Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Continuous Integration over SO(3) for IMU Preintegration

Cedric Le Gentil
Centre for Autonomous Systems
University of Technology Sydney
Australia
Email: cedric.legentil@student.uts.edu.au

Teresa Vidal-Calleja
Centre for Autonomous Systems
University of Technology Sydney
Australia
Email: teresa.vidalcalleja@uts.edu.au

***Abstract*—This paper presents a novel method for continuous integration over the 3D rotation group SO(3). The key idea is to model the system's dynamics with Gaussian Processes (GPs) and learn the GP training data to match the system's instantaneous angular velocity measurements. This is formulated as the minimisation of a simple cost function that leverages the application of linear operators over GP kernels. The proposed integration method over SO(3) is applied to the preintegration of inertial data provided by a 6-DoF-Inertial Measurement Unit (IMU). Unlike standard integration that requires the recomputation of the integrals every time the estimate changes, preintegration combines the IMU readings into pseudo-measurements that are independent from the pose estimate and allows for efficient multi-modal sensor-fusion. The pseudo-measurements generated by the proposed method are named Unified Gaussian Preintegrated Measurements (UGPMs). UGPMs rely on GP regression and linear operators to analytically integrate the acceleration data. Moreover, a mechanism for IMU bias and time-shift correction is introduced to allow for seamless multi-modal state estimation. Over quantitative experiments, we show that the UGPMs outperform the current state-of-the-art preintegration methods.**

I. Introduction

A core component of any autonomous system is its ability to localise itself in the environment. Throughout the years, an abundance of localisation algorithms has been designed based on many different sensor modalities. The development of consumer electronics and especially smartphones made some sensors, like digital cameras and Inertial Measurement Units (IMUs), widely available and affordable. IMUs are lightweight proprioceptive sensors that collect data about a system's acceleration and angular velocities. For these reasons, IMUs are now ubiquitous in the robotics research field and literature [5, 6, 20, 21].

While inertial data do not provide direct measurements of the system's pose (rotation and position) in space, given initial pose and velocity conditions, one can integrate these data following the classical mechanics' equations to obtain information about the system's trajectory through time. However, this operation is not straightforward as rotations lie on a *manifold*, called the Special orthonormal group in three dimensions (SO(3)), and rotations are not commutative. It is common to compute a discrete numerical integration assuming piecewise constant angular velocities, which can be inaccurate. This paper aims to overcome this issue by introducing a novel continuous integration method based on an optimisation approach using Gaussian Process (GP) models of the inertial data.

In the context of pose estimation [12], the original pose of the system (part of the state variables) is not accurately known and changes over time during the estimation process. The concept of preintegration [15] dissociates the computation of the inertial data integrals from the initial conditions. The authors of [8, 9] extended the preintegration theory from [15] to perform computations on the SO(3) manifold instead of the original Euler angle representation. Overall, preintegration allows for the pre-processing of the IMU into pseudo-measurements, greatly improving the efficiency of inertial-aided multi-sensor localisation algorithms [11, 14, 16, 17].

More recently, different techniques have been proposed to improve the seminal works on preintegration. The method presented in [11] relies on the assumption of constant true acceleration to compute "closed-form" preintegrated measurements as per [7]. A modular method relying on the assumptions of constant angular velocities and accelerations has been introduced in [22]. The authors claim faster computation compared to [8] as well as marginal accuracy improvements. These methods use motion models to create preintegrated measurements. However, relying on arbitrary motion models while a system's motion is not constrained leads to inaccuracies. Our previous work in [13] alleviates the dependence on a motion model by employing GPs for continuous data-driven probabilistic representation of the IMU measurements. This continuous representation of the inertial measurements also accommodates asynchronous sensors. While this last method improves the accuracy of the preintegrated measurements by using analytical integration over GP-modelled signals, it does rely on the numerical integration of the gyroscope data in the general scenario. The proper treatment of the angular velocities is crucial to the overall accuracy as the integrated rotations directly impact the velocity and position preintegrated measurements. We aim in this paper to overcome the need for numerical integration for the rotations, addressing a gap in the preintegration literature.

Processing inertial constraints in a multi-sensor estimation framework by using continuous state representations has also been addressed in the literature. Generally, these

representations lead to an increased number of residuals in the optimisation cost function (one per IMU reading while preintegration combines many readings into a few pseudo-measurements), but this type of method can also accommodate asynchronous sensors. In [3], a linear interpolation mechanism between estimated poses is used to continuously characterise the system's trajectory. It corresponds to the assumption of piecewise constant velocity. The framework presented in [10] uses the linear combination of basis functions (B-Splines). The GP-based approaches in [1] and [2] perform a maximum a posteriori estimation of state variables and use GP regression to query the state at any timestamp. The authors embed a simple motion model in the GP kernel to make the method computationally efficient. The proposed approach in this paper combines concepts from [2] to perform continuous integration of the system's angular velocities to later produce accurate preintegrated measurements.

The main contribution of this paper is to address the general case of angular velocities integration over SO(3) using GPs for direct inference of the system's orientation. The proposed approach entirely inherits the continuous and probabilistic characteristics of GP regression, offering highly accurate performances without relying on any explicit motion model. Our novel method is used as part of an IMU preintegration technique that computes new measurements named Unified Gaussian Preintegrated Measurements (UGPMs). We also provide the derivations associated with a postintegration correction mechanism that accounts for the IMU biases and the inter-sensor time-shift. Finally, through quantitative evaluations, we show that the proposed approach outperforms the current state-of-the-art preintegration methods.

II. Problem statement and overview

A. Inertial system

Let us consider a 6-Degree-of-Freedom (DoF)-IMU made of a 3-axis gyroscope and a 3-axis accelerometer. Let us also denote $\mathbf{R}_W^{t_i}$, $\mathbf{p}_W^{t_i}$, and $\mathbf{v}_W^{t_i}$ as the IMU orientation (a rotation matrix $\in$ SO(3)), position, and velocity at time t_i, respectively. The subscript $_W$ indicates that the world fixed frame $\mathfrak{F}_W$ is used as a reference frame and the sensor's dynamics is governed by the following differential equations:

$$\dot{\mathbf{R}}_W^t(t) = \mathbf{R}_W^t(t)\boldsymbol{\omega}(t)^\wedge, \tag{1}$$

$$\dot{\mathbf{v}}_W^t(t) = \mathbf{f}_W(t), \tag{2}$$

$$\dot{\mathbf{p}}_W^t(t) = \mathbf{v}_W^t(t), \tag{3}$$

where $\dot{}$ is the differentiation operator with respect to time t, $\mathbf{f}_W$ the linear acceleration of the sensor in $\mathfrak{F}_W$, $\boldsymbol{\omega}$ the angular velocity of the inertial frame relative to $\mathfrak{F}_W$, and $^\wedge$ the operator that transforms a 3-by-1 vector into a skew-symmetric matrix as follows

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix}^\wedge = \begin{bmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{bmatrix}. \tag{4}$$

The IMU provides proper accelerations $\tilde{\mathbf{f}}(\mathrm{t}_i)$ and angular velocities $\tilde{\boldsymbol{\omega}}(\mathrm{t}_i)$ measurements at time t_i $(i = 1, \ldots, Q)$ in the inertial frame $\mathfrak{F}_I$. The link between $\{\tilde{\mathbf{f}}(\mathrm{t}_i), \tilde{\boldsymbol{\omega}}(\mathrm{t}_i)\}$ and $\{\mathbf{f}_W(\mathrm{t}_i), \boldsymbol{\omega}(\mathrm{t}_i)\}$ is as follows:

$$\begin{aligned}\tilde{\mathbf{f}}(t) &= \mathbf{R}_W^t(t)^\top(\mathbf{f}_W(t) - \mathbf{g}) + \mathbf{b}_f(t) + \eta_f(t),\\ \tilde{\boldsymbol{\omega}}(t) &= \boldsymbol{\omega}(t) + \mathbf{b}_\omega(t) + \eta_\omega(t),\end{aligned}$$

where $\mathbf{g}$ is the gravity vector in $\mathfrak{F}_W$, $\mathbf{b}_f$ and $\mathbf{b}_\omega$ slowly varying sensor biases, and η_f and η_ω the zero-mean Gaussian noises of variances σ_f and σ_ω for the linear accelerations and angular velocities respectively.

B. Preintegration

As in the seminal preintegration work [15], using the fact that $\mathbf{R}_W^t(t) = \mathbf{R}_W^{t_1}\mathbf{R}_{t_1}^t(t)$ and given known initial conditions at $t = t_1$, the IMU pose and velocity at $t_2 > t_1$ can be computed by integrating (1), (2), and (3):

$$\begin{aligned}\mathbf{R}_W^{t_2} &= \mathbf{R}_W^{t_1}\Delta\mathbf{R}_{t_1}^{t_2},\\ \mathbf{v}_W^{t_2} &= \mathbf{v}_W^{t_1} + \mathbf{g}\Delta t + \mathbf{R}_W^{t_1}\Delta\mathbf{v}_{t_1}^{t_2},\\ \mathbf{p}_W^{t_2} &= \mathbf{p}_W^{t_1} + \mathbf{v}_W^{t_1}\Delta t + \frac{\mathbf{g}\Delta t^2}{2} + \mathbf{R}_W^{t_1}\Delta\mathbf{p}_{t_1}^{t_2},\end{aligned}$$

with the preintegrated measurements $\Delta\mathbf{R}_{t_1}^{t_2}$, $\Delta\mathbf{v}_{t_1}^{t_2}$, and $\Delta\mathbf{p}_{t_1}^{t_2}$ defined as

$$\Delta\mathbf{R}_{t_1}^{t_2} = \prod_{t_1}^{t_2}\exp\left((\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))^\wedge\right)^{dt}, \tag{5}$$

$$\Delta\mathbf{v}_{t_1}^{t_2} = \int_{t_1}^{t_2}\mathbf{R}_{t_1}^t(t)\big(\tilde{\mathbf{f}}(t) - \mathbf{b}_f(t)\big)dt, \tag{6}$$

$$\Delta\mathbf{p}_{t_1}^{t_2} = \int_{t_1}^{t_2}\int_{t_1}^{t}\mathbf{R}_{t_1}^s(s)\big(\tilde{\mathbf{f}}(s) - \mathbf{b}_f(s)\big)dsdt. \tag{7}$$

Both relevant preintegration works, the original [15] and the more recent on manifold [8], integrate (5), (6), and (7) numerically, directly using the discrete IMU measurements. The rectangle rule for integration is employed in these cases. It corresponds to the hypothesis of constant inertial measurements between two consecutive measurement timestamps. This can be intuitively seen as assuming piecewise constant accelerations and angular velocities.

Our previous work in [13] attempts to solve (1), (2), and (3) analytically using GPs as continuous models of the inertial measurements. Unfortunately, (1) does not have any known general analytical solution [4]. While an analytical method for the 1-axis-rotation scenario was proposed, they still rely on the iterative numerical integration of (5) over upsampled gyroscope readings to solve for the general case (as done in [14]). Overall, in real-world situations, the solution for the 1-axis-rotation is not practical due to the presence of biases in the gyroscope data breaking the property of rotation commutativity needed to analytically integrate the angular velocities. Hence, to leverage the 1-axis-rotation analytical solution in real-world scenarios, one needs to provide the method with either the perfect knowledge of the rotation axis or the exact values of the gyroscope biases.

This work addresses the issue of continuous integration over SO(3) in the general case. It allows for the derivation

of motion-model-free preintegrated measurements. Alleviating motion assumptions leads to the reduction of the integration noise and greater accuracy of the measurements.

C. Method overview

In the proposed method, the integration is first solved for the rotation part $\Delta\mathbf{R}_{t_1}^{t_2}$ of the UGPMs, before the projection of the accelerometer measurements in $\mathfrak{F}_I$ at time t_1 and the direct inference of $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$. Fig. 1 shows the block diagram of the proposed approach. This is similar to the pipeline proposed in [13] but with a "unified" treatment of the rotational integration.

The novel SO(3) integration method relies on the simple concept of finding the optimal (learning) training data for GP inference. The optimisation is formulated following the non-linear rules that govern the system's dynamics over SO(3). Further, from the optimised state variables (the learned training data), the system's orientation can be queried via standard GP inference with linear operators.

The derivations of the UGPMs in Sections IV and V assume that the IMU biases are known at the time of integration. As this is not true in real-world scenarios, Section VI presents a postintegration mechanism to correct the UGPMs according to the updated knowledge of the biases. Note that the rotational preintegrated measurements $\Delta\mathbf{R}_{t_1}^{t}(t)$ are equal to the rotation matrices $\mathbf{R}_{t_1}^{t}(t)$ under the assumption of noiseless inertial readings and known gyroscope biases.

III. Preliminaries

This section provides the reader with succinct background knowledge about the SO(3) mathematical tools, GP regression [18], and the application of linear operators to the covariance kernels [19].

A. SO(3) mathematical tools

Rotations in SO(3) can be represented with rotation matrices $\mathbf{R} \in \mathbb{R}^{3\times3}$ or rotation vectors $\mathbf{r} \in \mathbb{R}^3$. A rotation vector can be expressed as a rotation matrix with the exponential mapping

$$\mathbf{R} = \exp(\mathbf{r}^\wedge) = \mathbf{I} + \frac{\sin(\|\mathbf{r}\|)}{\|\mathbf{r}\|}\mathbf{r}^\wedge + \frac{1-\cos(\|\mathbf{r}\|)}{\|\mathbf{r}\|^2}(\mathbf{r}^\wedge)^2.$$

The other way around, a rotation matrix can be expressed as a rotation vector thanks to the logarithmic mapping

$$\mathbf{r} = \log(\mathbf{R})^\vee = \frac{\phi(\mathbf{R}-\mathbf{R}^\top)}{2\sin(\phi)} \text{ with } \phi = \cos^{-1}\Big(\frac{\mathrm{tr}(\mathbf{R})-1}{2}\Big),$$

and $\bullet^\vee$ the inverse operation of $\bullet^\wedge$ defined in (4).

The right Jacobian of SO(3), formally defined as

$$\mathbf{J}_r(\mathbf{r}) = \lim_{\partial\mathbf{r}\to0} \frac{\log\big(\exp\big(\mathbf{r}^\wedge\big)^\top \exp\big((\mathbf{r}+\partial\mathbf{r})^\wedge\big)\big)^\vee}{\partial\mathbf{r}},$$

corresponds intuitively to the mapping of a variation of the rotation vector $\dot{\mathbf{r}}$ into a variation on the tangent space of the manifold. It can be expressed in a closed-form solution as

$$\mathbf{J}_r(\mathbf{r}) = \mathbf{I} - \frac{1-\cos(\|\mathbf{r}\|)}{\|\mathbf{r}\|^2}\mathbf{r}^\wedge + \frac{\|\mathbf{r}\|-\sin(\|\mathbf{r}\|)}{\|\mathbf{r}\|^3}(\mathbf{r}^\wedge)^2. \quad (8)$$

B. Gaussian process regression

GP regression is a non-parametric (data-driven) probabilistic interpolation method that uses covariance functions to characterise a signal. If we consider a signal $h(x)$ of input space $x \in \mathbb{R}$ represented with a zero-mean GP,

$$h(x) \sim \mathcal{GP}\big(0, k_h(x,x')\big), \quad (9)$$

the function $k_h(x,x')$, also called the kernel, describes the covariance between instances of $h(x)$:

$$\mathrm{cov}\big(h(x), h(x')\big) = k_h(x,x').$$

Given noisy observations y_i of $h(x)$,

$$y_i = h(x_i) + \eta \quad \text{with} \quad \eta \sim \mathcal{N}(0, \sigma_y^2), \quad (10)$$

and $i = 1, \cdots, N$, the prediction $h^*(x)$ at any input x is obtained as

$$\begin{aligned} h^*(x) =& \mathbf{k}_h(x,\mathbf{x})\big[\mathbf{K}_h(\mathbf{x},\mathbf{x}) + \sigma_y^2\mathbf{I}\big]^{-1}\mathbf{y} \quad (11)\\ \mathrm{var}\big(h^*(x)\big) =& k_h(x,x) \\ & - \mathbf{k}_h(x,\mathbf{x})\big[\mathbf{K}_h(\mathbf{x},\mathbf{x}) + \sigma_y^2\mathbf{I}\big]^{-1}\mathbf{k}_h(\mathbf{x},x), \end{aligned}$$

where $\mathbf{y} = \begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix}^\top$, $\mathbf{x} = \begin{bmatrix} x_1 & \cdots & x_N \end{bmatrix}^\top$, $\mathbf{k}_h(x,\mathbf{x}) = \begin{bmatrix} k_h(x,x_1) & \cdots & k_h(x,x_N) \end{bmatrix}$, $\mathbf{k}_h(\mathbf{x},x) = \mathbf{k}_h(x,\mathbf{x})^\top$, and

$$\mathbf{K}_h(\mathbf{x},\mathbf{x}) = \begin{bmatrix} k_h(x_1,x_1) & k_h(x_1,x_2) & \cdots & k_h(x_1,x_N) \\ k_h(x_2,x_1) & k_h(x_2,x_2) & \cdots & k_h(x_2,x_N) \\ \vdots & \vdots & \ddots & \vdots \\ k_h(x_N,x_1) & k_h(x_N,x_2) & \cdots & k_h(x_N,x_N) \end{bmatrix}.$$

C. Linear operators and Gaussian process regression

Given the same GP-modelled signal $h(x)$ (9) and noisy observations y_i (10), it is possible to infer analytically any linear operation of the signal. Applying a linear operator $\mathcal{L}_g^x$ to the signal $h(x)$ is written as

$$g(x) = \mathcal{L}_g^x h(x).$$

Note that for linear operators, $\mathcal{L}_g^x h(x)$ does not correspond to the multiplication of a matrix or vector $\mathcal{L}_g^x$ with $h(x)$, but instead it is the application of the operator $\mathcal{L}_g^x$ on $h(x)$. As an example, the differentiation operator $\frac{\partial}{\partial x}$ applied to $h(x)$ can be written as $g(x) = \frac{\partial h(x)}{\partial x} = \mathcal{L}_g^x h(x)$. While this notation can be confusing at first, it becomes handy when defining the variance of the inferred values $g^*(x)$. The superscript $^\bullet$ of a linear operator $\mathcal{L}_\star^\bullet$ represents on which variable the operator is applied.

Thus, the signal $g(x)$ can be probabilistically inferred for any value of the input variable x as

$$\begin{aligned} g^*(x) =& \mathcal{L}_g^x\mathbf{k}_h(x,\mathbf{x})\big[\mathbf{K}_h(\mathbf{x},\mathbf{x}) + \sigma_y^2\mathbf{I}\big]^{-1}\mathbf{y} \quad (12)\\ \mathrm{var}\big(g^*(x)\big) =& \mathcal{L}_g^x k_h(x,x)\mathcal{L}_g^x \\ & - \mathcal{L}_g^x\mathbf{k}_h(x,\mathbf{x})\big[\mathbf{K}_h(\mathbf{x},\mathbf{x}) + \sigma_y^2\mathbf{I}\big]^{-1}\mathbf{k}_h(\mathbf{x},x)\mathcal{L}_g^x. \quad (13) \end{aligned}$$

Note that the right product of the linear operator implies its application to the second argument of the preceding kernel function. This is emphasised by the superscript of the linear operator.

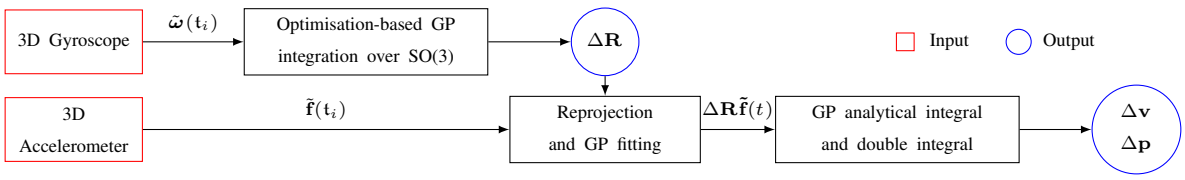

Fig. 1. Overview of the proposed preintegration method using Gaussian Processes (GP). It outputs pseudo-measurements called the Unified Gaussian Preintegrated Measurements.

IV. SO(3) Gaussian process integration

Let us consider the IMU orientation in $\mathfrak{F}_I$ at t_1 in its rotation vector form $\mathbf{r}^t_{t_1}(t) = \log(\mathbf{R}^t_{t_1}(t))^\vee$. The time derivative of $\mathbf{r}^t_{t_1}(t)$ is denoted $\dot{\mathbf{r}}^t_{t_1}(t)$ and represents a variation of the orientation in the rotation vector form. The angular velocities measured by the gyroscope correspond to the variation of the IMU orientation in the tangent space of the manifold. Note that using the right Jacobian of SO(3) (8), it is possible to map $\dot{\mathbf{r}}^t_W(t)$ to the tangent space of the manifold and obtain the following equality

$$\mathbf{J}_r(\mathbf{r}^t_{t_1}(t))\dot{\mathbf{r}}^t_{t_1}(t) = \boldsymbol{\omega}(t). \tag{14}$$

In simple words, the goal of our proposed GP integration over SO(3) is to learn/optimise the values of $\dot{\mathbf{r}}^{\mathfrak{t}_i}_{t_1}$ according to (14) and the given gyroscope readings $\tilde{\boldsymbol{\omega}}_{\mathfrak{t}_i}$ $(i = 1 \cdots Q)$. The key idea is that the estimates of the IMU orientation's time derivative, noted $\hat{\dot{\mathbf{r}}}^{\mathfrak{t}_i}_{t_1}$, allow for the direct inference of the IMU orientation ${\mathbf{r}^t_{t_1}}^*$ thanks to the use of GP regression and linear operators.

A. GP models

Let us model $\dot{\mathbf{r}}^t_{t_1}(t)$ as three independent GPs

$$(\dot{\mathbf{r}}^t_{t_1})_j \sim \mathcal{GP}\big(0, k_{r_j}(t,t')\big),$$

where $(\bullet)_j$ corresponds to the j^{th} component of $\bullet$. Then, it is possible to infer ${\dot{\mathbf{r}}^t_{t_1}}^*$ and ${\mathbf{r}^t_{t_1}}^*$ using (11) and (12):

$$(\dot{\mathbf{r}}^t_{t_1})^*_j = \mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t}) + \sigma_r^2\mathbf{I}\big]^{-1}\boldsymbol{\rho}_j \tag{15}$$

$$(\mathbf{r}^t_{t_1})^*_j = \mathcal{L}^t_r\mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t}) + \sigma_r^2\mathbf{I}\big]^{-1}\boldsymbol{\rho}_j \tag{16}$$

with $\mathfrak{t}$ being the vector of IMU timestamps $\mathfrak{t}_i$ $(i = 1 \cdots Q)$, $\boldsymbol{\rho}_j$ the vector of $(\hat{\dot{\mathbf{r}}}^{\mathfrak{t}_i}_{t_1})_j$, and $\mathcal{L}^t_r$ the integration operator defined as $\mathcal{L}^t_r h(t) = \int_{t_1}^t h(x)\partial x$.

B. Cost function

Let us formulate the optimisation problem using GP-inferred orientations and derivatives, and the gyroscope measurements to minimise the cost function C

$$\hat{\mathcal{S}} = \underset{\mathcal{S}}{\operatorname{argmin}}\ C(\mathcal{S}),\ \text{with } C(\mathcal{S}) = \sum_{i=1}^{Q}\Big(\|\mathbf{e}^i_I\|^2_{\Omega_{\boldsymbol{\omega}_i}} + \|\mathbf{e}^i_{gp}\|^2_{\Omega_{r_i}}\Big) \tag{17}$$

and find the optimal state $\hat{\mathcal{S}} = \begin{bmatrix}\boldsymbol{\rho}_1 & \boldsymbol{\rho}_2 & \boldsymbol{\rho}_3\end{bmatrix}$ (equal to stacking $\hat{\dot{\mathbf{r}}}^{\mathfrak{t}_1\top}_{t_1} \cdots \hat{\dot{\mathbf{r}}}^{\mathfrak{t}_Q\top}_{t_1}$).

The first residual corresponds to the algebraic manipulation of (14):

$$\mathbf{e}^i_I = \mathbf{J}_r({\mathbf{r}^{\mathfrak{t}_i}_{t_1}}^*){\dot{\mathbf{r}}^{\mathfrak{t}_i}_{t_1}}^* - \tilde{\boldsymbol{\omega}}_{\mathfrak{t}_i},$$

and $\Omega^{-1}_{\boldsymbol{\omega}_i}$ in (17) corresponds to the gyroscope's measurement covariance matrix.

GP inference in (15) can be seen as a dot product $\mathbf{c}^\top\boldsymbol{\rho}_j$ with $\mathbf{c}$ computed from the covariance vector and matrices, and $\boldsymbol{\rho}_j$ part of the estimated state. As this operation is not bijective, more than one ρ_j can lead to the same inference values. Therefore, additional residuals to properly constrain the estimated state are required:

$$\mathbf{e}^i_{gp} = {\dot{\mathbf{r}}^{\mathfrak{t}_i}_{t_1}}^* - \hat{\dot{\mathbf{r}}}^{\mathfrak{t}_i}_{t_1},$$

with $\Omega^{-1}_{r_i}$ in (17) corresponding to the diagonal matrix formed with the GP-inferred variances. Intuitively, this imposes the state variables to equal the inference at the same timestamps.

The optimisation problem (17) is solved using the Levenberg-Maquardt algorithm. Once the state has converged, (16) is used to infer the rotational part of the UGPM $\Delta\mathbf{R}^t_{t_1}$ at any timestamp t. The variance of $\Delta\mathbf{R}^t_{t_1}$ is computed as per (13).

V. Velocity and position preintegration

While $\Delta\mathbf{v}^t_{t_1}(t)$ and $\Delta\mathbf{p}^t_{t_1}(t)$ cannot be expressed as linear operations of the independent inertial signals (angular velocities and proper accelerations), one can project the accelerometer measurements in $\mathfrak{F}_I$ at time t_1 using the rotational preintegrated measurements $\Delta\mathbf{R}^t_{t_1}(t)$ inferred at $\mathfrak{t}_i$ $(i = 1, \ldots, Q)$ as per described in Section IV. This follows the formulation introduced in [13]. Therefore, (6) and (7) can be expressed as linear operations of the GP signals

$$\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_j \sim \mathcal{GP}\big(0, k_{f_j}(t,t')\big),$$

with $\bar{\mathbf{b}}_f$ being the prior knowledge of the accelerometer biases at the time of integration.

Rewriting (6) and (7) as

$$\Delta\mathbf{v}^{t_2}_{t_1} = \begin{bmatrix}(\Delta\mathbf{v}^{t_2}_{t_1})_1\\(\Delta\mathbf{v}^{t_2}_{t_1})_2\\(\Delta\mathbf{v}^{t_2}_{t_1})_3\end{bmatrix} = \begin{bmatrix}\mathcal{L}^t_v\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_1\\\mathcal{L}^t_v\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_2\\\mathcal{L}^t_v\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_3\end{bmatrix} \text{ and}$$

$$\Delta\mathbf{p}^{t_2}_{t_1} = \begin{bmatrix}(\Delta\mathbf{p}^{t_2}_{t_1})_1\\(\Delta\mathbf{p}^{t_2}_{t_1})_2\\(\Delta\mathbf{p}^{t_2}_{t_1})_3\end{bmatrix} = \begin{bmatrix}\mathcal{L}^t_p\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_1\\\mathcal{L}^t_p\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_2\\\mathcal{L}^t_p\big(\Delta\mathbf{R}^t_{t_1}(t)(\tilde{\mathbf{f}}(t) - \bar{\mathbf{b}}_f(t))\big)_3\end{bmatrix},$$

each component of $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ is inferred independently using (12):

$$\begin{aligned}(\Delta\mathbf{v}_{t_1}^t)_j{}^* &= \mathcal{L}_v^t \mathbf{k}_{f_j}(t,\mathfrak{t})\big[\mathbf{K}_{f_j}(\mathfrak{t},\mathfrak{t})+\sigma_{f_j}^2\mathbf{I}\big]^{-1}\mathbf{f}_j, \quad (18)\\ \text{var}((\Delta\mathbf{v}_{t_1}^t)_j{}^*) &= \mathcal{L}_v^t k_{f_j}(t,t)\mathcal{L}_v^t \\ &\quad - \mathcal{L}_v^t \mathbf{k}_{f_j}(t,\mathfrak{t})\big[\mathbf{K}_{f_j}(\mathfrak{t},\mathfrak{t})+\sigma_{f_j}^2\mathbf{I}\big]^{-1}\mathbf{k}_{f_j}(\mathfrak{t},t)\mathcal{L}_v^t,\end{aligned}$$

and

$$\begin{aligned}(\Delta\mathbf{p}_{t_1}^t)_j{}^* &= \mathcal{L}_p^t \mathbf{k}_{f_j}(t,\mathfrak{t})\big[\mathbf{K}_{f_j}(\mathfrak{t},\mathfrak{t})+\sigma_{f_j}^2\mathbf{I}\big]^{-1}\mathbf{f}_j, \quad (19)\\ \text{var}((\Delta\mathbf{p}_{t_1}^t)_j{}^*) &= \mathcal{L}_p^t k_{f_j}(t,t)\mathcal{L}_p^t \\ &\quad - \mathcal{L}_p^t \mathbf{k}_{f_j}(t,\mathfrak{t})\big[\mathbf{K}_{f_j}(\mathfrak{t},\mathfrak{t})+\sigma_{f_j}^2\mathbf{I}\big]^{-1}\mathbf{k}_{f_j}(\mathfrak{t},t)\mathcal{L}_p^t.\end{aligned}$$

The vector $\mathbf{f}_j$ consists of the set of training data $\big(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\bar{\mathbf{b}}_f(t))\big)_j$ at $t=\mathfrak{t}_i$ $(i=1\cdots Q)$.

VI. Postintegration bias and inter-sensor time-shift corrections

To correct the preintegrated measurements according to new estimates of the IMU biases, a first-order expansion mechanism has been presented in [15] under the assumption of constant biases all along the integration window. In [13], we extended this concept to inter-sensor time-shift correction. Here, we are adapting the formulation introduced in [13] to our novel SO(3) integration method.

In that regard, the preintegrated measurements (5), (6), and (7) are expanded as

$$\begin{aligned}\Delta\mathbf{R}_{t_1}^{t_2}(\mathbf{b}_\omega,\delta_t) \approx &\Delta\mathbf{R}_{t_1}^{t_2}(\bar{\mathbf{b}}_\omega,\bar{\delta}_t) \\ &\exp\left(\left(\frac{\partial\Delta\mathbf{R}_{t_1}^{t_2}}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}_\omega+\frac{\partial\Delta\mathbf{R}_{t_1}^{t_2}}{\partial\delta_t}\hat{\delta}_t\right)^\wedge\right), \quad (20)\\ \Delta\mathbf{v}_{t_1}^{t_2}(\mathbf{b}_f,\mathbf{b}_\omega,\delta_t) \approx &\Delta\mathbf{v}_{t_1}^{t_2}(\bar{\mathbf{b}}_f,\bar{\mathbf{b}}_\omega,\bar{\delta}_t)+\frac{\partial\Delta\mathbf{v}_{t_1}^{t_2}}{\partial\mathbf{b}_f}\hat{\mathbf{b}}_f \\ &+\frac{\partial\Delta\mathbf{v}_{t_1}^{t_2}}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}_\omega+\frac{\partial\Delta\mathbf{v}_{t_1}^{t_2}}{\partial\delta_t}\hat{\delta}_t, \quad (21)\\ \Delta\mathbf{p}_{t_1}^{t_2}(\mathbf{b}_f,\mathbf{b}_\omega,\delta_t) \approx &\Delta\mathbf{p}_{t_1}^{t_2}(\bar{\mathbf{b}}_f,\bar{\mathbf{b}}_\omega,\bar{\delta}_t)+\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\mathbf{b}_f}\hat{\mathbf{b}}_f \\ &+\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}_\omega+\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\delta_t}\hat{\delta}_t, \quad (22)\end{aligned}$$

with $\mathbf{b}_f=\bar{\mathbf{b}}_f+\hat{\mathbf{b}}_f$, $\mathbf{b}_\omega=\bar{\mathbf{b}}_\omega+\hat{\mathbf{b}}_\omega$, and $\delta_t=\bar{\delta}_t+\hat{\delta}_t$ ($\bar{\bullet}$ is the prior knowledge at the time of preintegration, and $\hat{\bullet}$ is the correction estimate). The different Jacobians involved in (20), (21), and (22) are detailed in the rest of this section.

A. Rotation Jacobians

1) Gyroscope biases: When inferring the components of the UGPM's rotational part $(\Delta\mathbf{r}_{t_1}^t)_j$ (16), one can see that the inferred values depend on the gyroscope biases solely through the training data $\boldsymbol{\rho}_j$. Consequently,

$$\frac{\partial(\Delta\mathbf{r}_{t_1}^t)_j}{\partial\mathbf{b}_\omega}=\mathcal{L}_r^t\mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t})+\sigma_r^2\mathbf{I}\big]^{-1}\frac{\partial\boldsymbol{\rho}_j}{\partial\mathbf{b}_\omega}. \quad (23)$$

The Jacobians $\frac{\partial\boldsymbol{\rho}_j}{\partial\mathbf{b}_\omega}$ are computed numerically from $\hat{\mathcal{S}}$: once the optimisation (17) converged, one component of the angular velocity measurements $\tilde{\boldsymbol{\omega}}_\mathfrak{t}$ is perturbed with a small increment before performing one step of the Levenberg-Maquardt algorithm. Repeating this operation three times (one for each of the bias components), the difference between the new estimates $\hat{\mathcal{S}}_{\mathbf{b}_{\omega_j}}$ and $\hat{\mathcal{S}}$ lead to the numerical differentiation of $\frac{\partial\boldsymbol{\rho}_j}{\partial\mathbf{b}_\omega}$ $(j=1\cdots3)$. The Jacobians computed with (23) corresponds to variations of the rotation vector. To match the Jacobians in (20), these need to be mapped to the tangent space of the manifold as $\frac{\partial\Delta\mathbf{R}_{t_1}^t}{\partial\mathbf{b}_\omega}=\mathbf{J}_r(\mathbf{r}_{t_1}^t{}^*)\frac{\partial\Delta\mathbf{r}_{t_1}^t}{\partial\mathbf{b}_\omega}$

2) Inter-sensor time-shift: The use of another linear operator allows the analytical derivation of $\frac{\partial\Delta\mathbf{r}_{t_1}^t}{\partial\delta_t}$:

$$\frac{\partial\Delta\mathbf{r}_{t_1}^{t_2}}{\partial\delta_t}=\frac{\partial}{\partial t_1}\int_{t_1}^{t_1+\Delta t}\dot{\mathbf{r}}_{t_1}^t(t)dt=\begin{bmatrix}\mathcal{L}_{r_{\delta_t}}^t(\dot{\mathbf{r}}_{t_1}^t(t))_1\\ \mathcal{L}_{r_{\delta_t}}^t(\dot{\mathbf{r}}_{t_1}^t(t))_2\\ \mathcal{L}_{r_{\delta_t}}^t(\dot{\mathbf{r}}_{t_1}^t(t))_3\end{bmatrix},$$

with $\Delta t=t_2-t_1$. This is inferred following (16) using $\mathcal{L}_{r_{\delta_t}}^t$ instead of $\mathcal{L}_r^t$. As in the previous paragraph, the rotation vector variation needs to be mapped to the tangent space of the manifold with $\frac{\partial\Delta\mathbf{R}_{t_1}^t}{\partial\delta_t}=\mathbf{J}_r(\mathbf{r}_{t_1}^t{}^*)\frac{\partial\Delta\mathbf{r}_{t_1}^t}{\partial\delta_t}$.

B. Velocity and position Jacobians

1) Accelerometer and gyroscope biases: Looking at (18) and (19), one can see that $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ depends on $\mathbf{b}_f$ and only through the training data $\mathbf{f}_{Wj}$. Consequently, the Jacobians with respect to the accelerometer and gyroscope biases in (21) and (22) can be inferred as

$$\frac{\partial(\Delta\mathbf{v}_{t_1}^t)_j}{\partial b_\bullet}=\mathcal{L}_v^t\mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t})+\sigma_{\omega_j}^2\mathbf{I}\big]^{-1}\frac{\partial\mathbf{f}_{Wj}}{\partial b_\bullet} \quad (24)$$

$$\frac{\partial(\Delta\mathbf{p}_{t_1}^t)_j}{\partial b_\bullet}=\mathcal{L}_p^t\mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t})+\sigma_{\omega_j}^2\mathbf{I}\big]^{-1}\frac{\partial\mathbf{f}_{Wj}}{\partial b_\bullet} \quad (25)$$

where $b_\bullet$ is either $\mathbf{b}_\omega$ or $\mathbf{b}_f$, the gyroscope and accelerometer biases, respectively. The training data $\mathbf{f}_{Wj}$ depends on $\mathbf{b}_\omega$ due to the dependence of $\Delta\mathbf{R}_{t_1}^t(t)$ on $\mathbf{b}_\omega$. The Appendix provides the expression of $\frac{\partial\mathbf{f}_{Wj}}{\partial\mathbf{b}_f}$ and $\frac{\partial\mathbf{f}_{Wj}}{\partial\mathbf{b}_\omega}$.

2) Inter-sensor time-shift: According to (18) and (19), $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ depends on the inter-sensor time-shift δ_t both through $\mathcal{L}_v^t\mathbf{k}_{f_j}(t,\mathfrak{t})$, $\mathcal{L}_v^t\mathbf{k}_{f_j}(t,\mathfrak{t})$, and $\mathbf{f}_{Wj}$ (as per the projection with $\Delta\mathbf{R}_{t_1}^{\mathfrak{t}_i}$). Consequently,

$$\begin{aligned}\frac{\partial(\Delta\mathbf{v}_{t_1}^t)_j}{\partial\delta_t}=&\mathcal{L}_{v_{\delta_t}}^t\mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t})+\sigma_{\omega_j}^2\mathbf{I}\big]^{-1}\mathbf{f}_{Wj} \\ &+\mathcal{L}_v^t\mathbf{k}_{r_j}(t,\mathfrak{t})\big[\mathbf{K}_{r_j}(\mathfrak{t},\mathfrak{t})+\sigma_{\omega_j}^2\mathbf{I}\big]^{-1}\frac{\partial\mathbf{f}_{Wj}}{\partial\delta_t}, \quad (26)\end{aligned}$$

with $\mathcal{L}_{v_{\delta_t}}^t=\frac{\partial}{\partial t_1}\mathcal{L}_v^t=\mathcal{L}_d^{t_1}\mathcal{L}_v^t$. The computation $\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\delta_t}$ follows (26) using $\mathcal{L}_{p_{\delta_t}}^t=\frac{\partial}{\partial t_1}\mathcal{L}_p^t=\mathcal{L}_d^{t_1}\mathcal{L}_p^t$ instead of $\mathcal{L}_{v_{\delta_t}}$. One can obtain $\frac{\partial\mathbf{f}_{Wj}}{\partial\delta_t}$ numerically or deducing it from $\frac{\partial(\Delta\mathbf{r}_{t_1}^t)_j}{\partial\delta_t}$.

VII. Experiments

In this section, we present quantitative evaluations of the UGPMs compared with state-of-the-art IMU preintegration approaches. All the quantitative experiments shown in this

section have been computed over 100-trial Monte-Carlo simulations. We also provide the reader with an example of the seamless integration of the UGPMs in a lidar-inertial localisation and mapping framework to validate the postintegration bias and time-shift corrections and the overall soundness of the UGPMs.

A. Implementation

To perform the estimation of the state $\hat{\mathcal{S}}$, we provide the minimisation problem (17) with a prior based on a variant of the Gaussian Preintegrated Measurements (GPMs) using linear interpolation between IMU measurements instead of GP regression. We name this computationally efficient variant Linear Preintegrated Measurement (LPM).

Due to the matrix inversion in (11), GP regression suffers from a cubic $\mathcal{O}(n^3)$ computational complexity. To mitigate this issue, we propose to divide a given integration window $[t_1, t_2]$ into a succession of smaller intervals, e.g. $[t_1, t_a]$ and $[t_a, t_2]$ with $t_1 < t_a < t_2$. The resulting preintegrated measurements are combined as $\Delta\mathbf{R}_{t_1}^{t_2} = \Delta\mathbf{R}_{t_1}^{t_a}\Delta\mathbf{R}_{t_a}^{t_2}$, $\Delta\mathbf{v}_{t_1}^{t_2} = \Delta\mathbf{v}_{t_1}^{t_a} + \Delta\mathbf{R}_{t_1}^{t_a}\Delta\mathbf{v}_{t_a}^{t_2}$, and $\Delta\mathbf{p}_{t_1}^{t_2} = \Delta\mathbf{p}_{t_1}^{t_a} + (t_2 - t_a)\Delta\mathbf{v}_{t_1}^{t_a} + \Delta\mathbf{R}_{t_1}^{t_a}\Delta\mathbf{p}_{t_a}^{t_2}$. This "divide-and-conquer" strategy will be referred as *UGPM (chunk)* and will be compared to the direct computation of the UGPMs over the full window $[t_1, t_2]$.

Our UGPM implementation[1] is based on the square exponential kernel parameterised by a length scale and the signal's variance (the so-called hyperparameters) [18]. For the velocity and position part of the UGPMs, it is possible to learn these hyperparameters directly from the data. For the rotational part, we use a simple arbitrary heuristic as the current stage of the UGPMs cannot learn these hyperparameters from the data. The length scale is set as five times the period of the IMU readings, and the variance is empirically computed from the state $\mathcal{S}$ prior. Learning the hyperparameters for the rotations is part of our future work, as we believe it can further improve the accuracy of the UGPMs.

The computations of the UGPMs and GPMs between t_1 and t_2 leverage the IMU measurements collected in a time window equal to $[t_1 - o,\ t_2 + o]$, with o being a temporal overlap of $150\,\mathrm{ms}$. All the methods tested have been implemented in C++ using *Ceres*[2] to solve the optimisation problem (17) of the UGPMs.

B. Simulation

1) Accuracy: With this set-up we aim at evaluating the accuracy of the proposed UGPMs against different techniques: the standard Preintegrated Measurements (PMs) [8], the GPMs [13], and the LPMs (a variant of the GPMs using linear interpolation instead of GP regression, equivalent to constant jerk and constant angular acceleration motion models). This panel of methods represents the range of methods present in the literature with the PMs using a restrictive motion model (constant accelerations and angular velocities at IMU frequency), the LPMs based on a very permissive model (constant jerk and angular acceleration at high frequency), and the GPMs as a motion-model-free approach. The UGPMs and GPMs use a type of covariance kernel parameterised with a few hyperparameters that can be either learnt from the data at hand or empirically set. In this experiment, we evaluate both methods with and without the hyperparameter training. However, as mentioned above for the rotational part of the UGPMs, the hyperparameters are only heuristically set. The arbitrary priors detailed in Subsection VII-A are used in this case.

The preintegration methods are benchmarked over integration windows from $50\,\mathrm{ms}$ to $1\,\mathrm{s}$ of length. These durations can typically be found in Visual-Inertial Odometry (VIO) frameworks to constrain the system's motion estimate between consecutive keyframes. The simulated IMU provides data at $100\,\mathrm{Hz}$ with realistic noise of $0.04\,\mathrm{m/s^2}$ and $0.01\,\mathrm{rad/s}$. Two motion types are used: *Slow* and *Fast*. Respectively, they correspond to average velocities of $9.7\,\mathrm{m/s}$ and $32.2\,\mathrm{m/s}$, and average angular velocities of $3.4\,\mathrm{rad/s}$ and $19.4\,\mathrm{rad/s}$. These are constructed based on sinusoidal functions of frequencies between $0.05Hz$ to $1Hz$ for the rotation and $0.05Hz$ to $1.5Hz$ for the velocity. The results are shown in Table I.

For slow trajectories, the LPMs, GPMs and UGPMs perform quite similarly, with a slight advantage for the UGPMs on longer integration windows. It is interesting to observe that the GPM hyperparameter learning is needed to outperform the simple LPMs in this scenario. We believe that the inertial data from these slow trajectories can be well approximated with high-frequency piecewise linear models and that the representativity of the GPs offers only a marginal improvement.

In the case of fast motion, the UGPMs significantly outperform any other technique. The 4-fold gap between the UGPMs and GPMs shows how important the integration of the angular velocities is for the overall accuracy (rotation and position) as both methods share the same pipeline for the generation of the position measurements. Additionally, it seems that learning the hyperparameters for the position part of the UGPMs does not improve significantly the method's accuracy. Also, it is worth noting that the per-chunk UGPM variant (based on intervals of maximum $0.2s$) performs similarly to the standard UGPMs while being computationally much more efficient as it will be shown in Subsection VII-B4.

2) Robustness to noise: This set-up analyses the impact that the sensor noise has on the preintegrated measurements generated with three different methods. The graph in Fig. 2 shows the independent variations of the accelerometer and gyroscope noises (if variating the accelerometer noise, the gyroscope noise is null, and vice-versa). *Fast* and *slow* trajectories over $1.0\,\mathrm{s}$ are used in the experiment. Here both the UGPMs and GPMs use learnt hyperparameters when applicable.

While the plots confirm the global superiority of the UGPMs in terms of accuracy, it also shows that the proposed method reduces the integration noise to negligible values. Indeed, on both plots, the errors of the UGPMs (rotation and position) converge towards zero as the sensor noise standard deviation goes towards zero. In the case of the LPMs and GPMs, the

[1] https://github.com/UTS-CAS/ugpm.

[2] Non-linear optimisation library http://ceres-solver.org/.

Slow motion - Average absolute and relative rotation and position errors

Δt [ms] $(t_2 - t_1)$	Error type	PM [8]	LPM	GPM [13]	GPM [13](train)	UGPM	UGPM (train)	UGPM (chunk)
0.05	Rot abs. er. [mrad]	4.18 ± 2.88	0.372 ± 0.149	0.322 ± 0.143	**0.255** ± 0.119	0.337 ± 0.142	0.337 ± 0.142	0.337 ± 0.14
	Rot rel. er.	9.63%	1%	0.849%	0.677%	0.911%	0.911%	0.91%
	Pos abs. er. [mm]	2.25 ± 1.29	0.037 ± 0.016	0.035 ± 0.014	**0.020** ± 0.011	0.035 ± 0.014	**0.020** ± 0.011	0.035 ± 0.014
	Pos rel. er.	2.48%	0.0403%	0.0384%	0.0214%	0.0384%	0.0215%	0.0384%
0.1	Rot abs. er. [mrad]	5.08 ± 3.19	0.521 ± 0.224	0.49 ± 0.204	**0.428** ± 0.16	0.463 ± 0.199	0.463 ± 0.199	0.462 ± 0.199
	Rot rel. er.	5.66%	0.641%	0.599%	0.528%	0.564%	0.564%	0.563%
	Pos abs. er. [mm]	5.12 ± 2.76	0.108 ± 0.046	0.108 ± 0.047	**0.073** ± 0.033	0.108 ± 0.047	**0.073** ± 0.033	0.108 ± 0.047
	Pos rel. er.	2.64%	0.0524%	0.0522%	0.0351%	0.0523%	0.035%	0.0523%
0.5	Rot abs. er. [mrad]	6.54 ± 3.79	1.39 ± 0.504	1.73 ± 0.796	1.38 ± 0.508	1.17 ± 0.48	1.17 ± 0.48	**1.16** ± 0.484
	Rot rel. er.	1.69%	0.405%	0.468%	0.397%	0.344%	0.344%	0.342%
	Pos abs. er. [mm]	24.9 ± 14.6	1.38 ± 0.56	1.79 ± 0.925	**1.33** ± 0.543	1.41 ± 0.554	1.34 ± 0.548	1.39 ± 0.552
	Pos rel. er.	2.7%	0.138%	0.176%	0.133%	0.141%	0.133%	0.138%
1	Rot abs. er. [mrad]	9.5 ± 5.14	1.76 ± 0.819	3.65 ± 2.27	1.77 ± 0.827	**1.47** ± 0.638	**1.47** ± 0.638	**1.47** ± 0.634
	Rot rel. er.	1.18%	0.226%	0.44%	0.227%	0.19%	0.19%	0.189%
	Pos abs. er. [mm]	53.5 ± 28.2	4.75 ± 1.72	12.2 ± 9.22	4.84 ± 1.76	4.64 ± 1.75	**4.41** ± 1.78	4.49 ± 1.72
	Pos rel. er.	2.74%	0.249%	0.633%	0.252%	0.241%	0.227%	0.233%

Fast motion - Average absolute and relative rotation and position errors

Δt [ms] $(t_2 - t_1)$	Error type	PM [8]	LPM	GPM [13]	GPM [13](train)	UGPM	UGPM (train)	UGPM (chunk)
0.05	Rot abs. er. [mrad]	25.2 ± 15.1	1.33 ± 0.49	1.34 ± 0.486	1.31 ± 0.491	**0.326** ± 0.142	**0.326** ± 0.142	0.327 ± 0.142
	Rot rel. er.	10.6%	0.621%	0.626%	0.608%	0.157%	0.157%	0.158%
	Pos abs. er. [mm]	27.2 ± 14.4	0.82 ± 0.409	0.063 ± 0.028	0.057 ± 0.028	0.039 ± 0.016	**0.034** ± 0.015	0.039 ± 0.016
	Pos rel. er.	7.2%	0.215%	0.017%	0.0153%	0.0105%	0.00897%	0.0105%
0.1	Rot abs. er. [mrad]	28.9 ± 16	2.73 ± 1.06	2.75 ± 1.05	2.72 ± 1.05	**0.483** ± 0.194	**0.483** ± 0.194	**0.483** ± 0.194
	Rot rel. er.	6.32%	0.634%	0.641%	0.635%	0.112%	0.112%	0.111%
	Pos abs. er. [mm]	63.7 ± 31	2.02 ± 0.781	0.374 ± 0.223	0.37 ± 0.222	0.128 ± 0.060	**0.127** ± 0.061	0.128 ± 0.060
	Pos rel. er.	8.41%	0.277%	0.0501%	0.0497%	0.0175%	0.0174%	0.0176%
0.5	Rot abs. er. [mrad]	66.7 ± 25	8.19 ± 2.6	8.21 ± 2.6	8.2 ± 2.61	**1.18** ± 0.487	**1.18** ± 0.487	**1.18** ± 0.485
	Rot rel. er.	2.82%	0.345%	0.346%	0.346%	0.0503%	0.0503%	0.0504%
	Pos abs. er. [mm]	300 ± 111	19.5 ± 9.61	14.6 ± 8.2	14.6 ± 8.16	**2.89** ± 1.52	**2.89** ± 1.5	**2.89** ± 1.52
	Pos rel. er.	7.47%	0.489%	0.36%	0.359%	0.0716%	0.0718%	0.0716%
1	Rot abs. er. [mrad]	49.3 ± 22.3	5.79 ± 2.27	5.73 ± 2.24	5.74 ± 2.23	**1.59** ± 0.759	**1.59** ± 0.759	**1.59** ± 0.756
	Rot rel. er.	1.07%	0.124%	0.123%	0.124%	0.0338%	0.0338%	0.0337%
	Pos abs. er. [mm]	574 ± 211	50.9 ± 22.8	42.6 ± 20.2	42.4 ± 20.3	**8.7** ± 3.62	8.72 ± 3.65	**8.7** ± 3.64
	Pos rel. er.	7.28%	0.642%	0.532%	0.531%	0.11%	0.11%	0.11%

TABLE I

AVERAGE ABSOLUTE AND RELATIVE ERROR OF PREINTEGRATED MEASUREMENTS FOR DIFFERENT TRAJECTORIES IN SIMULATED ENVIRONMENTS (OVER 100 TRIALS) FOR DIFFERENT FIXED INTEGRATION WINDOWS.

errors do not get under a certain value even when considering noiseless inertial data. The *Gyroscope noise variation* plots of Fig. 2 also shows the importance of accurate preintegrated rotations for the overall accuracy. One can see that an increase in the gyroscope noise (with noiseless accelerometer measurements) eventually lead to position inaccuracies because of the rotational errors that are larger than when applying only accelerometer noise.

3) Biases, time-shift, and corrections: This series of experiments aim at both analysing the sensitivity of the UGPMs with respect to non-Gaussian perturbations of the input signals, and demonstrating the postintegration correction mechanism developed in this paper. In that regard, the gyroscope and accelerometer measurements have been offset individually with constant additive biases. Table II (a) and (b) show the relative errors of the "raw" UGPMs (without any prior knowledge of the biases present in the data, and without any correction) for different magnitudes of biases. The numbers are consistent with the curves of the previous experiment in the fact that preintegration is more sensitive to perturbation on the gyroscope data than on the accelerometer data. Table II also provides the pose error of the UGPMs after correction using the first-order Taylor expansion presented in (20) and (22). One can see that for reasonable bias offset ($< 0.3rad/s$ and $< 1m/s^2$) with respect to the prior knowledge of the biases, the postintegration correction performs well and leads to errors in the same order of magnitude as for the bias-free scenario.

Additionally to the gyroscope and accelerometer bias analysis, Table II (c) shows the effect of inter-sensor temporal asynchrony. To simulate the issue of time-shift between an IMU and another sensor, preintegration has been performed using erroneously stamped inertial data (constant offset). The results show that for slow motion UGPMs can be corrected to bias-free levels when the time-shift is under $\approx 50ms$. In the case of fast motion, the assumption of local linearity between the preintegrated measurements with respect to the time-shift is much weaker and decent correction can only be expected when the asynchrony is under $\approx 10ms$.

4) Computation time: In this subsection, we discuss the computation time of the different methods benchmarked in

UGPM relative error [%] vs gyroscope biases perturbation - Raw UGPMs and corrected UGPMs ("Cor.")																
Bias norm $[rad/s] \rightarrow$	0.01		0.05		0.1		0.2		0.4		0.6		0.8		1	
Motion and error type $\downarrow$	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.
Slow Rot er.	1.31	0.21	6.49	0.21	13	0.23	25.9	0.33	51.9	0.98	77.8	2.13	104	3.75	130	5.77
Fast Rot er.	0.15	0.03	0.75	0.03	1.51	0.04	3.01	0.08	6.03	0.32	9.05	0.71	12.1	1.26	15.1	1.95
Slow Pos er.	0.71	0.26	3.24	0.27	6.45	0.30	12.9	0.69	25.7	2.57	38.4	5.75	51	10.2	63.4	15.8
Fast Pos er.	0.25	0.11	1.11	0.11	2.2	0.12	4.39	0.23	8.77	0.83	13.1	1.86	17.5	3.31	21.9	5.17

(a)

UGPM relative error [%] vs accelerometer biases perturbation - Raw UGPMs and corrected UGPMs ("Cor.")																
Bias norm $[m/s^2] \rightarrow$	0.01		0.05		0.1		0.2		0.4		0.6		0.8		1	
Motion and error type $\downarrow$	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.
Slow Pos er.	0.37	0.26	1.35	0.26	2.67	0.26	5.33	0.26	10.6	0.27	16	0.26	21.3	0.26	26.6	0.26
Fast Pos er.	0.12	0.11	0.26	0.11	0.49	0.11	0.97	0.11	1.94	0.11	2.92	0.11	3.89	0.11	4.86	0.11

(b)

UGPM relative error [%] vs inter-sensor time-shift - Raw UGPMs and corrected UGPMs ("Cor.")																
Time-shift $[s] \rightarrow$	0.001		0.005		0.01		0.02		0.04		0.06		0.08		0.1	
Motion and error type $\downarrow$	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.	Raw	Cor.
Slow Rot er.	0.24	0.21	0.70	0.24	1.37	0.31	2.73	0.48	5.46	0.88	8.19	1.33	10.9	1.82	13.7	2.36
Fast Rot er.	0.13	0.06	0.65	0.27	1.3	0.54	2.6	1.07	5.19	2.15	7.75	3.31	10.3	4.57	12.7	5.96
Fast Pos er.	0.30	0.26	0.93	0.26	1.83	0.26	3.65	0.27	7.31	0.36	11	0.64	14.6	1.08	18.2	1.62
Slow Pos er.	1.43	0.11	7.13	0.30	14.2	1.12	28.3	4.47	55.5	17.9	80.5	40	103	70.4	121	108

(c)

TABLE II

AVERAGE RELATIVE POSE ERRORS OF THE UGPMS (OVER 100 TRIALS WITH INTEGRATION WINDOWS OF $1s$) IN THE PERSENCE PERTURBATIONS OF THE INERTIAL DATA (ADDITIVE BIASES AND CONSTANT TIME-SHIFT). BOTH THE ERROR FOR THE RAW UGPMS AND THE CORRECTED UGPMS ACCORDING TO THE FIRST-ORDER TAYLOR EXPANSION ARE REPORTED IN THIS TABLE.

VII-B1. We compute the preintegrated measurements over different integration intervals with a low-end desktop computer equipped with an Intel i5-4590 CPU working at 3.30 GHz, and 24 GiB of RAM.

While GP regression is an accurate, probabilistic interpolation method, it suffers from a cubic computation complexity with respect to the number of noisy observations. Consequently, the UGPMs and GPMs can be computationally intensive for longer integration windows. Table III shows the average run time over 100 runs for each method. UGPMs are significantly slower than the other methods but still are able to perform in real-time for integration intervals under 3.5 s (accounting for the fact that the hyperparameters' training in the UGPMs can be omitted due to the marginal improvement of the results, c.f. Subsection VII-B1). However, the per-chunk variant of the UGPMs displays a linear computational complexity $\mathcal{O}(n)$ by dividing long integration intervals in a succession of intervals under $0.2s$. This allows for real-time operations regardless of the length of the integration windows and without any significant difference in accuracy as discussed in Subsection VII-B1.

Table III also shows the efficiency of the LPMs. They represent a great alternative to the GPMs for fast computations, giving similar performances, as shown in Table I.

Computation time [ms]							
Δt $[ms]$	PM [8]	LPM	GPM [13]	GPM [13](train)	UGPM	UGPM (train)	UGPM (chunk)
50	0.013	0.44	1.81	4.08	5.56	8.91	5.04
100	0.019	0.52	2.13	4.84	7.79	10.2	6.04
500	0.069	0.62	8.05	20.3	40.0	55.9	29.7
1000	0.13	3.08	25.9	65.8	176	231	134
1500	0.20	1.61	52.7	131	420	537	136
2000	0.27	1.19	109	235	833	1059	105
2500	0.32	1.44	181	389	1397	1778	127
3000	0.39	1.81	269	624	2180	2798	158
3500	0.45	2.23	383	937	3246	4180	182

TABLE III

AVERAGE COMPUTATION TIME OF PREINTEGRATED MEASUREMENTS (OVER 100 TRIALS) FOR DIFFERENT INTEGRATION INTERVAL LENGTHS $\Delta t = t_2 - t_1$. IMU FREQUENCY 100 Hz, UPSAMPLED FREQUENCIES 500 Hz (GPMS' ROTATIONS AND LPM).

C. Real-world validation

In real-world data, the IMU measurements are subject to slowly varying additive biases. As these are not accurately known at the time of integration, a mechanism is needed to later compensate these data collection artefacts. This set-up aims at demonstrating the soundness of the UGPMs and their first-order Taylor expansion for postintegration IMU bias and time-shift correction in a real-world scenario. The second objective of this experiment is to show the applicability of the UGPMs for inertial-aided multi-modal state estimation.

For that purpose, we have integrated the UGPMs into a lidar-inertial localisation and mapping framework [14]. This framework uses the UGPMs to asynchronously characterise the system's motion while moving in the environment. It allows for the rigorous correction of the motion distortion in the lidar scans and constrains the system's pose between two consecutive pose estimates. The lidar data is used in frame-to-frame constraints based on the minimisation of point-to-line and point-to-plane distances.

Data have been collected in an indoor environment using a

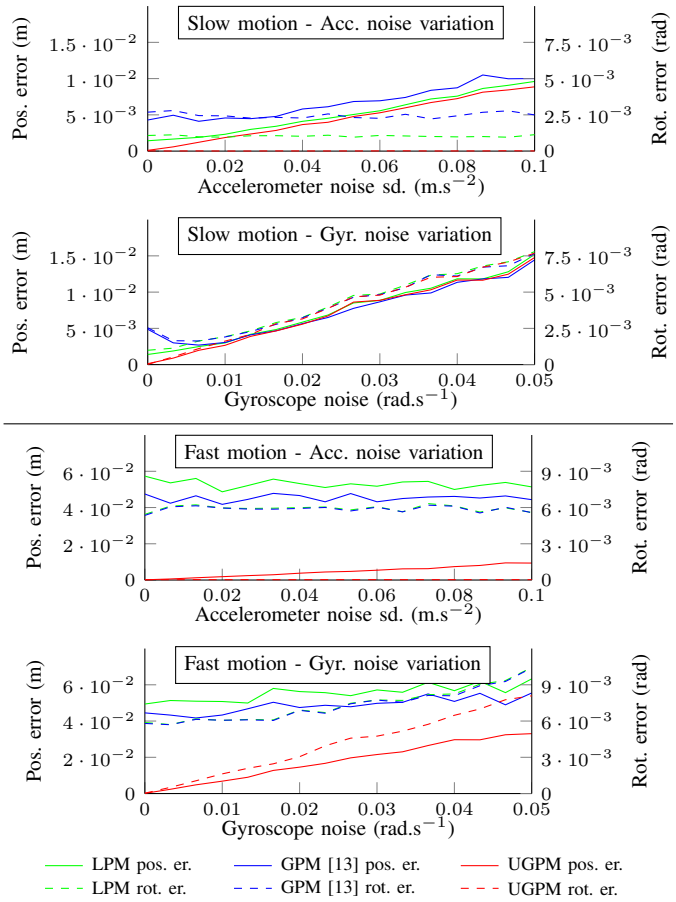

Fig. 2. Average preintegrated error in simulated environments (over 100 trials) for different levels of IMU noise. The position error (pos. error) plots correspond to the left axis and the rotation error (rot. error) to the right axis. The integration interval length is 1.0 s. IMU frequency 100 Hz, upsampled frequencies 500 Hz (GPMs' rotations and LPM).

Velodyne VLP-16 lidar and an Xsens Mti3 IMU. The dataset has a duration of 30 s. Fig. 3 shows the resulting map and trajectory. It demonstrates the ability for the UGPMs to be integrated seamlessly in any inertial-aided state estimation framework.

VIII. Conclusion

We presented a novel continuous integration method over SO(3). This is a challenging task as there is no known closed-form solution to the differential equations that govern the rotational dynamics of a system. The proposed method uses GPs to model the temporal derivatives of the system's rotation vector. The training data of these GPs are found through the minimisation of a cost function that relates the GP-modelled dynamics and the system's angular velocity measurements. Using GP regression and the application of linear operators to the kernels, the orientation of the system can be inferred at any timestamp. This integration method is used for the generation of UGPMs that are our proposed new type of preintegrated measurements for analytical inference of the acceleration integrals. Through experiments, we demonstrated that the UGPMs

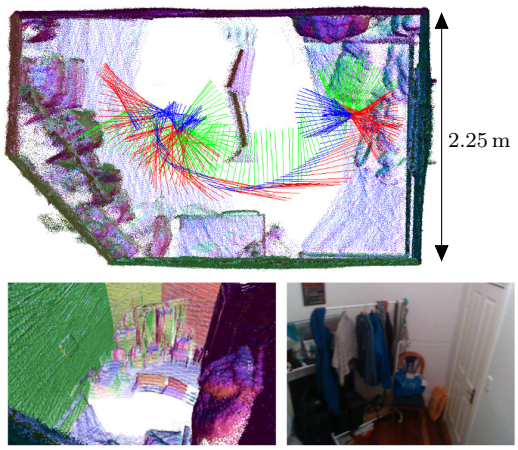

Fig. 3. Validation of the applicability of the proposed preintegrated measurements (UGPMs) for inertial-aided state estimation (here with a lidar-inertial localisation and mapping framework). Top: Top-view of the resulting map and trajectory. Bottom left: Inner view of the map. Bottom right: Camera picture for reference (not used in the estimation process).

outperform the current state-of-the-art preintegration methods and that they are suited for seamless integration in multi-modal inertial-aided navigation systems. Our results also highlight the importance of the proper treatment of the angular velocities as inaccuracies in the integrated rotations lead to inaccurate position preintegrated measurements.

To date, the UGPMs' components are not correlated (diagonal covariance matrix) as per the use of independent GPs. Additionally, the kernels' hyperparameters for the rotational part are set according to an arbitrary heuristic. Our future work includes the investigation of multi-output GP regression for the generation of correlated UGPMs, and studying the literature on GP hyperparameter optimisation.

Appendix
Jacobians for IMU biases correction

This appendix provide the expression of $\frac{\partial \mathbf{f}_{Wj}}{\partial \mathbf{b}_f}$ and $\frac{\partial \mathbf{f}_{Wj}}{\partial \mathbf{b}_\omega}$ used in (24) and (25). The projected accelerometer values are represented by the $\mathbf{f}_{Wj}$ vectors: $\mathbf{f}_{Wj} = \begin{bmatrix} \left(\Delta\mathbf{R}_{t_1}^{\mathrm{t}_1}(\mathrm{t}_1)(\tilde{\mathbf{f}}(\mathrm{t}_1)-\bar{\mathbf{b}}_f(\mathrm{t}_1))\right)_j \\ \cdots \\ \left(\Delta\mathbf{R}_{t_1}^{\mathrm{t}_Q}(\mathrm{t}_Q)(\tilde{\mathbf{f}}(\mathrm{t}_Q)-\bar{\mathbf{b}}_f(\mathrm{t}_Q))\right)_j \end{bmatrix}$. As $\frac{\partial \tilde{\mathbf{f}}(t)}{\partial \mathbf{b}_f(t)} = \mathbf{I}$, $\frac{\partial \mathbf{f}_{Wj}}{\partial \mathbf{b}_f} = \begin{bmatrix} [\Delta\mathbf{R}_{t_1}^{\mathrm{t}_1}(\mathrm{t}_1)]_{(j,:)} \\ \cdots \\ [\Delta\mathbf{R}_{t_1}^{\mathrm{t}_Q}(\mathrm{t}_Q)]_{(j,:)} \end{bmatrix}$ with $[\mathbf{M}]_{(j,:)}$ being the operator that isolates the j^{th} row of $\mathbf{M}$.

As $\Delta\mathbf{R}_{t_1}^{\mathrm{t}_i}(\mathrm{t}_i)$ depends on $\mathbf{b}_\omega$,

$$\frac{\partial \mathbf{f}_{Wj}}{\partial \mathbf{b}_\omega} = \begin{bmatrix} \frac{\partial \left(\Delta\mathbf{R}_{t_1}^{\mathrm{t}_1}(\mathbf{b}_\omega,\delta_t)(\tilde{\mathbf{f}}(\mathrm{t}_1)-\bar{\mathbf{b}}_f(\mathrm{t}_1))\right)_j}{\partial \mathbf{b}_\omega} \\ \cdots \\ \frac{\partial \left(\Delta\mathbf{R}_{t_1}^{\mathrm{t}_Q}(\mathbf{b}_\omega,\delta_t)(\tilde{\mathbf{f}}(\mathrm{t}_Q)-\bar{\mathbf{b}}_f(\mathrm{t}_Q))\right)_j}{\partial \mathbf{b}_\omega} \end{bmatrix} = \begin{bmatrix} J_{f_j}^1 \\ \cdots \\ J_{f_j}^Q \end{bmatrix}$$

where

$$J^i_{f_j} = \left[\left(\Delta\mathbf{R}^{\mathrm{t}_i}_{t_1}(\bar{\mathbf{b}}_\omega, \bar{\delta}_t)\right)^\top \mathbf{q}_j \left(\tilde{\mathbf{f}}(\mathrm{t}_i) - \bar{\mathbf{b}}_f(\mathrm{t}_i)\right)^\top\right]_{(:)^T} \left.\frac{\partial[\exp\left((\mathbf{a})^\wedge\right)]_{(:)^T}}{\partial\mathbf{a}}\right|_{\mathbf{a}=\mathbf{0}} \frac{\partial\Delta\mathbf{R}^{\mathrm{t}_i}_{t_1}(t)}{\partial\mathbf{b}_\omega},$$

$\mathbf{q}_1 = \begin{bmatrix}1&0&0\end{bmatrix}^\top$, $\mathbf{q}_2 = \begin{bmatrix}0&1&0\end{bmatrix}^\top$, $\mathbf{q}_3 = \begin{bmatrix}0&0&1\end{bmatrix}^\top$, $[\mathbf{M}]_{(:)^T}$ the operator that transforms a r-by-c matrix $\mathbf{M}$ into a rc-by-1 column-major vector, and

$$\left.\frac{\partial[\exp\left((\mathbf{a})^\wedge\right)]_{(:)^T}}{\partial\mathbf{a}}\right|_{\mathbf{a}=\mathbf{0}} = \begin{bmatrix}0&0&0&0&0&1&0&-1&0\\0&0&-1&0&0&0&1&0&0\\0&1&0&-1&0&0&0&0&0\end{bmatrix}^\top.$$

REFERENCES

[1] Sean Anderson and Timothy D. Barfoot. Full STEAM ahead: Exactly sparse Gaussian process regression for batch continuous-time trajectory estimation on SE(3). *IEEE International Conference on Intelligent Robots and Systems*, 2015-Decem(3):157–164, 2015.

[2] Timothy D. Barfoot, Chi Hay Tong, and Simo Särkkä. Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression. *Robotics: Science and Systems*, 2014.

[3] Michael Bosse, Robert Zlot, and Paul Flick. Zebedee : Design of a spring-mounted 3-D range sensor with application to mobile mapping. *IEEE Transactions on Robotics*, 28(October):1–15, 2012.

[4] Michael Boyle. The Integration of Angular Velocity. *Advances in Applied Clifford Algebras*, 27(3):2345–2374, 2017.

[5] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[6] Jeffrey Delmerico and Davide Scaramuzza. A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots. *IEEE International Conference on Robotics and Automation*, pages 2502–2509, 2018.

[7] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. Closed-form preintegration methods for graph-based visual–inertial navigation. *International Journal of Robotics Research*, 38(5):563–586, 2019.

[8] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. *Robotics: Science and Systems*, pages 6–15, 2015.

[9] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017.

[10] Paul Furgale, Timothy D Barfoot, and Gabe Sibley. Continuous-Time Batch Estimation using Temporal Basis Functions. *IEEE International Conference on Robotics and Automation*, pages 2088–2095, 2012.

[11] Patrick Geneva and Kevin Eckenhoff. LIPS: LiDAR-Inertial 3D Plane SLAM. *IEEE International Conference on Intelligent Robots and Systems*, 2018.

[12] Giorgio Grisetti, R Kummerle, Cyrill Stachniss, and W Burgard. A Tutorial on Graph-Based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[13] Cedric Le Gentil, Teresa Vidal-Calleja, and Shoudong Huang. Gaussian Process Preintegration for Inertial-Aided State Estimation. *IEEE Robotics and Automation Letters*, 5(2):2108–2114, 2020.

[14] Cedric Le Gentil, Teresa Vidal-Calleja, and Shoudong Huang. IN2LAAMA: INertial Lidar Localisation Autocalibration And MApping. *IEEE Transactions on Robotics*, 2021.

[15] Todd Lupton and Salah Sukkarieh. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics*, 28(1):61–76, 2012.

[16] Raul Mur-Artal and Juan D. Tardos. Visual-Inertial Monocular SLAM with Map Reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.

[17] Tong Qin, Peiliang Li, and Shaojie Shen. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Transactions on Robotics*, 34(4):1–17, 2018.

[18] C E Rasmussen and C K I Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[19] Simo Särkkä. Linear operators and stochastic partial differential equations in Gaussian process regression. *Artificial Neural Networks and Machine Learning–ICANN 2011*, pages 151–158, 2011.

[20] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142, 2020.

[21] Tixiao Shan and Brendan Englot. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. *IEEE International Conference on Intelligent Robots and Systems*, pages 4758–4765, 2018.

[22] Yulin Yang, Benzun Pious, Wisely Babu, Chuchu Chen, Guoquan Huang, and Liu Ren. Analytic Combined IMU Integration (ACI 2) For Visual Inertial Navigation. *IEEE International Conference on Robotics and Automation*, 2020.