barfoot, and B. Randal, "Continuous-time trajectory estimation: A comparative study between Gaussian process and spline-based approaches," 2024, *arXiv:2402.00399*.

[48] M. Labbé and F. Michaud, "Memory management for real-time appearance-based loop closure detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 1271–1276.

[49] M. Labbé and F. Michaud, "RTAB-Map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, 2019.

[50] J. Vincent, M. Labbé, J.-S. Lauzon, F. Grondin, P.-M. Comtois-Rivet, and F. Michaud, "Dynamic object tracking and masking for visual SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 4974–4979.

[51] M. Labbé and F. Michaud, "Multi-session visual SLAM for illumination-invariant re-localization in indoor environments," *Front. Robot. AI*, vol. 9, 2022, Art. no. 801886.

[52] M. Colosi et al., "Plug-and-play SLAM: A unified SLAM architecture for modularity and ease of use," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5051–5057.

[53] J. Solà et al., "WOLF: A modular estimation framework for robotics based on factor graphs," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4710–4717, Apr. 2022.

[54] M. Warren, M. Greeff, B. Patel, J. Collier, A. P. Schoellig, and T. D. Barfoot, "There's no place like home: Visual teach and repeat for emergency return of multirotor UAVs during GPS failure," *IEEE Robot. Automat. Lett.*, vol. 4, no. 1, pp. 161–168, Jan. 2019.

[55] F. Dellaert and M. Kaess, "Factor graphs for robot perception," *Foundations Trends Robot.*, vol. 6, no. 1/2, pp. 1–139, 2017.

[56] J. Subirana, J. Zornoza, M. Hernández-Pajares, E. S. Agency, and K. Fletcher, *GNSS Data Processing (ESA TM)*. Noordwijk, The Netherlands: ESA Commun., 2013.

[57] M. Heikkinen, "Geschlossene Formeln zur Berechnung rauemlicher geodaetischer Koordinaten aus rechtwinklingen Koordinaten," *Zeitschrift fuer Vermessungswessen*, vol. 107, pp. 207–211, 1982.

[58] J. Zhu, "Conversion of earth-centered earth-fixed coordinates to geodetic coordinates," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 30, no. 3, pp. 957–961, Jul. 1994.

[59] "United States Department of Defense, World Geodetic System 1984: Its definition and relationships with local geodetic systems," Nat. Geospatial-Intell. Agency, Springfield, VA, USA, Tech. Rep. TR8350.2, Jan. 2000.

[60] T. D. Barfoot, *State Estimation for Robotics*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2024.

[61] S. Anderson and T. D. Barfoot, "Full STEAM ahead: Exactly sparse Gaussian process regression for batch continuous-time trajectory estimation on SE(3)," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 157–164.

[62] J. N. Wong, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "A data-driven motion prior for continuous-time trajectory estimation on SE(3)," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1429–1436, Apr. 2020.

[63] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 679–693, Jun. 2014.

[64] J. Dong, M. Mukadam, B. Boots, and F. Dellaert, "Sparse Gaussian processes on matrix lie groups: A unified framework for optimizing continuous-time trajectories," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 6497–6504.

[65] P. Teunissen and O. Montenbruck, *Springer Handbook of Global Navigation Satellite Systems*, 1st ed. Berlin, Germany: Springer, 2017.

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

[67] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.

[68] M. Agrawal, "A Lie algebraic approach for consistent pose registration for general Euclidean motion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 1891–1897.

[69] S. W. Anderson, "Batch continuous-time trajectory estimation," Ph.D. dissertation, Graduate Dept. Aerosp. Sci. Eng., Univ. of Toronto, Toronto, ON, Canada, Jun. 2017. [Online]. Available: https://hdl.handle.net/1807/77974

[70] P. J. Huber and E. M. Roncetti, *Robust Statistics*, 2nd ed. Hoboken, NJ, USA: Wiley, 2009.

[71] A. Ranganathan, M. Kaess, and F. Dellaert, "Fast 3D pose estimation with out-of-sequence measurements," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2007, pp. 2486–2493.

[72] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 3281–3288.

[73] D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, inertial, LiDAR, and leg odometry for all-terrain legged robots," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 309–326, Feb. 2023.

[74] D. J. Yoon et al., "Need for speed: Fast correspondence-free LiDAR odometry using Doppler velocity," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 5304–5310.

[75] J. N. Wong, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "Variational inference with parameter learning applied to vehicle trajectory estimation," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5291–5298, Oct. 2020.

Haoming Zhang (Member, IEEE) received the MASc. degree in mechatronics from the University of Duisburg-Essen, Duisburg, Germany, in 2017. He is currently working toward the Ph.D. degree in mechanical engineering with the Institute of Automatic Control, RWTH Aachen University, Aachen, Germany.

He works on research projects to enable robust vehicle/vessel localization in complex and large-scale environments. His research interests include state estimation, multisensor fusion, and learning-based online noise distribution estimation methods.

Chih-Chun Chen (Graduate Student Member, IEEE) received the BASc. degree in engineering science program (robotics option) from the University of Toronto, Toronto, ON, Canada, in 2019, and the MASc. degree in aerospace science and engineering from the University of Toronto Institute for Aerospace Studies, Toronto, in 2021. She is currently working toward the Ph.D. degree in mechanical engineering with the Institute of Automatic Control, RWTH Aachen University, Aachen, Germany.

Her research interests include multiagent path planning, state estimation, and multisensor fusion.

Heike Vallery (Member, IEEE) received the Dipl.-Ing. degree in mechanical engineering from RWTH Aachen University, Aachen, Germany, in 2004, and the Dr.-Ing. degree in engineering from the Technische Universität München, Munich, Germany, in 2009.

She was with ETH Zürich, Zürich, Switzerland; Khalifa University, Abu Dhabi, United Arab Emirates; and TU Delft, Delft, The Netherlands. She is currently a Full Professor with RWTH Aachen and TU Delft, and also holds an honorary professorship with Erasmus MC, Rotterdam, The Netherlands. Her research interests include design and control of minimalistic robotics.

Timothy D. Barfoot (Fellow, IEEE) received the B.A.Sc. degree in engineering science and the Ph.D. degree in aerospace science and engineering from the University of Toronto, Toronto, ON, Canada, in 1997 and 2002, respectively.

He is currently a Professor with the University of Toronto Robotics Institute, Toronto, ON, Canada. He works in the areas of guidance, navigation, and control of autonomous systems in a variety of applications. His research interests include developing methods to allow robotic systems to operate over long periods of time in large-scale, unstructured, 3-D environments, using rich onboard sensing (e.g., cameras and laser rangefinders) and computation.