Continuous-Time Ultra-Wideband-Inertial Fusion

Kailai Li, Ziyu Cao, and Uwe D. Hanebeck

***Abstract*—We introduce a novel framework of continuous-time ultra-wideband-inertial sensor fusion for online motion estimation. Quaternion-based cubic cumulative B-splines are exploited for parameterizing motion states continuously over time. Systematic derivations of analytic kinematic interpolations and spatial differentiations are further provided. Based thereon, a new sliding-window spline fitting scheme is established for asynchronous multi-sensor fusion and online calibration. We conduct a dedicated validation of the quaternion spline fitting method, and evaluate the proposed system, SFUISE (spline fusion-based ultra-wideband-inertial state estimation), in real-world scenarios using public data set and experiments. The proposed sensor fusion system is real-time capable and delivers superior performance over state-of-the-art discrete-time schemes. We release the source code and own experimental data at https://github.com/KIT-ISAS/SFUISE.**

***Index Terms*—Sensor fusion, localization.**

I. Introduction and Related Work

ONLINE estimation of dynamical motions is of fundamental importance in achieving reliable autonomy of mobile robots [1], [2], [3], [4]. Recent advancements in ultra-wideband (UWB) technology have offered promising alternative solutions to localization in GPS-denied environments. Compared with common sensing principles, e.g., cameras or LiDARs, UWB sensors are lightweight, low-cost, and more scalable in large-scale deployment, particularly, in indoor scenarios [5]. Despite the high spatial resolution of ultra-wideband impulse radio, which transmits at the nanosecond level, there are still technical challenges to achieving high-performance UWB-based tracking in practice. Non-line-of-sight (NLOS) and multipath conditions are well-known issues in UWB ranging, which are dependent on sensor placements and can be further exacerbated by complex and time-varying environments, such as those with moving obstacles [6], [7]. In addition, UWB ranging often exhibits non-Gaussian noise patterns, with surrounding-dependent interference and diffraction that are impossible to model parametrically [8], [9], [10]. Thus, basic UWB tracking solutions using multilateration are almost always insufficient for high-performance motion estimation.

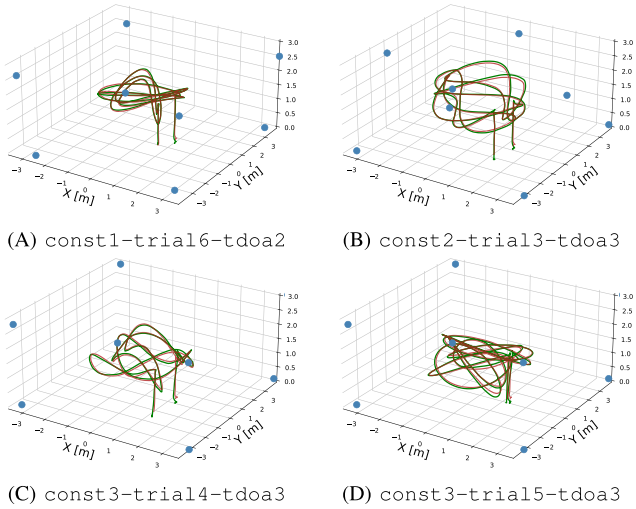


Fig. 1. Exemplary runs of SFUISE on `UTIL`. The proposed system delivers accurate trajectory estimates (green) compared to ground truth (red). Blue dots depict anchor positions.

The performance of UWB tracking can be improved through sensor fusion with other modalities. Inertial measurement units (IMUs) provide instantaneous and higher-order motion information that can bridge the gap between consecutive UWB readings. They are cost- and resource-efficient, and can be easily integrated into UWB sensor networks. Conventional UWB-inertial fusion methods often rely on recursive filtering algorithms, particularly the extended Kalman filter (EKF), with inertial measurements facilitating state propagation and ultra-wideband ranging updating the predicted prior. One basic application was introduced in [11], where an EKF was used to localize micro aerial vehicles with a UWB-inertial setup. Further practices involve utilizing six-DoF motion estimation through quaternion kinematics, and incorporating error-states to enhance overall system performance [8], [12], [13].

However, recursive filters rely on the Markov assumption, where evidence for predicting the current state is only traced back to the last state [4]. Sensor measurements of different modalities are usually fused in a decoupled manner, inducing substantial information loss in correlations across multi-sensor readings [2]. These issues can be substantially mitigated by fusing sensor measurements into one joint graph-based nonlinear

optimization. It improves tracking accuracy, while still maintaining tractable computational complexity thanks to its sparse structure [14]. Such a paradigm shift has been predominantly reflected in visual or LiDAR based odometry [1], [15]. For UWB-inertial state estimation, current techniques have not fully embraced state-of-the-art methodologies, with most use cases limited to batchwise (offline) or planar motion estimation [5], [16].

Conventional sensor fusion schemes are built atop discrete timestamps of constant interval, necessitating temporal alignment of measurements w.r.t. the estimation [15]. However, different sensors fire asynchronously without any common time instant. In a single-sensor setup, measurements can also be non-uniformly sampled over time due to timestamp jitter, particularly in UWB sensing [8]. Interpolating asynchronous range measurements from different anchors results in degraded tracking performance, especially under outliers and complex noise patterns. Thus, continuous-time state estimation is appealing. In this regard, data-driven approaches based on Gaussian processes have been systematically investigated, where various motion priors are learned within the stochastic differential equation formulations [17], [18].

B-splines parameterize trajectories atop *knots*, or *control points*, through temporal polynomials, enabling interpolation at any given time instant with locality and smoothness. This concept can be practically generalized to Lie groups or nonlinear manifolds via reformulation into a cumulative form, based on which rigid body motions can be modeled continuously over time. In [19], [20], B-splines were applied to estimating continuous-time states via batchwise maximum a posteriori. Similar offline schemes have also been proposed for attitude estimation, multi-sensor calibration, and trajectory estimation using visual/event/LiDAR-inertial setups [21], [22], [23], [24], [25]. However, these batchwise schemes are computationally expensive due to the rudimentary strategies of computing time derivatives (via product rule) and Jacobians on B-splines (via numerical or automatic differentiations). A breakthrough in efficient continuous-time motion estimation using B-splines was made in [26], where recursive computation of time derivatives and Jacobians on SO(3) trajectories was introduced for spline fitting. This has inspired successive work in online motion estimation, with applications including RGB-D tracking, LiDAR/visual-inertial calibration and odometry [27], [28], [29], [30], [31].

While significant progress has been made in continuous-time sensor fusion, there is still a considerable gap towards more extensive engineering practice. To the best knowledge of the authors, no continuous-time solution is currently available for UWB-inertial fusion. Unit quaternions are widely accepted in robotics as a nonsingular rotation representation and have certain favorable attributes in memory efficiency, numerical stability, and computation [2], [15], [32], [33]. However, current spline-based state estimation systems rely heavily on the theory in [26] using rotation matrices, while unit quaternions are only utilized for basic arithmetic in implementation [30]. Thus, there lacks an open-source quaternion-based B-spline sensor fusion framework, with analytic kinematic interpolations and spatial differentiations unified systematically.

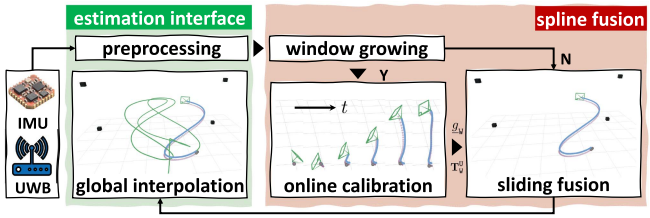


Fig. 2. System pipeline of SFUISE.

I. Contribution

We introduce SFUISE, a novel Spline Fusion-based Ultra-wideband-Inertial State Estimation scheme (Section II). Quaternion-based cubic cumulative B-splines serve as the backbone of state representation, with systematic and unified derivations of analytic kinematic interpolations and Jacobians (Section III). Based thereon, an efficient sliding-window spline fitting scheme is established to fuse UWB-inertial readings at raw timestamps with an added option of online calibration (Section IV). A dedicated study is first conducted to validate the viability of the quaternion spline fitting scheme. Afterward, we evaluate SFUISE for UWB-inertial tracking in various real-world scenarios using public data sets and experiments, including comparisons with state-of-the-art discrete-time fusion schemes (Section V). The proposed system delivers real-time and superior performance over the discrete-time counterparts. Considering the generality of the proposed scheme to extensive scenarios, we open-source our implementation together with own experimental data sets.

II. System Overview

In the considered scenario, we aim to estimate the following motion-related variables

$$\underline{x}(t) = [\,\underline{q}(t)^\top,\, \underline{p}(t)^\top,\, \underline{b}(t)^\top\,]^\top \in \mathbb{S}^3 \times \mathbb{R}^3 \times \mathbb{R}^6 \subset \mathbb{R}^{13} \quad (1)$$

as a function over time. Quaternion $\underline{q}(t) \in \mathbb{S}^3$ and vector $\underline{p}(t) \in \mathbb{R}^3$ denote orientations and positions, respectively, and $\underline{b}(t) = [\underline{b}_{\mathtt{acc}}^\top, \underline{b}_{\mathtt{gyro}}^\top]^\top \in \mathbb{R}^6$ incorporates biases of accelerometer and gyroscope. The proposed spline fusion-based ultra-wideband-inertial state estimation (SFUISE) system is depicted in Fig. 2. It composes an estimation interface and a functional core of spline fusion. Asynchronous UWB and IMU measurements are first preprocessed for potential downsampling, to which a cubic cumulative B-spline is fitted over a time window of limited span. As sensor data are streamed in, the spline fusion window first grows to a pre-given width, afterward slides, both w.r.t. the current timestamp for online state estimation. In the growing stage, additional online calibrations are performed to obtain the gravity vector $\underline{g}^{\mathtt{W}}$ and transformation $\mathbf{T}_{\mathtt{W}}^{\mathtt{U}}$ from world ($\mathtt{W}$) to UWB ($\mathtt{U}$) frames. Further, knot estimates given by spline fusion are sent back to estimation interface, where a global spline is maintained and interpolated for visualization.

III. State Estimation on Cumulative B-Splines

A. Continuous-Time State Modeling

We deploy cubic (fourth-order) cumulative B-splines for continuous-time state representation. Being established upon a set of control points $\{(\underline{\mathcal{p}}_i, t_i)\}_{i=1}^{n}$ associated with time t_i of uniform interval, it allows for fusing sensor readings up to the second order of motion (e.g., from accelerometer) [26]. State at an arbitrary timestamp $t \in [\, t_i, t_{i+1}\,)$ can be interpolated w.r.t. a local set of knots $\{\underline{\mathcal{p}}_{i+j-2}\}_{j=0}^{3}$ according to

$$\underline{p}(t) = \underline{\mathcal{p}}_{i-2} + \sum\nolimits_{j=1}^{3} \lambda_j(u)\, \underline{\delta}_j\,, \quad \text{with} \quad u = \frac{t - t_i}{t_{i+1} - t_i} \tag{2}$$

being the normalized time and $\underline{\delta}_j = \underline{\mathcal{p}}_{i+j-2} - \underline{\mathcal{p}}_{i+j-3}$ the distance between neighboring knots. Note that we use calligraphic fonts to denote spline knots, highlighting that they do not lie on the trajectory itself. The cumulative basis functions $\{\lambda_j(u)\}_{j=1}^{3}$ follow $[\,\lambda_1(u), \lambda_2(u), \lambda_3(u)\,]^\top = \Phi\, \underline{u}$, with

$$\Phi = \frac{1}{6}\begin{bmatrix} 5 & 3 & -3 & 1 \\ 1 & 3 & 3 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \underline{u} = [\,1, u, u^2, u^3\,]^\top$$

endowing cubic B-splines with $\mathcal{C}^2$-continuity. Expression (2) can be applied to model positions and IMU biases in (1). Based on Riemannian geometry, the concept of cumulative B-splines can be naturally extended to the manifold of unit quaternions [34]. It follows

$$\underline{q}(t) = \underline{\mathcal{q}}_{i-2} \otimes \prod\nolimits_{j=1}^{3} \mathrm{Exp}_{\mathbb{1}}\big(\lambda_j(u)\, \underline{\delta}_j\big)\,, \tag{3}$$

with $\otimes$ denoting the Hamilton product and $\lambda_j(u)$ the basis functions in (2). Distance between adjacent knots is quantified via logarithm map $\underline{\delta}_j = \mathrm{Log}_{\mathbb{1}}(\underline{\mathcal{q}}_{i+j-3}^{-1} \otimes \underline{\mathcal{q}}_{i+j-2})$ at identity $\mathbb{1} = [1, 0, 0, 0]^\top$ and contributes to the on-manifold interpolation via exponential map $\mathrm{Exp}_{\mathbb{1}}(\cdot)$ [35]. Unless otherwise specified, the term 'B-spline' in the following content refers to the cumulative formulation of uniform intervals.

On modeling motion-related states, knots are optimally estimated by fitting the spline to measurements in the least squares sense. Constructing residuals in the objective refers to kinematic interpolations on cubic B-splines w.r.t. related sensory modalities. This can be computationally expensive due to large data volume and complexity in deriving motion derivatives. Meanwhile, solving nonlinear least squares requires Jacobians of on-manifold kinematic interpolations w.r.t. knots. In the remainder of this section, we provide these theoretical building blocks for quaternion-based B-splines towards high-performance continuous-time sensor fusion.

B. Kinematic Interpolations

We now present time derivatives of cubic B-splines for interpolating linear and angular velocities, and acceleration.

1) Linear velocity and acceleration: Given a position B-spline $\underline{p}(t)$ in (2), its first-order time derivative (denoted by dot atop the variable) can be derived via $\dot{\underline{p}}(t) = \underline{p}'(u)u'(t)$, with $u'(t) = 1/(t_{i+1} - t_i) := 1/\Delta_i$. This leads to

$$\dot{\underline{p}}(t) = \sum\nolimits_{j=1}^{3} \lambda_j'(u)\, \underline{\delta}_j u'(t) = \sum\nolimits_{j=1}^{3} \dot{\lambda}_j(u)\, \underline{\delta}_j\,,$$

with $\dot{\lambda}_j$ being derivatives of basis functions in (2) given by $[\dot{\lambda}_1(u), \dot{\lambda}_2(u), \dot{\lambda}_3(u)]^\top = \Phi\,[0, 1, 2u, 3u^2\,]^\top/\Delta_i$. Further, the acceleration over time follows

$$\ddot{\underline{p}}(t) = \sum\nolimits_{j=1}^{3} \ddot{\lambda}_j(u)\, \underline{\delta}_j\,, \tag{4}$$

with $[\ddot{\lambda}_1(u), \ddot{\lambda}_2(u), \ddot{\lambda}_3(u)]^\top = \Phi\,[0, 0, 2, 6u\,]^\top/\Delta_i^2$.

2) Angular velocity: By definition [12], directly computing the time derivative of the quaternion spline function in (3) leads to the angular velocity $\underline{\omega}(t)$ in body frame via relation

$$\dot{\underline{q}}(t) = 0.5\, \underline{q}(t) \otimes \underline{\omega}(t)\,. \tag{5}$$

In practice, however, this requires quadratic complexity w.r.t. spline order due to the chain of Hamilton products. For cumulative B-splines on matrix Lie group SO(3), an efficient recursive method has been introduced in [26]. We hereby provide the full derivation for its quaternion counterpart.

For brevity, we discard the time variable in (3) and split the product chain into $\underline{q} = \underline{q}_k \otimes \prod_{j=k+1}^{3} \mathrm{Exp}_{\mathbb{1}}(\lambda_j\, \underline{\delta}_j)$, where $\underline{q}_k = \underline{\mathcal{q}}_{i-2} \otimes \prod_{j=1}^{k} \mathrm{Exp}_{\mathbb{1}}(\lambda_j\, \underline{\delta}_j)$ for $k \in \{1, 2, 3\}$. The quaternion spline interpolation can then be expressed in a recursive fashion according to

$$\underline{q}_k = \underline{q}_{k-1} \otimes \underline{e}_k\,, \tag{6}$$

with $\underline{e}_k := \mathrm{Exp}_{\mathbb{1}}(\lambda_k\, \underline{\delta}_k)$ denoting the k-th spline increment obtained from exponential map. Applying the kinematic relation in (5) to $\underline{q}_k$ yields

$$\dot{\underline{q}}_k = 0.5\, \underline{q}_k \otimes \underline{\omega}_k\,, \tag{7}$$

with $\underline{\omega}_k$ being the angular velocity to be computed recursively. For that, we compute the time derivative of (6) and obtain

$$\dot{\underline{q}}_k = \dot{\underline{q}}_{k-1} \otimes \underline{e}_k + \underline{q}_{k-1} \otimes \dot{\underline{e}}_k\,, \tag{8}$$

where time derivative of increment $\underline{e}_k$ is given by $\dot{\underline{e}}_k = \underline{e}_k \otimes (\dot{\lambda}_k \underline{\delta}_k)$ according to (5). Expression in (8) then follows

$$\begin{aligned} \dot{\underline{q}}_k &= 0.5\, \underline{q}_{k-1} \otimes \underline{\omega}_{k-1} \otimes \underline{e}_k + \underline{q}_{k-1} \otimes \underline{e}_k \otimes (\dot{\lambda}_k \underline{\delta}_k) \\ &= 0.5\, \underline{q}_k \otimes \big(\underline{e}_k^{-1} \otimes \underline{\omega}_{k-1} \otimes \underline{e}_k + 2\dot{\lambda}_k \underline{\delta}_k\big)\,, \end{aligned}$$

based on (6) and (7). Subsequently, the following recursive expression for angular velocity interpolation can be established

$$\underline{\omega}_k = \mathcal{R}(\underline{e}_k^{-1})\, \underline{\omega}_{k-1} + 2\dot{\lambda}_k \underline{\delta}_k\,, \tag{9}$$

where $\mathcal{R}(\underline{e}_k^{-1})\, \underline{\omega}_{k-1} = \underline{e}_k^{-1} \otimes \underline{\omega}_{k-1} \otimes \underline{e}_k$ rotates angular velocity $\underline{\omega}_{k-1}$ with the inverse quaternion increment $\underline{e}_k^{-1}$. To bootstrap the recursion, we can derive $\underline{\omega}_1$ via time derivative of $\underline{q}_1 = \underline{\mathcal{q}}_{i-2} \otimes \underline{e}_1$, namely, $\dot{\underline{q}}_1 = \underline{\mathcal{q}}_{i-2} \otimes \dot{\underline{e}}_1 = \underline{q}_1 \otimes (\dot{\lambda}_1 \underline{\delta}_1)$, leading to $\underline{\omega}_1 = 2\dot{\lambda}_1 \underline{\delta}_1$.

C. Spatial Differentiations

On solving the nonlinear least squares for spline fitting, the gradient of the objective can be computed via the chain rule, where the major complexity goes to the kinematic terms. For that, we consider a spline segment ranging over $t \in [\, t_i, t_{i+1})$ as

introduced in Section III-A. Jacobians of interpolated kinematic states $\underline{x}$ w.r.t. knots $\{\underline{\boldsymbol{x}}_{i+j-2}\}_{j=0}^{3}$ are in principle expressed as

$$\frac{d\underline{x}}{d\underline{\boldsymbol{x}}_{i+j-2}} = \mathbb{I}_{j=0}\mathbf{J}_{i-2} + \mathbb{I}_{j\neq 0}\mathbf{J}_{j} + \mathbb{I}_{j\neq 3}\mathbf{J}_{j+1}\,, \tag{10}$$

with $\mathbb{I}$ being the indicator function. Depending on the knot indices, Jacobian components in (10) are calculated via

$$\mathbf{J}_{i-2} = \frac{\partial \underline{x}}{\partial \underline{\boldsymbol{x}}_{i-2}}, \mathbf{J}_{j} = \frac{\partial \underline{x}}{\partial \underline{\delta}_{j}}\frac{\partial \underline{\delta}_{j}}{\partial \underline{\boldsymbol{x}}_{i+j-2}}, \mathbf{J}_{j+1} = \frac{\partial \underline{x}}{\partial \underline{\delta}_{j+1}}\frac{\partial \underline{\delta}_{j+1}}{\partial \underline{\boldsymbol{x}}_{i+j-2}}$$

that are concretized for the kinematic types as follows.

1) Jacobians of position and orientation interpolations: Following (10), it is trivial to obtain the Jacobian of the position spline w.r.t knot as $\mathit{d}\underline{p}/\ \mathit{d}\underline{\boldsymbol{p}}_{i+j-2} = (\mathbb{I}_{j=0} + \mathbb{I}_{j\neq 0}\lambda_j - \mathbb{I}_{j\neq 3}\lambda_{j+1})\,\mathbf{I}_3$. For the orientation spline defined in (3), its first Jacobian component in (10) is derived as $\mathbf{J}_{i-2} = \mathcal{Q}^{\lrcorner}(\prod_{j=1}^{3}\underline{e}_j)$, with function $\mathcal{Q}^{\lrcorner}$ mapping a quaternion multiplied from right-hand side into its matrix representation [35, 3.5]. To obtain the Jacobian components $\mathbf{J}_j$ and $\mathbf{J}_{j+1}$ in (10), we reformulate the quaternion interpolation (3) into

$$\underline{q} = \left(\underline{\boldsymbol{q}}_{i-2} \otimes \prod\nolimits_{k=1}^{j-1}\underline{e}_k\right) \otimes \underline{e}_j \otimes \prod\nolimits_{k=j+1}^{3}\underline{e}_k = \mathbf{Q}_{\leftarrow}^{\llcorner}\mathbf{Q}_{\rightarrow}^{\lrcorner}\,\underline{e}_j$$

to expose the j-th increment $\underline{e}_j$. $\mathbf{Q}_{\leftarrow}^{\llcorner}$ and $\mathbf{Q}_{\rightarrow}^{\lrcorner}$ are matrices representing the quaternions multiplied on the left- and right-hand sides of $\underline{e}_j$ via $\mathcal{Q}^{\llcorner}$ and $\mathcal{Q}^{\lrcorner}$, respectively, i.e.,

$$\mathbf{Q}_{\leftarrow}^{\llcorner} = \mathcal{Q}^{\llcorner}\left(\underline{\boldsymbol{q}}_{i-2} \otimes \prod\nolimits_{k=1}^{j-1}\underline{e}_k\right), \mathbf{Q}_{\rightarrow}^{\lrcorner} = \mathcal{Q}^{\lrcorner}\left(\prod\nolimits_{k=j+1}^{3}\underline{e}_k\right).$$

We then obtain the gradient of quaternion spline w.r.t. $\underline{\delta}_j$ as

$$\frac{\partial \underline{q}}{\partial \underline{\delta}_j} = \lambda_j\,\mathbf{Q}_{\leftarrow}^{\llcorner}\,\mathbf{Q}_{\rightarrow}^{\lrcorner}\,\frac{\partial \mathrm{Exp}_{\mathbb{1}}(\underline{v})}{\partial \underline{v}}\bigg|_{\underline{v}=\lambda_j\underline{\delta}_j}, \tag{11}$$

with the gradient of exponential map provided in (19). The gradient w.r.t. $\underline{\delta}_{j+1}$ takes the same expression as in (11), which can be computed recursively given the computation result for $\underline{\delta}_j$. The gradient of $\underline{\delta}_j$ w.r.t. knot follows

$$\frac{\partial \underline{\delta}_j}{\partial \underline{\boldsymbol{q}}_{i+j-2}} = \frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{\mathscr{d}}_j)}{\partial \underline{\boldsymbol{q}}_{i+j-2}} = \frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial \underline{q}}\bigg|_{\underline{q}=\underline{\mathscr{d}}_j} \mathcal{Q}^{\llcorner}(\underline{\boldsymbol{q}}_{i+j-3}^{-1}), \tag{12}$$

with $\underline{\mathscr{d}}_j := \underline{\boldsymbol{q}}_{i+j-3}^{-1} \otimes \underline{\boldsymbol{q}}_{i+j-2}$. We provide the gradient of logarithm map in (20). The gradient of $\underline{\delta}_{j+1}$ is derived as

$$\frac{\partial \underline{\delta}_{j+1}}{\partial \underline{\boldsymbol{q}}_{i+j-2}} = \frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial \underline{q}}\bigg|_{\underline{q}=\underline{\mathscr{d}}_{j+1}} \mathcal{Q}^{\lrcorner}(\underline{\boldsymbol{q}}_{i+j-1})\,\mathbf{D}, \tag{13}$$

where $\mathbf{D} = \mathrm{diag}(1, -1, -1, -1)$ denotes a diagonal matrix.

2) Jacobian of angular velocity interpolation: In reference to (10), the Jacobian takes the following general form

$$\frac{d\underline{\omega}}{d\underline{\boldsymbol{q}}_{i+j-2}} = \mathbb{I}_{j\neq 0}\boldsymbol{\Omega}_j + \mathbb{I}_{j\neq 3}\boldsymbol{\Omega}_{j+1}\,, \tag{14}$$

with the two partial derivatives expressed as follows

$$\boldsymbol{\Omega}_j = \frac{\partial \underline{\omega}}{\partial \underline{\omega}_j}\frac{\partial \underline{\omega}_j}{\partial \underline{\delta}_j}\frac{\partial \underline{\delta}_j}{\partial \underline{\boldsymbol{q}}_{i+j-2}}, \boldsymbol{\Omega}_{j+1} = \frac{\partial \underline{\omega}}{\partial \underline{\omega}_{j+1}}\frac{\partial \underline{\omega}_{j+1}}{\partial \underline{\delta}_{j+1}}\frac{\partial \underline{\delta}_{j+1}}{\partial \underline{\boldsymbol{q}}_{i+j-2}}.$$

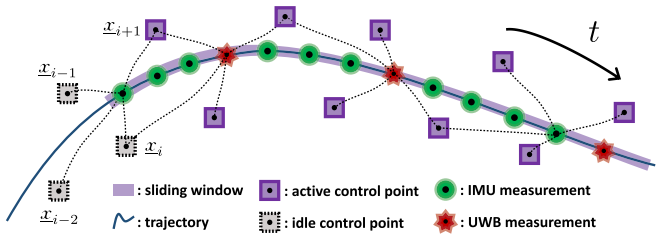


Fig. 3. Sliding-window spline fitting for online estimation.

The two components above are computed in the same fashion, and we now only demonstrate the derivation for $\boldsymbol{\Omega}_j$. The first term in $\boldsymbol{\Omega}_j$ can be obtained via the recursion in (9) according to the following derivation. For cubic cumulative B-spline, we have $\underline{\omega}(t) = \underline{\omega}_k(t)$ with $k = 3$. Thus, we obtain

$$\frac{\partial \underline{\omega}}{\partial \underline{\omega}_j} = \frac{\partial \underline{\omega}}{\partial \underline{\omega}_3}\prod\nolimits_{k=1}^{3-j}\frac{\partial \underline{\omega}_{4-k}}{\partial \underline{\omega}_{3-k}} = \prod\nolimits_{k=1}^{3-j}\mathcal{R}(\underline{e}_{4-k}^{-1})\,,$$

and the second term is derived as

$$\frac{\partial \underline{\omega}_j}{\partial \underline{\delta}_j} = \lambda_j\frac{\partial(\mathcal{R}(\underline{q})\,\underline{\omega}_{j-1})}{\partial \underline{q}}\bigg|_{\underline{q}=\underline{e}_j^{-1}}\mathbf{D}\frac{\partial \mathrm{Exp}_{\mathbb{1}}(\underline{v})}{\partial \underline{v}}\bigg|_{\underline{v}=\lambda_j\underline{\delta}_j} + 2\dot{\lambda}_j\mathbf{I}_3\,.$$

$\partial(\mathcal{R}(\underline{q})\underline{\omega}_{j-1})/\partial\underline{q}$ denotes the Jacobian of quaternion rotation that is given by in [12, Eq.19]. And the last term $\partial\underline{\delta}_j/\partial\underline{\boldsymbol{q}}_{i+j-2}$ is available in (12).

3) Jacobian of acceleration interpolation: We apply the general formulation (10) to the acceleration interpolation (4). It is then straightforward to obtain the Jacobian as follows

$$\frac{d\ddot{\underline{p}}}{d\underline{\boldsymbol{p}}_{i+j-2}} = (\mathbb{I}_{j\neq 0}\ddot{\lambda}_j - \mathbb{I}_{j\neq 3}\ddot{\lambda}_{j+1})\mathbf{I}_3\,.$$

IV. Online UWB-Inertial Spline Fusion

A. Sliding-Window Spline Fitting

Shown in Fig. 3, we exploit cubic B-splines to parameterize state (1) continuously over time with knots concatenated as $\underline{\boldsymbol{x}} = [\underline{\boldsymbol{q}}^{\top}, \underline{\boldsymbol{p}}^{\top}, \underline{\boldsymbol{\ell}}^{\top}]^{\top} \in \mathbb{R}^{13}$. To keep the computational intensity tractable for online performance, we bound the spline fitting problem over a window of recent τ_{w} knots $\mathcal{X}_{\mathrm{w}} = [\underline{\boldsymbol{x}}_1, \ldots, \underline{\boldsymbol{x}}_{\tau_{\mathrm{w}}}] \in \mathbb{R}^{13\times\tau_{\mathrm{w}}}$, namely,

$$\mathcal{X}_{\mathrm{w}}^{\star} = \underset{\mathcal{X}_{\mathrm{w}}}{\mathrm{argmin}}\,\mathscr{F}(\mathcal{X}_{\mathrm{w}})\,, \tag{15}$$

with the objective function formulated as $\mathscr{F}(\mathcal{X}_{\mathrm{w}}) =$

$$\upsilon_{\mathtt{u}}\sum_{i=1}^{m}\left\|\mathscr{E}_{\mathtt{u}}(\mathcal{X}_{\mathrm{w}}, \hat{z}_{\mathtt{u},i})\right\|_{\mathbf{C}_{\mathtt{u}}}^{2} + \upsilon_{\mathtt{i}}\sum_{k=1}^{n}\left\|\mathscr{E}_{\mathtt{i}}(\mathcal{X}_{\mathrm{w}}, \hat{z}_{\mathtt{i},k})\right\|_{\mathbf{C}_{\mathtt{i}}}^{2}. \tag{16}$$

$\mathscr{E}_{\mathtt{u}}$ and $\mathscr{E}_{\mathtt{i}}$ denote residual terms built upon UWB and IMU measurements, $\{\hat{z}_{\mathtt{u},i}\}_{i=1}^{m}$ and $\{\hat{z}_{\mathtt{i},k}\}_{k=1}^{n}$, respectively, each observed at raw timestamps. $\mathbf{C}_{\circ}$ and $\upsilon_{\circ}$ indicate the sensor noise covariances and tunable weights, respectively, for each error term (subscript '$\circ$' stands for '$\mathtt{u}$' or '$\mathtt{i}$' denoting UWB or IMU, respectively). As the window slides, the so-called *active* knots within the current window are updated over time until to be dropped out. Meanwhile, the three knots that are most

recently removed from the window are turned into an *idle* state. They are no longer being optimized, however, still participate in computing residuals in the current window via kinematic interpolations, which enables motion continuation across sliding windows over time [29]. Note that timestamps of residuals and underlying knots are not to be aligned. This allows for flexible multi-sensor fusion and efficient motion representation compared with discrete-time paradigm.

We now specify the residual terms in (16). Throughout the following derivations, we use B-splines to describe the 6-DoF motion of the IMU body ($\mathtt{I}$) w.r.t. the world frame ($\mathtt{W}$). Anchor positions are given w.r.t. UWB frame ($\mathtt{U}$) with a transformation $\mathbf{T}_{\mathtt{W}}^{\mathtt{U}} \in \mathrm{SE}(3)$ from the world frame.

1) UWB residual: We demonstrate the UWB residual in (16) for time-of-arrival (ToA) ranging. Given a range measurement $\hat{z}_{\mathtt{u},i}$ at timestamp t_i, its residual term follows

$$\mathscr{E}_{\mathtt{u}}(\mathcal{X}_{\mathrm{w}}, \hat{z}_{\mathtt{u},i}) = \left\| \mathbf{T}_{\mathtt{W}}^{\mathtt{U}} \underline{p}_{\mathtt{ta},i}^{\mathtt{W}} - \underline{\alpha}_{\mathtt{an},i}^{\mathtt{U}} \right\| - \hat{z}_{\mathtt{u},i}\,, \tag{17}$$

with $\underline{p}_{\mathtt{ta},i}^{\mathtt{W}} = \underline{q}(t_i) \otimes \underline{\nu}_{\mathtt{ta}}^{\mathtt{I}} \otimes \underline{q}^{-1}(t_i) + \underline{p}(t_i)$ being the UWB tag position in world frame obtained by transforming tag coordinates $\underline{\nu}_{\mathtt{ta}}^{\mathtt{I}}$ w.r.t. body pose interpolated at t_i. $\underline{\alpha}_{\mathtt{an},i}^{\mathtt{U}}$ denotes coordinates of the corresponding anchor. Residual for time-difference-of-arrival (TDoA) ranging can be obtained similarly, which we do not specify due to page limit.

2) IMU residual: Given the k-th IMU reading $\hat{\underline{z}}_{\mathtt{i},k}$ of acceleration $\hat{\underline{a}}_k^{\mathtt{I}}$ and angular velocity $\hat{\underline{\omega}}_k^{\mathtt{I}}$, we construct residual $\mathscr{E}_{\mathtt{i}}(\mathcal{X}_{\mathrm{w}}, \hat{\underline{z}}_{\mathtt{i},k}) = [\,\underline{\varkappa}_{\mathtt{acc},k}^{\top}, \underline{\varkappa}_{\mathtt{gyro},k}^{\top}, \underline{\varkappa}_{\mathtt{bias},k}^{\top}\,]^{\top}$ for spline fitting to inertial data. The accelerometer residual $\underline{\varkappa}_{\mathtt{acc},k}$ is

$$\underline{\varkappa}_{\mathtt{acc},k} = \underline{a}_k^{\mathtt{I}} + \underline{b}_{\mathtt{acc},k} - \hat{\underline{a}}_k^{\mathtt{I}}\,, \tag{18}$$

with $\underline{a}_k^{\mathtt{I}} = \underline{q}^{-1}(t_k) \otimes (\ddot{\underline{p}}(t_k) + \underline{g}^{\mathtt{W}}) \otimes \underline{q}(t_k)$ transforming the interpolated acceleration $\ddot{\underline{p}}(t_k)$ together with gravity $\underline{g}^{\mathtt{W}}$ to body frame. $\underline{b}_{\mathtt{acc},k}$ is the accelerometer bias interpolated on spline $\underline{b}(t)$ at t_k. The gyroscope residual $\underline{\varkappa}_{\mathtt{gyro},k}$ follows $\underline{\varkappa}_{\mathtt{gyro},k} = \underline{\omega}^{\mathtt{I}}(t_k) + \underline{b}_{\mathtt{gyro},k} - \hat{\underline{\omega}}_k^{\mathtt{I}}$, with $\underline{\omega}^{\mathtt{I}}(t_k)$ interpolated recursively as shown in (9). $\underline{b}_{\mathtt{gyro},k}$ denotes the gyroscope bias interpolated at t_k. Further, we compute the difference of consecutive bias interpolations at t_k and t_{k+1} as the IMU bias residual, namely, $\underline{\varkappa}_{\mathtt{bias},k} = \underline{b}(t_{k+1}) - \underline{b}(t_k)$.

B. Concurrent Calibration

In general, the transformation $\mathbf{T}_{\mathtt{W}}^{\mathtt{U}}$ (parameterized by a quaternion $\underline{q}_{\mathtt{W}}^{\mathtt{U}} \in \mathbb{S}^3$ and a translation vector $\underline{t}_{\mathtt{W}}^{\mathtt{U}} \in \mathbb{R}^3$) from world to UWB frames in (17) is unknown and typically dependent on the anchor coordinates and the tag pose at system initialization. Also, the gravity orientation $\bar{\underline{g}}^{\mathtt{W}}$ (obtained via normalizing the gravity $\underline{g}^{\mathtt{W}} = \|\underline{g}^{\mathtt{W}}\|\,\bar{\underline{g}}^{\mathtt{W}}$) in (18) is in general not available upon navigation. To approximate it, a common practice is to average the first several accelerometer readings under the assumption of a static motion at start. This can be easily violated by an undesirable starting condition (e.g., on the fly). To address these issues, we concatenate $(\underline{q}_{\mathtt{W}}^{\mathtt{U}}, \underline{t}_{\mathtt{W}}^{\mathtt{U}})$ and $\bar{\underline{g}}^{\mathtt{W}}$ after the knots in the state vector (15) during window-growing phase and estimate them via spline fitting.

C. Implementation

The proposed spline fusion-based UWB-inertial state estimation (SFUISE) scheme is developed in C++ using ROS [36]. The system composes two individual nodes corresponding to the two functional blocks in the system pipeline Fig. 2. Besides basic computational tools such as Eigen (https://eigen.tuxfamily.org), no further external dependency is required. We customize Levenberg-Marquardt (LM) algorithm to solve the nonlinear least squares in (15) iteratively. The closed-form gradients w.r.t. quaternion states are further established in the tangent space at estimate $\check{\underline{q}}$ w.r.t. a local perturbation $\underline{\phi} \in \mathbb{R}^3$ by multiplying with $\frac{\partial(\check{\underline{q}} \otimes \mathrm{Exp}_{\mathbb{I}}(\underline{\phi}))}{\partial \underline{\phi}}\big|_{\underline{\phi}=\underline{0}} = \mathcal{Q}^{\mathsf{L}}(\check{\underline{q}}) \frac{\partial \mathrm{Exp}_{\mathbb{I}}(\underline{\phi})}{\partial \underline{\phi}}\big|_{\underline{\phi}=\underline{0}}$. Solving the linearized system yields an increment $\underline{\phi}^*$, which updates the estimate through $\check{\underline{q}} \leftarrow \check{\underline{q}} \otimes \mathrm{Exp}_{\mathbb{I}}(\underline{\phi}^*)$ [2], [37]. Furthermore, the gravitation orientation $\bar{\underline{g}}^{\mathtt{W}} \in \mathbb{S}^2$ is handled in a similar fashion as introduced in [15].

V. Evaluation

In this section, we first provide a rigorous validation for the proposed quaternion spline fitting scheme. Afterward, we conduct in-depth benchmarks of SFUISE in diverse real-world scenarios. All evaluations are conducted using a laptop (Intel i7-12800H CPU, 32 GB RAM) running Ubuntu 20.04.

A. Validation of Quaternion Spline Fitting

The major theoretical complexity for the proposed spline fusion schemes lies in the part for orientation. Thus, we adopt the same synthesis as pres#ented in [26, Sec. 6.1] for batchwise continuous-time SO(3) estimation using both orientation and angular velocity measurements. We compare our quaternion-based scheme with the one using rotation matrices in [26] for fitting cubic cumulative B-splines. Both schemes are equipped with the same objective and stopping criteria. For each scheme, we exploit the Ceres Solver (http://ceres-solver.org) using LM algorithm with auto-differentiation for gradient computation, and the custom LM solvers using corresponding analytic gradients. In addition, we integrate our quaternion-based analytic gradient into Ceres for validation. We scale up the number of knots and measurements in the original sequence by a factor of $\{1, 5, 10\}$ and compute average runtime over 500 runs with knot initializations around identities perturbed by a small random noise. All methods have converged with an averaged RMSE of 2.21×10^{-4} w.r.t. the ground truth in terms of the SO(3) metric in [38, Eq. (19)]. As shown in Table I, the proposed quaternion spline fitting scheme achieves superior runtime efficiency, with gradients obtained from both auto-differentiation and analytic expressions. By utilizing closed-form gradients, our quaternion-based scheme runs slightly faster within Ceres than the custom LM in [26], which employs splines defined on rotation matrices.

B. Benchmarking Setup

In the upcoming sections, the proposed system is evaluated in real-world scenarios based on the public data set UTIL [7]

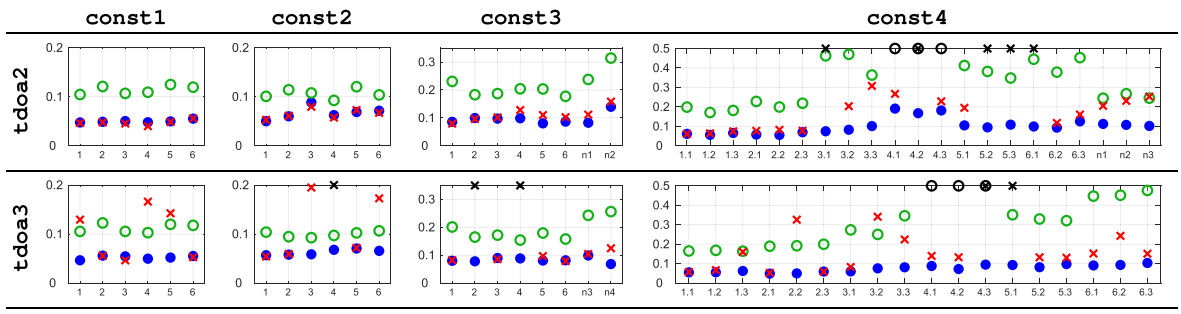


Fig. 4. APEs obtained from benchmark on all sequences of UTIL data set. The vertical axes denote RMSEs in meters. The horizontal axes denote trial# in const1 to const3 and trial#.traj# in const4, where n indicates a manual sequence. '#' denotes the sequence index. Results from SFUISE are plotted with ●. Results from ESKF and GUIF are given by ○ and ×, with ○ and × indicating tracking fails, respectively.

TABLE I
COMPARISONS OF DIFFERENT SCHEMES OF ORIENTATION SPLINE FITTING W.R.T. RUNTIME IN SECONDS

	Auto. Diff.*		Analytic		
Factor	[26]	Ours	[26]	Ours*	Ours
×1	0.0365	**0.0274**	0.0201	0.0184	**0.0137**
×5	0.2161	**0.1748**	0.1198	0.1157	**0.0833**
×10	0.4983	**0.4083**	0.2711	0.2660	**0.1850**

Methods with '*' exploit Ceres for optimization. Otherwise, the custom solvers are implemented using the same LM algorithm. The best runtime in each category is highlighted in bold.

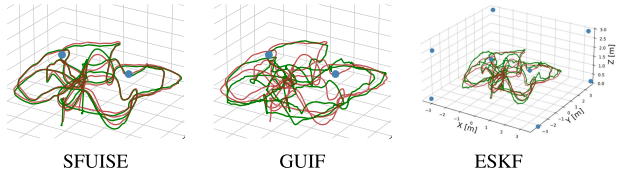


Fig. 5. A qualitative comparison of evaluated methods on const4-trial7-tdoa2-manual1. Ground truth and estimates are depicted in red and green, respectively.

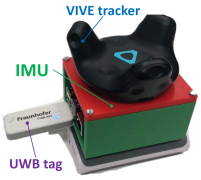
and own experiments, incorporating both ToA and TDoA ultra-wideband data. Two major discrete-time sensor fusion schemes from the state of the art are considered for comparison. An own composition of graph-based UWB-inertial fusion (GUIF) system is developed with reference to [39] and [1]. Here, discrete-time states are estimated via sliding window optimization with residuals of IMU preintegration, UWB ranging, and the prior factor from marginalization. Online calibration of $\mathbf{T}_{\mathrm{W}}^{\mathrm{U}}$ is enabled during window-growing stage. Further, we deploy the error-state Kalman filter (ESKF) provided by [7] with default calibration parameters for evaluation against the recursive estimation scheme. In order to achieve functional UWB ranging under signal interference, the three systems are equipped with a simple thresholding step to reject outliers in UWB measurements. Throughout the benchmark, SFUISE is configured with a sliding window of 100 knots at 10 Hz. All UWB readings are exploited for state estimation without downsampling. In order to devote the focus to investigating the core sensor fusion scheme, we avoid fine-tuning of system configuration parameters.

C. Public Data Set

We exploit UTIL flight data set for evaluating the proposed UWB-inertial fusion scheme with TDoA ultra-wideband ranging. Overall 79 sequences are collected onboard a quadcopter mounted with a UWB tag and an IMU (1000 Hz). The UWB sensor network is low-cost and operated at both centralized (tdoa2) and decentralized (tdoa3) modes under four different anchor constellations (const1-4). The TDoA measurements are collected with data rates ranging from 200 to 500 Hz including numerous challenging scenarios created by static and dynamic obstacles of various types of materials. Some sequences also exhibit an absence of IMU measurements. Given knot estimates from SFUISE, we obtain the global pose trajectory via interpolation at the frame rate of ground truth (200 Hz), w.r.t. which we compute the RMSE of the absolute position error (APE) to quantify the tracking accuracy.

As shown in Fig. 4, results given by the three systems are summarized w.r.t. UWB operation modes and anchor constellations. The proposed spline fusion scheme delivers superior performance over discrete-time approaches using recursive estimation or graph-based optimization. In particular, it exhibits a fairly good robustness against challenging conditions, e.g., in sequences with const4, where the tracking space is cluttered with static and dynamic obstacles of different materials including metal. In the face of time-varying NLOS and multi-path interference, discrete-time approaches are more susceptible to these effects, especially during online calibration, leading to large tracking errors or even complete failure. For demonstration, we plot results of a few representative runs of SFUISE in Fig. 1. Another qualitative comparison of spline fusion with discrete-time schemes is shown in Fig. 5 based on a representative sequence.

1) Runtime: As shown in Fig. 2, the two functional modules in the system pipeline run in parallel, and the spline fusion module dominates the computational cost compared with the lightweight estimation interface. Therefore, we record runtime of the backend fusion module w.r.t. the frame rate of knots at 10 Hz (thus 100 ms available for computation in real time). The proposed system delivers real-time performance with average runtime of 42.3 ± 1.9 ms and 37.2 ± 1.8 ms per sliding step throughout subsets tdoa2 and tdoa3, respectively. The small standard deviations indicate a well-bounded computational cost during sliding-window spline fusion. According to our investigation, solving the nonlinear least squares in (16)

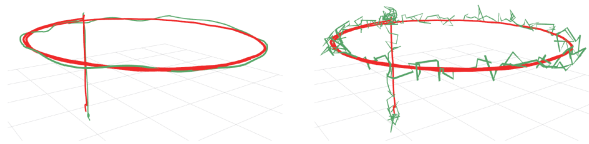


(A) sensor suite

Sequence	SFUISE	GUIF	ESKF
Walk1	0.117	0.143	0.225
Walk2	0.094	0.098	0.245
Walk3	0.094	0.229	0.270

(B) APE (RMSE) in meters.

Fig. 6. Miniature sensor suite for recording ISAS-Walk and corresponding APEs as shown in (a) and (b), respectively.



(A) spline-based (B) graph-based

Fig. 7. UWB-only tracking using SFUISE (a) and GUIF (b). Estimates (green) are depicted w.r.t. ground truth (red) at 200 Hz. The spline-based approach guarantees physically feasible estimates inherently.

using our custom LM method usually converges within five iterations.

D. Experiment

To further evaluate the proposed scheme on UWB-inertial fusion using ToA ranging, a miniature sensor suite has been instrumented as shown in Fig. 6(A). It is composed of a UWB tag provided by Fraunhofer IOSB-AST and an IMU embedded on Sense HAT (B), both mounted to a Raspberry Pi (https://www.waveshare.com) for sensor coordination and data recording. An additional VIVE tracker (https://www.vive.com) is added to provide the ground truth. Overall three sequences, ISAS-Walk1, ISAS-Walk2 and ISAS-Walk3, are recorded during indoor walks with inertial and ultra-wideband (including five anchors) readings both at a frame rate of about 80 Hz. We list the APE (RMSE) in meters given by the three systems in Fig. 6(B). The proposed scheme SFUISE produces the best tracking accuracy consistently throughout the data set.

E. Discussion

As verified in the mass evaluation based on UTIL and ISAS-Walk, SFUISE shows superior performance over discrete-time fusion schemes, particularly, in unfavorable scenarios of signal interference and complex noise patterns that are hard or infeasible to detect and model. Since any kinematic interpolation refers to four consecutive knots (for cubic B-splines), the induced pose trajectory inherently guarantees motion continuation up to the second order. In comparison with discrete-time fusions, this implicitly brings extra constraints to solving the nonlinear optimization problem in spline fitting, while still acknowledging the locality of observations. As a result, sensor fusion exhibits better stability and robustness under challenging conditions even together with online calibration. To further highlight the strength of spline fusion in motion estimation, we select const1-trial1-tdoa2 of UTIL, and discard the IMU readings for UWB-only navigation. We run SFUISE and GUIF online at estimation frame rates of 1 Hz and 10 Hz, respectively, with sliding windows both configured as 10 seconds to guarantee same amount of TDoA measurements. No explicit kinematic constraint, such as motion smoothness, is involved into state estimation. A qualitative comparison is demonstrated in Fig. 7. Due to signal noise and interference, graph-based approach produces physically-infeasible motion estimates. The proposed spline-based scheme delivers a smooth trajectory with an efficient state representation using knots estimated at 1 Hz (1/10 memory consumption of the discrete-time states delivered at 10 Hz by GUIF).

VI. Conclusion

In this work, we propose a new framework for continuous-time state estimation using ultra-wideband-inertial sensors. Quaternion-based cubic B-splines are exploited for six-DoF motion representation, based on which we systematically derive a unified set of theoretical tools for efficient spline fitting, including analytic kinematic interpolations and spatial differentiations on B-splines. This further facilitates the establishment of the novel sliding-window spline fusion scheme for online UWB-inertial state estimation. The resulted system, SFUISE, is evaluated in real-world scenarios based on public data set and experiments. It is real-time capable and shows superior performance over major discrete-time estimation approaches w.r.t. tracking accuracy, robustness and deployment flexibility. There still remains considerable potential to exploit the proposed scheme. B-splines of nonuniform knots can be further investigated to achieve more efficient state representation and estimation. Based on our theoretical contribution and open-source implementation, extensive engineering practice can be equipped with continuous-time paradigm in areas of automatic control, path planning, odometry and mapping, etc.

Appendix

A. Jacobian of Quaternion Exponential Map

For any point $\underline{v} \in \mathbb{R}^3$ in the tangent space at identity $\mathbb{1}$ on the manifold of unit quaternions, it can be retracted to $\mathbb{S}^3$ via exponential map according to

$$\mathrm{Exp}_{\mathbb{1}}(\underline{v}) = [\cos\|\underline{v}\|, \underline{v}^\top \operatorname{sinc}\|\underline{v}\|]^\top \in \mathbb{S}^3,$$

with $\|\cdot\|$ denoting the $\mathscr{L}_2$ norm [35]. The Jacobian of exponential map w.r.t. $\underline{v}$ then follows

$$\frac{\partial \mathrm{Exp}_{\mathbb{1}}(\underline{v})}{\partial \underline{v}} = \left[\left(\frac{\partial(\cos\|\underline{v}\|)}{\partial \underline{v}}\right)^\top, \left(\frac{\partial(\underline{v}\operatorname{sinc}\|\underline{v}\|)}{\partial \underline{v}}\right)^\top\right]^\top \in \mathbb{R}^{4\times 3}, \tag{19}$$

with the two items expressed as

$$\frac{\partial(\cos\|\underline{v}\|)}{\partial \underline{v}} = -\underline{v}^\top \operatorname{sinc}\|\underline{v}\| \quad \text{and}$$

$$\frac{\partial(\underline{v}\operatorname{sinc}\|\underline{v}\|)}{\partial \underline{v}} = \frac{\cos\|\underline{v}\| - \operatorname{sinc}\|\underline{v}\|}{\|\underline{v}\|^2}\underline{v}\,\underline{v}^\top + \mathbf{I}_3 \operatorname{sinc}\|\underline{v}\|.$$

B. Jacobian of Quaternion Logarithm Map

Given any unit quaternion $\underline{q} = [\, q_0, \underline{q}_\mathrm{v}^\top]^\top \in \mathbb{S}^3$ with q_0 and $\underline{q}_\mathrm{v}$ denoting the scalar and vector components, the rotation angle and axis can be retrieved by definition as $\theta = 2\arctan(\|\underline{q}_\mathrm{v}\|/q_0)$ and $\underline{u} = \underline{q}_\mathrm{v}/\|\underline{q}_\mathrm{v}\|$, respectively. It can be mapped to the tangent space at $\mathbb{1}$ via logarithm map

$$\mathrm{Log}_{\mathbb{1}}(\underline{q}) = \theta\underline{u}/2 = \arctan(\|\underline{q}_\mathrm{v}\|/q_0)\underline{q}_\mathrm{v}/\|\underline{q}_\mathrm{v}\| \in \mathbb{R}^3\,.$$

The Jacobian of logarithm map follows

$$\frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial \underline{q}} = \left[\frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial q_0}, \frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial \underline{q}_\mathrm{v}}\right] \in \mathbb{R}^{3\times 4}\,, \tag{20}$$

where $\dfrac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial q_0} = -\underline{q}_\mathrm{v}$ and

$$\frac{\partial \mathrm{Log}_{\mathbb{1}}(\underline{q})}{\partial \underline{q}_\mathrm{v}} = \frac{1}{\|\underline{q}_\mathrm{v}\|^2}\left(q_0\underline{q}_\mathrm{v}\underline{q}_\mathrm{v}^\top + \frac{\|\underline{q}_\mathrm{v}\|^2\mathbf{I}_3 - \underline{q}_\mathrm{v}\underline{q}_\mathrm{v}^\top}{\|\underline{q}_\mathrm{v}\|}\arctan\frac{\|\underline{q}_\mathrm{v}\|}{q_0}\right).$$

ACKNOWLEDGMENT

The authors would like to thank Norbert Fränzel from Fraunhofer IOSB-AST and Christopher Funk for help with instrumenting the sensor suite to record `ISAS-Walk` data set.

REFERENCES

[1] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-LiDAR-inertial odometry and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

[2] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015.

[3] H. Möls, K. Li, and U. D. Hanebeck, "Highly parallelizable plane extraction for organized point clouds using spherical convex hulls," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 7920–7926.

[4] M.-G. Li, H. Zhu, S.-Z. You, and C.-Q. Tang, "UWB-based localization system aided with inertial sensor for underground coal mine applications," *IEEE Sensors J.*, vol. 20, no. 12, pp. 6652–6669, Jun. 2020.

[5] S. Zheng, Z. Li, Y. Liu, H. Zhang, and X. Zou, "An optimization-based UWB-IMU fusion framework for UGV," *IEEE Sensors J.*, vol. 22, no. 5, pp. 4369–4377, Mar. 2022.

[6] W. Zhao, A. Goudar, and A. P. Schoellig, "Finding the right place: Sensor placement for UWB time difference of arrival localization in cluttered indoor environments," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6075–6082, Jul. 2022.

[7] W. Zhao, A. Goudar, X. Qiao, and A. P. Schoellig, "UTIL: An ultra-wideband time-difference-of-arrival indoor localization dataset," 2022, *arXiv:2203.14471*.

[8] J. D. Hol, F. Dijkstra, H. Luinge, and T. B. Schön, "Tightly coupled UWB/IMU pose estimation," in *Proc. IEEE Int. Conf. Ultra-wideband*, 2009, pp. 688–692.

[9] M. Kok, J. D. Hol, and T. B. Schön, "Indoor positioning using ultrawideband and inertial measurements," *IEEE Trans. Veh. Technol.*, vol. 64, no. 4, pp. 1293–1303, Apr. 2015.

[10] W. Zhao, J. Panerati, and A. P. Schoellig, "Learning-based bias correction for time difference of arrival ultra-wideband localization of resource-constrained mobile robots," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3639–3646, Apr. 2021.

[11] J. Li, Y. Bi, K. Li, K. Wang, F. Lin, and B. M. Chen, "Accurate 3D localization for MAV swarms by UWB and IMU fusion," in *Proc. IEEE Int. Conf. Control Automat.*, 2018, pp. 100–105.

[12] J. Sola, "Quaternion kinematics for the error-state Kalman filter," 2017, *arXiv:1711.02508*.

[13] A. Goudar and A. P. Schoellig, "Online spatio-temporal calibration of tightly-coupled ultrawideband-aided inertial localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 1161–1168.

[14] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2010, pp. 2657–2664.

[15] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[16] Y. Song and L.-T. Hsu, "Tightly coupled integrated navigation system via factor graph for UAV indoor localization," *Aerosp. Sci. Technol.*, vol. 108, 2021, Art. no. 106370.

[17] 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.

[18] T. D. Barfoot, *State Estimation for Robotics*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[19] 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.

[20] S. Anderson and T. D. Barfoot, "Towards relative continuous-time SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1033–1040.

[21] H. Sommer, J. R. Forbes, R. Siegwart, and P. Furgale, "Continuous-time estimation of attitude using B-splines on lie groups," *J. Guid., Control, Dyn.*, vol. 39, no. 2, pp. 242–261, 2016.

[22] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 2088–2095.

[23] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *Int. J. Comput. Vis.*, vol. 113, no. 3, pp. 208–219, 2015.

[24] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1425–1440, Dec. 2018.

[25] G. Cioffi, T. Cieslewski, and D. Scaramuzza, "Continuous-time vs. discrete-time vision-based SLAM: A comparative study," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 2399–2406, Apr. 2022.

[26] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative B-splines on lie groups," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11148–11156.

[27] A. J. Yang, C. Cui, I. A. Bârsan, R. Urtasun, and S. Wang, "Asynchronous multi-view SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5669–5676.

[28] J. Tirado and J. Civera, "Jacobian computation for cumulative b-splines on SE(3) and application to continuous-time object tracking," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7132–7139, Jul. 2022.

[29] J. Lv, K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo, "CLINS: Continuous-time trajectory estimation for LiDAR-inertial system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 6657–6663.

[30] 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.

[31] M. Persson, G. Häger, H. Ovrén, and P.-E. Forssén, "Practical pose trajectory splines with explicit regularization," in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 156–165.

[32] E. Fresk and G. Nikolakopoulos, "Full quaternion based attitude control for a quadrotor," in *Proc. Eur. Control Conf.*, 2013, pp. 3864–3869.

[33] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza, "A comparative study of nonlinear MPC and differential-flatness-based control for quadrotor agile flight," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3357–3373, Dec. 2022.

[34] M.-J. Kim, M.-S. Kim, and S. Y. Shin, "A general construction scheme for unit quaternion curves with simple high order derivatives," in *Proc. Annu. Conf. Comput. Graph. Interactive Techn.*, 1995, pp. 369–376.

[35] K. Li, "On-manifold recursive Bayesian estimation for directional domains," Ph.D. dissertation, Dept. Inform., Karlsruhe Inst. Technol., Karlsruhe, Germany, 2022.

[36] M. Quigley et al., "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, 2009, vol. 3, pp. 5–10.

[37] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intell. Transp. Syst. Mag.*, vol. 2, no. 4, pp. 31–43, Winter 2010.

[38] D. Q. Huynh, "Metrics for 3D rotations: Comparison and analysis," *J. Math. Imag. Vis.*, vol. 35, pp. 155–164, 2009.

[39] 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.