Gaussian Process Preintegration for Inertial-Aided State Estimation

Cedric Le Gentil, Teresa Vidal-Calleja, and Shoudong Huang

***Abstract*—In this paper, we present Gaussian Process Preintegration, a preintegration theory based on continuous representations of inertial measurements. A novel use of linear operators on Gaussian Process kernels is employed to generate the proposed Gaussian Preintegrated Measurements (GPMs). This formulation allows the analytical integration of inertial signals on any time interval. Consequently, GPMs are especially suited for asynchronous inertial-aided estimation frameworks. Unlike discrete preintegration approaches, the proposed method does not rely on any explicit motion-model and does not suffer from numerical integration noise. Additionally, we provide the analytical derivation of the Jacobians involved in the first-order expansion for postintegration bias and inter-sensor time-shift correction. We benchmarked the proposed method against existing preintegration methods on simulated data. Our experiments show that GPMs produce the most accurate results and their computation time allows close-to-real-time operations. We validated the suitability of GPMs for inertial-aided estimation by integrating them into a lidar-inertial localisation and mapping framework.**

***Index Terms*—Sensor Fusion, SLAM, Localization, Autonomous Vehicle Navigation**

I. Introduction

PERCEPTION and ego-motion estimation are critical components of every autonomous system. Despite tremendous improvements in sensing technologies, a system cannot rely only on one modality for robust and safe operations. Numerous state estimation algorithms have been proposed to estimate a system trajectory based on complementary multi-modal data. Inertial Measurements Units (IMUs) became ubiquitous in the field of multi-sensor state estimation as they are affordable, light-weight and provide proprioceptive information. In 2012, the authors of [1] introduced a concept called preintegration. It consists of pre-processing discrete inertial measurements into a smaller amount of pseudo-measurements. The preintegrated measurements had become widely popular for inertial-aided estimation as they allow a reduction of the overall trade-off between accuracy and computation time. This popularity gain has been accelerated by the integration of ready-to-use preintegrated inertial factors in robotic state estimation frameworks like GTSAM [2]. In this present work, we extended the concept of preintegration to pre-process inertial data using Gaussian Processes as non-parametric continuous representations of the measurements. Compared to the original preintegrated measurements, the proposed Gaussian Preintegrated Measurements (GPMs) yield more accurate pose predictions within contained computation time.

Manuscript received: September, 10, 2019; Revised December, 5, 2019; Accepted January, 13, 2020.

This paper was recommended for publication by Editor Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments.

All the authors are with the Centre for Autonomous Systems at the Faculty of Engineering and IT, University of Technology Sydney, Australia. cedric.legentil@student.uts.edu.au, {teresa.vidalcalleja, shoudong.huang}@uts.edu.au

Digital Object Identifier (DOI): see top of this page.

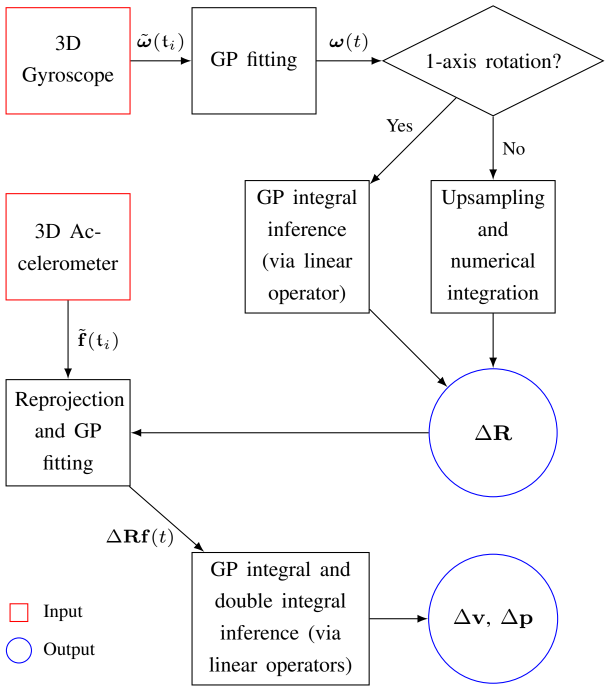

3D Gyroscope
$\tilde{\omega}(t_i)$
GP fitting
$\omega(t)$
1-axis rotation?
Yes
No
GP integral inference (via linear operator)
Upsampling and numerical integration
3D Accelerometer
$\tilde{\mathbf{f}}(t_i)$
Reprojection and GP fitting
$\Delta\mathbf{R}$
$\Delta\mathbf{R}\mathbf{f}(t)$
GP integral and double integral inference (via linear operators)
$\Delta\mathbf{v}$, $\Delta\mathbf{p}$
Input
Output

Fig. 1: Overview of the proposed method of Gaussian Process (GP) Preintegration

The affordability of both monocular cameras and IMUs makes visual-inertial algorithms quite popular in the literature. Initially limited by computational power, online state estimation was mostly relying on filter-based methods such as the one presented in [3]. Later frameworks like [4] and [5] combined both local and global optimisations to estimate the system trajectory in real-time as well as creating consistent maps with loop-closure detections. Originally expensive and bulky, lidars are now affordable sensors that provide reliable geometric information about the surrounding environment. The system presented in [6] combines both a 2D lidar with an IMU on a spring mechanism to create 3D maps. Still considered as a state-of-the-art method, LOAM [7] cleverly uses frame-to-frame and frame-to-map scan registration loosely coupled with IMU data to outperform any other lidar-inertial approaches on the KITTI odometry benchmark [8]. All the methods above leverage only two sensors at a time but fusion with more sensors is possible as demonstrated in [9] with a visual-inertial-GPS system, or [10] with a lidar-visual-inertial odometry pipeline.

Whether it is for visual-inertial [4], [5] or lidar-inertial [11], [12] estimation, many methods rely on the concept of preintegration originally presented in [1]. In the context of visual-inertial localisation and mapping, [1] introduced preintegrated measurements that combine IMU readings collected between two visual keyframes. Unlike the classic integration of inertial data, preintegration provides pseudo-measurements that are independent of the linearisation point preventing repetitive computations during the process of state estimation. As shown in [13], the authors of [14] and [15] introduced a lightweight and accurate Visual-Inertial Odometry (VIO) algorithm. The high performance of this algorithm has been made possible with the extension of the preintegration theory to rotation manifold [14], [15], instead of the Euler angles representation used in [1]. In [11], the inertial data are combined into closed-form preintegrated measurements [16] relying on the assumption of constant local acceleration. In our previous work [17], we introduced the concept of Upsampled Preintegrated Measurements (UPMs). These measurements are based on GP regression and do not rely on any explicit assumption about the system dynamics.

One issue that arises with multi-sensor systems is the data synchronisation. Even if the readings from each modality are collected along with accurate real-time timestamps, the sensor frequencies are generally different and out of phase. Frameworks like [6] and [18] address this problem by using continuous state-representations. While providing tools to handle rolling-shutter-like sensors (spinning lidars, rolling-shutter cameras, etc.), these methods rely on heuristics about the system motion. In a similar way to the UPMs, the proposed method is based on a nonparametric continuous representation of the inertial measurements. It allows inference of inertial information for any given timestamp without the need for any explicit motion model. Consequently, GPMs can be used with asynchronous inertial-aided platforms as well as with rolling-shutter-like sensors. In addition, the accelerometer and gyroscope measurements themselves could even be collected at different frequencies.

The main contribution of this work is the theoretical derivation of the GPMs leveraging GP models of the inertial measurements and linear operators applied on GP kernels [19]. Moreover, this paper also provides the reader with the tools to integrate the GPMs into any inertial-aided state estimation algorithms with the derivation of postintegration IMU biases and inter-sensor time-shift corrections.

The structure of this paper is as follows. Section II introduces definitions and background about preintegration. Section III presents the theory of GPMs. The derivation of postintegration IMU biases and inter-sensor time-shift corrections is detailed in Section IV. Section V is dedicated to the experiments through the benchmark of GPMs against existing methods, and the validation in a multi-modal estimation framework. Finally, Section VI presents the conclusion and suggestions for future work.

II. Definitions and background

A. System description

Let us consider a 6-DoF-IMU composed of a 3-axis accelerometer and a 3-axis gyroscope. The inertial data acquired consists of proper accelerations $\tilde{\mathbf{f}}(\mathrm{t}_i)$ and angular velocities $\tilde{\boldsymbol{\omega}}(\mathrm{t}_i)$ at time t_i $(i = 1, \dots, Q)$ measured in the inertial frame $\mathfrak{F}_I$. The IMU orientation (rotation matrix), position and velocity at time t_i are denoted $\mathbf{R}_W^{t_i}$, $\mathbf{p}_W^{t_i}$ and $\mathbf{v}_W^{t_i}$, respectively. The subscript $_W$ corresponds to the world fixed frame $\mathfrak{F}_W$.

The inertial measurements are modelled considering additive biases and noise terms as follows:

$$\tilde{\mathbf{f}}(t) = \mathbf{R}_W^t(t)^\top(\mathbf{f}(t) - \mathbf{g}_W) + \mathbf{b}_f(t) + \eta_f(t), \tag{1}$$

$$\tilde{\boldsymbol{\omega}}(t) = \boldsymbol{\omega}(t) + \mathbf{b}_\omega(t) + \eta_\omega(t), \tag{2}$$

with $\mathbf{f}$ being the true linear acceleration of the sensor in the world frame $\mathfrak{F}_W$, $\boldsymbol{\omega}$ the true instantaneous angular velocity of the inertial frame relative to $\mathfrak{F}_W$, $\mathbf{g}_W$ the gravity vector in $\mathfrak{F}_W$, $\mathbf{b}_f$ and $\mathbf{b}_\omega$ slowly varying sensor biases, η_f and η_ω zero-mean Gaussian noises for the linear accelerations and angular velocities respectively.

By definition, the dynamics of the sensor is given by:

$$\dot{\mathbf{R}}_W^t(t) = \mathbf{R}_W^t(t)\boldsymbol{\omega}(t)^\wedge, \tag{3}$$

$$\dot{\mathbf{v}}_W^t(t) = \mathbf{f}(t), \tag{4}$$

$$\dot{\mathbf{p}}_W^t(t) = \mathbf{v}_W^t(t), \tag{5}$$

where $\dot{}$ is the differentiation operator, and $^\wedge$ is the operator that transforms a vector into a skew-symmetric matrix as

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix}^\wedge = \begin{bmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{bmatrix}. \tag{6}$$

B. IMU preintegration

Given known initial conditions at $t = t_1$, the sensor pose and velocity at $t_2 > t_1$ can be computed by integrating (3), (4) and (5):

$$\mathbf{R}_W^{t_2} = \mathbf{R}_W^{t_1}\Big(\prod_{t_1}^{t_2}\exp\big(\boldsymbol{\omega}(t)^\wedge\big)^{dt}\Big), \tag{7}$$

$$\mathbf{v}_W^{t_2} = \mathbf{v}_W^{t_1} + \int_{t_1}^{t_2}\mathbf{f}(t)dt, \tag{8}$$

$$\mathbf{p}_W^{t_2} = \mathbf{p}_W^{t_1} + \mathbf{v}_W^{t_1}\Delta t + \int_{t_1}^{t_2}\int_{t_1}^{t}\mathbf{f}(s)dsdt \tag{9}$$

with $\Delta t = t_2 - t_1$. In the absence of sensor noise, (7), (8) and (9) can be expressed as a function of the IMU readings $\tilde{\mathbf{f}}$ and $\tilde{\boldsymbol{\omega}}$:

$$\mathbf{R}_W^{t_2} = \mathbf{R}_W^{t_1}\Big(\prod_{t_1}^{t_2}\exp\big((\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))^\wedge\big)^{dt}\Big), \tag{10}$$

$$\mathbf{v}_W^{t_2} = \mathbf{v}_W^{t_1} + \mathbf{g}_W\Delta t + \int_{t_1}^{t_2}\mathbf{R}_W^t(t)\big(\tilde{\mathbf{f}}(t) - \mathbf{b}_f(t)\big)dt, \tag{11}$$

$$\begin{aligned}\mathbf{p}_W^{t_2} =& \mathbf{p}_W^{t_1} + \mathbf{v}_W^{t_1}\Delta t + \frac{\mathbf{g}_W\Delta t^2}{2}\\ &+ \int_{t_1}^{t_2}\int_{t_1}^{t}\mathbf{R}_W^s(s)\big(\tilde{\mathbf{f}}(s) - \mathbf{b}_f(s)\big)dsdt.\end{aligned} \tag{12}$$

The preintegration originally presented in [1] reformulates (11) and (12) using the fact that $\mathbf{R}_W^t(t) = \mathbf{R}_W^{t_1}\mathbf{R}_{t_1}^t(t)$:

$$\mathbf{v}_W^{t_2} = \mathbf{v}_W^{t_1} + \mathbf{g}_W\Delta t + \mathbf{R}_W^{t_1}\int_{t_1}^{t_2}\mathbf{R}_{t_1}^t(t)\big(\tilde{\mathbf{f}}(t) - \mathbf{b}_f(t)\big)dt, \tag{13}$$

$$\begin{aligned}\mathbf{p}_W^{t_2} =& \mathbf{p}_W^{t_1} + \mathbf{v}_W^{t_1}\Delta t + \frac{\mathbf{g}_W\Delta t^2}{2}\\ &+ \mathbf{R}_W^{t_1}\int_{t_1}^{t_2}\int_{t_1}^{t}\mathbf{R}_{t_1}^s(s)\big(\tilde{\mathbf{f}}(s) - \mathbf{b}_f(s)\big)dsdt.\end{aligned} \tag{14}$$

The preintegrated measurement are defined as

$$\Delta\mathbf{R}_{t_1}^{t_2} = \prod_{t_1}^{t_2}\exp\big((\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))^\wedge\big)^{dt}, \tag{15}$$

$$\Delta\mathbf{v}_{t_1}^{t_2} = \int_{t_1}^{t_2}\mathbf{R}_{t_1}^t(t)\big(\tilde{\mathbf{f}}(t) - \mathbf{b}_f(t)\big)dt, \tag{16}$$

$$\Delta\mathbf{p}_{t_1}^{t_2} = \int_{t_1}^{t_2}\int_{t_1}^{t}\mathbf{R}_{t_1}^s(s)\big(\tilde{\mathbf{f}}(s) - \mathbf{b}_f(s)\big)dsdt. \tag{17}$$

In [1] and [14], (15), (16) and (17) are integrated numerically directly considering the discrete raw IMU measurements.

III. Gaussian Preintegrated Measurements

Our previous works [17] and [12] consider continuous representations of the inertial data to independently upsample these signals before numerically integrating (15), (16) and (17). We named these interpolated measurements Upsampled Preintegrated Measurements (UPMs). Here, the proposed method also leverages continuous models of inertial measurements. The major difference is that in this case we leverage the use of linear operators [19] to analytically compute the integral of inertial readings. Fig. 1 shows the two step approach (rotation first, then velocity and position) proposed for inference of GPMs. In this section, we assume that the sensors biases are known. In real-world scenarios, the value of the biases is not accurately known a priori. A first-order expansion technique is presented in Section IV to allow postintegration bias and inter-sensor time-shift corrections.

A. Gaussian Process regression

In [17] and [12], Gaussian Process regression [20] is used to perform non-parametric probabilistic interpolation. This inference method does not assume any motion model. Given a signal modelled with a GP as

$$\begin{aligned}h(x) &\sim \mathcal{GP}\big(0, \mathbf{K}_h(x, x')\big),\\ y_i &= h(x_i) + \eta, \qquad \eta \sim \mathcal{N}(0, \sigma_y^2),\end{aligned} \tag{18}$$

the interpolated mean and covariance are

$$\begin{aligned}y^*(x) &= \mathbf{K}_h(x, \mathbf{x})\big[\mathbf{K}_h(\mathbf{x}, \mathbf{x}) + \sigma_y^2\mathbf{I}\big]^{-1}\mathbf{y},\\ \operatorname{cov}(y^*) &= \mathbf{K}_h(x, x)\\ &\quad -\mathbf{K}_h(x, \mathbf{x})\big[\mathbf{K}_h(\mathbf{x}, \mathbf{x}) + \sigma_y^2\mathbf{I}\big]^{-1}\mathbf{K}_h(x, \mathbf{x})^\top,\end{aligned} \tag{19}$$

with $\mathbf{y}$ being the vector of training values at $\mathbf{x}$, and $\mathbf{K}_h(.,.)$ the matrix of covariances evaluated with the kernel covariance function between each pair of arguments.

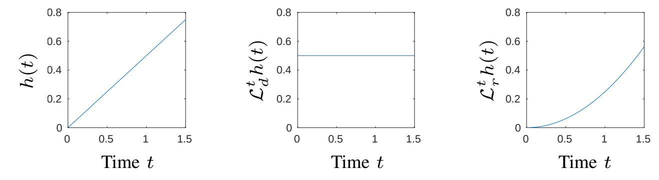


Fig. 2: Graphical examples of two linear operators used in this work (center and right) applied to the signal $h(t)$ (shown on the left). With $\mathcal{L}_d^t h(t) = \frac{\partial h(t)}{\partial t}$ and $\mathcal{L}_r^t h(t) = \int_{t_1}^t h(x)dx$ (with $t_1 = 0$).

B. GPM - Rotation

Equation (3) does not have a general solution [21]. Nonetheless, if the sensor rotation is constrained around one-axis, as in many ground vehicle applications, the infinitesimal rotations become commutative. Therefore, in that case, (3) can be solved as:

$$\Delta\mathbf{R}_{t_1}^{t_2} = \exp\Big(\Delta\mathbf{r}_{t_1}^{t_2\wedge}\Big), \quad \Delta\mathbf{r}_{t_1}^{t_2} = \int_{t_1}^{t_2}(\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))dt, \tag{20}$$

where the three components of the integral can be computed independently. The integral is then rewritten with a linear operator

$$\Delta\mathbf{r}_{t_1}^{t_2} = \int_{t_1}^{t_2}(\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))dt = \begin{bmatrix}\mathcal{L}_r^t(\omega_1(t) - b_{\omega_1}(t))\\ \mathcal{L}_r^t(\omega_2(t) - b_{\omega_2}(t))\\ \mathcal{L}_r^t(\omega_3(t) - b_{\omega_3}(t))\end{bmatrix}, \tag{21}$$

where the superscript of $\mathcal{L}_r^\bullet$ indicates the variable it is operating on. Note that in the case of linear operators, $\mathcal{L}^\bullet h(\bullet)$ does not correspond to the multiplication of a matrix or vector $\mathcal{L}^\bullet$ with $h(\bullet)$. It represents the application of the operator $\mathcal{L}^\bullet$ on $h(\bullet)$. For example, given $\mathcal{L}^\bullet$ the differentiation operator, $\mathcal{L}^\bullet h(\bullet)$ corresponds to $\frac{\partial h(\bullet)}{\partial \bullet}$. Fig. 2 provides examples of the application of $\mathcal{L}_r^t$ and $\mathcal{L}_d^t$ (presented later in (35) and (36)). As per [19], by modelling $(\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))$ with GPs,

$$(\omega_j(t) - b_{\omega_j}(t)) \sim \mathcal{GP}\big(0, \mathbf{K}_{r_j}(t, t')\big), \tag{22}$$

the component of $\Delta\mathbf{r}_{t_1}^{t_2}$ are inferred as

$$\Delta r_{j\,t_1}^{\,t_2 *} = \mathcal{L}_r^t\mathbf{K}_{r_j}(t, \mathrm{t})\big[\mathbf{K}_{r_j}(\mathrm{t}, \mathrm{t}) + \sigma_{\omega_j}^2\mathbf{I}\big]^{-1}\boldsymbol{\omega}_j, \tag{23}$$

$$\begin{aligned}\sigma^2_{\Delta r_{j\,t_1}^{\,t_2}}{}^* &= \mathcal{L}_r^t\mathbf{K}_{r_j}(t, t)\mathcal{L}_r^t\\ &\quad -\mathcal{L}_r^t\mathbf{K}_{r_j}(t, \mathrm{t})\big[\mathbf{K}_{r_j}(\mathrm{t}, \mathrm{t}) + \sigma_{\omega_j}^2\mathbf{I}\big]^{-1}\mathbf{K}_{r_j}(t, \mathrm{t})^\top\mathcal{L}_r^t.\end{aligned} \tag{24}$$

with $\boldsymbol{\omega}_j$ the vector of training values $(\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t))_j$. 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.

If the rotation spans over multiple axes between t_1 and t_2, the rotational preintegrated measurement $\Delta\mathbf{R}_{t_1}^{t_2}$ is computed numerically after upsampling the gyroscope data with the classic GP inference (19) as in [17] and [12].

C. GPM - Velocity and position

In the case of noiseless measurements and perfect knowledge of the gyroscope biases, $\mathbf{R}_{t_1}^\bullet(\bullet) = \Delta\mathbf{R}_{t_1}^\bullet(\bullet)$. Consequently, velocity and position preintegrated measurements are functions of both the gyroscope and accelerometer readings. Unfortunately, the preintegrated measurements $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ cannot be expressed analytically solely based on linear operators and the independent GP models of raw inertial measurements. To overcome this issue and allow the direct (non-iterative) GP inference of velocity and position preintegrated measurements, the proposed method models the accelerometer data after their reprojection in $\mathfrak{F}_I^{t_1}$, the inertial frame at t_1, according to the rotational preintegrated measurements. Thus, modelling these new signals with GP models as

$$\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_j \sim \mathcal{GP}\left(0,\mathbf{K}_{f_j}(t,t')\right), \tag{25}$$

and rewriting (16) and (17) as

$$\Delta\mathbf{v}_{t_1}^{t_2} = \begin{bmatrix}\mathcal{L}_v^t\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_1\\ \mathcal{L}_v^t\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_2\\ \mathcal{L}_v^t\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_3\end{bmatrix} \text{ and} \tag{26}$$

$$\Delta\mathbf{p}_{t_1}^{t_2} = \begin{bmatrix}\mathcal{L}_p^t\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_1\\ \mathcal{L}_p^t\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_2\\ \mathcal{L}_p^t\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_3\end{bmatrix}, \tag{27}$$

each component of $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ is inferred independently using (23) and (24) replacing $\mathbf{K}_{r_j}(.,.)$ with $\mathbf{K}_{f_j}(.,.)$, $\boldsymbol{\omega}_j$ with $\mathbf{f}_j$, and $\mathcal{L}_r$ with $\mathcal{L}_v$ for $\Delta\mathbf{v}_{t_1}^{t_2}$, and $\mathcal{L}_p$ for $\Delta\mathbf{p}_{t_1}^{t_2}$. The vector $\mathbf{f}_j$ consists of the set of training values $\left(\Delta\mathbf{R}_{t_1}^t(t)(\tilde{\mathbf{f}}(t)-\mathbf{b}_f(t))\right)_j$ at $t=\mathrm{t}_i$ for every t_i present in the training data.

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

By itself, the integration of inertial data is prone to large drift. In addition to Gaussian noise and the actual physical values of acceleration or angular velocity, the raw inertial measurements contain a generally unknown bias component. In the context of sensor fusion, and under some specific observability conditions [22], these biases can be estimated. For that purpose, the authors of the pioneer paper on preintegration [1] introduced a first-order expansion to correct biases after integration under the assumption of constant biases during the integration interval. In our previous work [17], we extended that concept to integrate the estimation of inter-sensor time-shift δ_t. The preintegrated measurements (15), (16) and (17) can be rewritten as

$$\begin{aligned}\Delta\mathbf{R}_{t_1}^{t_2}(\mathbf{b}_\omega,\delta_t) \approx& \Delta\mathbf{R}_{t_1}^{t_2}(\bar{\mathbf{b}}_\omega,\bar{\delta}_t)\\ &\exp\left(\left(\frac{\partial\Delta\mathbf{r}_{t_1}^{t_2}}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}_\omega+\frac{\partial\Delta\mathbf{r}_{t_1}^{t_2}}{\partial\delta_t}\hat{\delta}_t\right)^\wedge\right)\\ \Delta\mathbf{v}_{t_1}^{t_2}(\mathbf{b}_f,\mathbf{b}_\omega,\delta_t) \approx& \Delta\mathbf{v}_{t_1}^{t_2}(\bar{\mathbf{b}}_f,\bar{\mathbf{b}}_\omega,\bar{\delta}_t)+\frac{\partial\Delta\mathbf{v}_{t_1}^{t_2}}{\partial\mathbf{b}_f}\hat{\mathbf{b}}_f\\ &+\frac{\partial\Delta\mathbf{v}_{t_1}^{t_2}}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}_\omega+\frac{\partial\Delta\mathbf{v}_{t_1}^{t_2}}{\partial\delta_t}\hat{\delta}_t\\ \Delta\mathbf{p}_{t_1}^{t_2}(\mathbf{b}_f,\mathbf{b}_\omega,\delta_t) \approx& \Delta\mathbf{p}_{t_1}^{t_2}(\bar{\mathbf{b}}_f,\bar{\mathbf{b}}_\omega,)+\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\mathbf{b}_f}\hat{\mathbf{b}}_f\\ &+\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}_\omega+\frac{\partial\Delta\mathbf{p}_{t_1}^{t_2}}{\partial\delta_t}\hat{\delta}_t,\end{aligned} \tag{28}$$

with $\mathbf{b}_f=\bar{\mathbf{b}}_f+\hat{\mathbf{b}}_f$, $\mathbf{b}_\omega=\bar{\mathbf{b}}_\omega+\hat{\mathbf{b}}_\omega$, and $\delta_t=\bar{\delta}_t+\hat{\delta}_t$. Note that $\bar{\bullet}$ denotes the prior knowledge of the value at the time of preintegration and $\hat{\bullet}$ represents the correction.

The rest of this section describes how the different Jacobians in (28) are computed according to the GPM formulation.

A. Rotation GPM Jacobians

Similarly to the GPMs calculation, the computation of the Jacobians for the rotational preintegrated measurements is split into two scenarios. If the motion contains 3D rotations, the Jacobian of $\Delta\mathbf{R}_{t_1}^{t_2}$ with respect to the gyroscope biases $\mathbf{b}_\omega$ is computed iteratively as in [14]. The Jacobian of $\Delta\mathbf{R}_{t_1}^{t_2}$ with respect to the inter-sensor time-shift δ_t can easily be computed numerically by offsetting the integration limits of $\mathcal{L}_r^t$. The following paragraphs detail the computation of these Jacobians in the case of 1D rotations.

1) Gyroscope biases: In the inference of $\Delta r_{t_1}^{t_2}$, as per (23), the biases only impact the training values $\boldsymbol{\omega}_j$. Consequently,

$$\frac{\partial\Delta r_{j\,t_1}^{\ t_2}}{\partial\mathbf{b}_\omega}=\mathcal{L}_r^t\mathbf{K}_{r_j}(t,\mathrm{t})\left[\mathbf{K}_{r_j}(\mathrm{t},\mathrm{t})+\sigma_{\omega_j}^2\mathbf{I}\right]^{-1}\frac{\partial\boldsymbol{\omega}_j}{\partial\mathbf{b}_\omega} \tag{29}$$

The Jacobian of $\boldsymbol{\omega}_j$ with respect to $\mathbf{b}_\omega$ is a column vector of -1 (a simple additive bias).

2) Inter-sensor time-shift: The Jacobian of $\Delta r_{t_1}^{t_2}$ with respect to δ_t (the derivative of (21) with respect to t_1) can be written using another linear operator:

$$\frac{\partial\Delta\mathbf{r}_{t_1}^{t_2}}{\partial\delta_t}=\frac{\partial}{\partial t_1}\int_{t_1}^{t_1+\Delta t}(\tilde{\boldsymbol{\omega}}(t)-\mathbf{b}_\omega(t))dt \tag{30}$$

$$=\begin{bmatrix}\mathcal{L}_{r_{\delta_t}}^t(\omega_1(t)-b_{\omega_1}(t))\\ \mathcal{L}_{r_{\delta_t}}^t(\omega_2(t)-b_{\omega_2}(t))\\ \mathcal{L}_{r_{\delta_t}}^t(\omega_3(t)-b_{\omega_3}(t))\end{bmatrix}, \tag{31}$$

with $\Delta t=t_2-t_1$. Therefore the Jacobian can be computed with (23) replacing $\mathcal{L}_r^t$ with $\mathcal{L}_{r_{\delta_t}}^t$.

B. Velocity and position GPM Jacobians

The derivation of the Jacobians of (28) is similar for both, the velocity and position cases.

1) Accelerometer and gyroscope biases: As for $\Delta\mathbf{R}_{t_1}^{t_2}$, the inference of $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ depends on the IMU biases solely via their training values $\mathbf{f}_j$. Therefore,

$$\frac{\partial\Delta\mathbf{v}_{j\,t_1}^{\ t_2}}{\partial b_\bullet}=\mathcal{L}_v^t\mathbf{K}_{r_j}(t,\mathrm{t})\left[\mathbf{K}_{r_j}(\mathrm{t},\mathrm{t})+\sigma_{\omega_j}^2\mathbf{I}\right]^{-1}\frac{\partial\mathbf{f}_j}{\partial b_\bullet} \tag{32}$$

$$\frac{\partial\Delta\mathbf{p}_{j\,t_1}^{\ t_2}}{\partial b_\bullet}=\mathcal{L}_p^t\mathbf{K}_{r_j}(t,\mathrm{t})\left[\mathbf{K}_{r_j}(\mathrm{t},\mathrm{t})+\sigma_{\omega_j}^2\mathbf{I}\right]^{-1}\frac{\partial\mathbf{f}_j}{\partial b_\bullet} \tag{33}$$

with $b_\bullet$ being either the accelerometer biases $\mathbf{b}_f$ or the gyroscope biases $\mathbf{b}_\omega$. Note that $\mathbf{f}_j$ depends on $\mathbf{b}_\omega$ as $\Delta\mathbf{R}_{t_1}^t(t)$ depends on $\mathbf{b}_\omega$.

2) Inter-sensor time-shift: The differentiation of $\Delta\mathbf{v}_{t_1}^{t_2}$ and $\Delta\mathbf{p}_{t_1}^{t_2}$ with respect to δ_t cannot be directly inferred with the help of a linear operator as in the case of $\Delta\mathbf{r}_{t_1}^{t_2}$. The main difference comes from the fact that $\mathbf{f}_j$ depends on δ_t as $\Delta\mathbf{R}_{t_1}^t(t)$ depends on δ_t. Consequently the Jacobian of $\Delta\mathbf{v}_{t_1}^{t_2}$ is computed as

$$\begin{aligned}\frac{\partial\Delta\mathbf{v}_{j\,t_1}^{\ t_2}}{\partial\delta_t}=&\mathcal{L}_{v_{\delta_t}}^t\mathbf{K}_{r_j}(t,\mathrm{t})\left[\mathbf{K}_{r_j}(\mathrm{t},\mathrm{t})+\sigma_{\omega_j}^2\mathbf{I}\right]^{-1}\mathbf{f}_j\\ &+\mathcal{L}_v^t\mathbf{K}_{r_j}(t,\mathrm{t})\left[\mathbf{K}_{r_j}(\mathrm{t},\mathrm{t})+\sigma_{\omega_j}^2\mathbf{I}\right]^{-1}\frac{\partial\mathbf{f}_j}{\partial\delta_t}\end{aligned} \tag{34}$$

with

$$\mathcal{L}^t_{v_{\delta_t}} = \frac{\partial}{\partial t_1}\mathcal{L}^t_v = \mathcal{L}^{t_1}_d \mathcal{L}^t_v. \tag{35}$$

The computation of the Jacobian of $\Delta \mathbf{v}^{t_2}_{t_1}$ with respect to δ_t follows (34) replacing $\mathcal{L}_{v_{\delta_t}}$ with

$$\mathcal{L}^t_{p_{\delta_t}} = \frac{\partial}{\partial t_1}\mathcal{L}^t_p = \mathcal{L}^{t_1}_d \mathcal{L}^t_p. \tag{36}$$

Note that the Jacobians of $\mathbf{f}_j$ with respect to δ_t can be computed numerically or deduced from $\frac{\partial \Delta r^t_{t_1}(t)}{\partial \delta_t}$ computed in the previous subsection.

V. Experiments and results

The proposed method has been evaluated quantitatively on simulated data and validated for inertial-aided state estimation with a lidar-inertial platform. Our implementation uses the square exponential kernel.

A. GPM benchmarks

In this subsection we are benchmarking our proposed method (GPM) against our previous work [17] (UPM), and the original on-manifold preintegration method [14] (PM). The evaluation in simulated data consists of generating random trajectories from sinusoidal functions and computing the preintegrated measurements using each of the three aforementioned methods between randomly chosen timestamps (t_1 and t_2). The frequencies of the sinusoidal functions range between 0.05 and $0.4\,\mathrm{Hz}$ for the position, and between 0.15 and $0.7\,\mathrm{Hz}$ for the rotation. The different metrics for evaluation are presented below.

1) Accuracy: This set of experiments have been designed to simulate the VIO scenarios where preintegration is performed at low frequency to constrain the system pose between consecutive keyframes (set-up one), and at higher frequency for feature tracking (set-up two). Our first set-up computes preintegrated measurements over durations ranging anywhere between 1 and 5 seconds. As the trajectories have different characteristics and the integration interval length is not fixed across runs, the chosen metric is the relative error with respect to the travelled (linear or angular) distance. The second experiment has been designed around smaller integration intervals. We compute the absolute pose error for different fixed inference frequencies between 1 and $20\,\mathrm{Hz}$. Looking at the values reported in Table I and II, GPMs and UPMs outperform PMs by around an order of magnitude. In other words, the assumption of constant acceleration during the IMU period introduces a non-negligible integration noise. UPMs upsample the inertial measurements (here by a factor 10) before conducting numerical integration. It reduces the integration noise significantly. The amount of integration noise in UPMs directly depends on the upsampled frequency. This creates a trade-off between accuracy and computation time. GPMs do not suffer from the integration noise as per their analytical approach to the integration. The factor that limits the accuracy of the GPMs is the ability for the GPs to accurately model the true underlying signal.

1D rotations (relative errors %)

Motion	PM		UPM		GPM	
	Rot er.	Pos er.	Rot er.	Pos er.	Rot er.	Pos er.
Slow	0.253	3.54	0.038	0.381	**0.028**	**0.143**
Fast	0.221	2.90	0.024	0.301	**0.008**	**0.073**

3D rotations (relative errors %)

Motion	PM		UPM		GPM	
	Rot er.	Pos er.	Rot er.	Pos er.	Rot er.	Pos er.
Slow	0.316	4.79	**0.035**	0.493	**0.035**	**0.311**
Fast	0.308	4.35	**0.031**	0.433	**0.031**	**0.396**

TABLE I: Average relative error of preintegrated measurements with respect to trajectory length in simulated environments (over 100 trials) for different types of motion. The integration interval length is between 1s and 5s. Characteristics of the trajectories during integration interval (1D rot. slow/1D rot. fast/3D rot. slow/3D rot. fast): avg. distance = $9.7/15/8.8/23\,\mathrm{m}$, avg. velocity = $1.8/3.9/1.9/4.7\,\mathrm{m/s}$, avg angular distance = $2.2/6.8/3.8/16\,\mathrm{rad}$, avg angular velocity = $0.7/2.5/1.3/4.9\,\mathrm{rad/s}$. IMU frequency $100\,\mathrm{Hz}$, accelerometer noise sd $0.02\,\mathrm{m/s^2}$, gyroscope noise sd $0.002\,\mathrm{rad/s}$, UPM upsampled frequency $1\,\mathrm{kHz}$.

1D rotations (absolute errors in mrad and mm)

Query rate (Hz)	PM		UPM		GPM	
	Rot er.	Pos er.	Rot er.	Pos	Rot er.	Pos er.
20	2.41	5.61	0.28	0.60	**0.10**	**0.02**
10	4.91	12.1	1.61	1.32	**1.40**	**0.16**
2	21.3	61.3	3.09	7.19	**0.20**	**0.61**
1	24.9	125	2.51	12.6	**0.28**	**1.80**

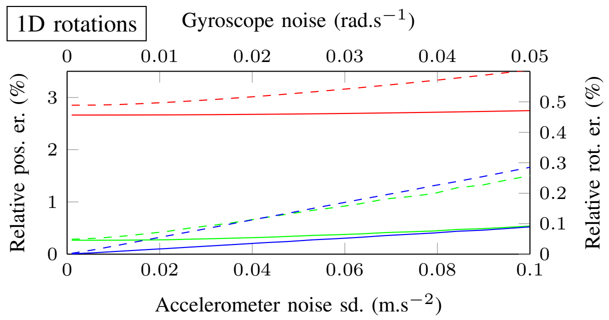
3D rotations (absolute errors in rad and mm)

Query rate (Hz)	PM		UPM		GPM	
	Rot er.	Pos er.	Rot er.	Pos	Rot er.	Pos er.
20	5.48	6.06	**0.80**	0.65	**0.80**	**0.02**
10	11.1	13.1	7.53	2.27	**7.18**	**0.48**
2	35.6	65.0	4.72	7.29	**3.56**	**1.74**
1	37.5	159	**3.75**	15.9	**3.75**	**10.3**

TABLE II: Average absolute error of preintegrated measurements for fast trajectories in simulated environments (over 100 trials) for different fixed query rates.

2) Robustness to noise: This set-up evaluates the three methods for different variations of noise. Figure 3 plots the corresponding relative errors. For motion containing only 1D rotations, the absence of noise pushes the GPMs error towards zero where UPMs and PMs cannot go below a certain error. This observation supports the hypothesis that GPMs are not subject to integration noise. The final error is mainly driven by the kernel ability to model noisy signals. In the presence of 3D rotations, UPMs and GPMs share the same numerical approach to estimate the rotational part of the preintegrated measurements. For that reason, the error difference between the two methods is smaller but GPMs still outperform UPMs.

3) Computation time: While providing more accurate estimates, UPMs and GPMs can be significantly slower than the original preintegration. This experiment collects the average run time (over 50 runs) of the benchmarked methods. The code is executed on a laptop equipped with an Intel i5-6300U CPU working at $2.40\,\mathrm{GHz}$, and $24\,\mathrm{GiB}$ of RAM. Table III presents the values obtained for different length of integration interval (from $10\,\mathrm{Hz}$ to $0.2\,\mathrm{Hz}$). The total execution time of both the UPMs and GPMs can be divided into two parts: hyper-parameter training and inference. In the absence of prior knowledge about the signals, the hyper-parameter training



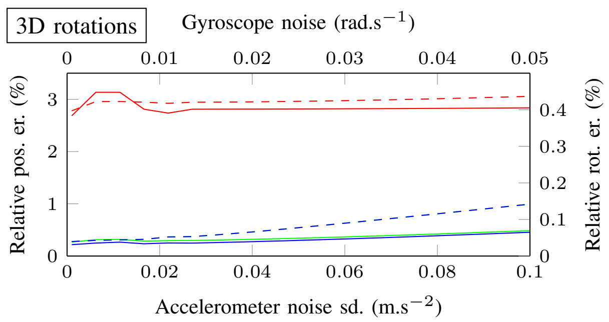


Fig. 3: Average relative error of preintegrated measurements with respect to trajectory length in simulated environments (over 50 trials) for different levels of IMU noise. All the plots in one graph are subject to the bottom and top axis (in other words, the gyroscope and accelerometer noises are simultaneously changing along the x axis). The position error (pos er.) plots relate to the left axis and the rotation error (rot er.) relate to the right axis. The integration interval lengths are between 1 s and 2 s. IMU frequency 100 Hz, UPM upsampled frequency 1 kHz.

1D rotations (computation time ms)					
Interval length (s)	PM	UPM		GPM	
		Train.	Infer.	Train.	Infer.
0.05	0.8	294	18.7	356	1.3
0.1	0.9	335	25.6	399	1.8
0.5	3.1	392	67.3	396	4.2
1	5.8	498	113	502	8.1
2	11.6	1088	279	1085	28.2
3	18.6	2349	534	2332	68.9
4	23.3	4159	824	4144	127
5	28.5	6939	1244	6832	212

3D rotations (computation time ms)					
Interval length (s)	PM	UPM		GPM	
		Train.	Infer.	Train.	Infer.
0.05	0.6	269	16.9	277	10.2
0.1	0.9	291	22.0	298	11.1
0.5	3.1	362	60.2	367	24.8
1	5.8	530	121	529	51.9
2	11.8	1148	297	1110	138
3	17.5	2347	538	2281	270
4	24.1	4352	875	4243	482
5	28.5	7295	1392	7102	817

TABLE III: Average computation time of preintegrated measurements (over 50 trials) for different integration interval lengths. For the UPMs and GPMs, both the hyper-parameter training time and the inference time are shown. IMU frequency 100 Hz, UPM upsampled frequency 1 kHz.

step is needed. Nonetheless, for signals that keep consistent smoothness characteristics over time, the hyper-parameters can be safely reused from one integration interval to another. In such a case, the UPM and GPM computations comply with real-time constraints as the inference time stays under the integration interval length. The difference of computation time between UPMs and GPMs is explained by the larger number of inferences and the iterative numerical integration carried out to generate UPMs. While PMs and UPMs execution times are consistent for the different motion types (1D and 3D rotations), the GPMs are significantly slower when computed over movements that contains 3D rotations. As pointed out before, this is due to the numerical approach needed to estimate $\Delta\mathbf{R}$.

The results of this experiment show that for integration intervals of few seconds, GPMs can be used in close to real-time operations. Furthermore, one should note that these values, for all of the three methods, are generated from single-thread Matlab code that is not optimised for high performance. The GPM computation can easily be parallelised on 3 CPU cores as the hyper-parameter training and inferences are independent on each axis.

The training data covariance matrix $\mathbf{K}_h(\mathbf{x},\mathbf{x})$, in (19), solely depends on the kernel hyper-parameters, the number of training samples used for the integration interval, and the relative temporal position of these samples. In many practical scenarios the IMU readings are generated at a fixed frequency and the integration interval stays the same all along the estimation pipeline. When hyper-parameter training is not required, these conditions can be leveraged to speed-up the UPMs and GPMs inference time by pre-computing the matrix $\left[\mathbf{K}_h(\mathbf{x},\mathbf{x}) + \sigma_y^2\mathbf{I}\right]^{-1}$. Consequently, the inference complexity is reduced from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ as matrix inversion is no longer needed.

B. Validation for inertial-aided state estimation

To validate the suitability of GPMs for inertial-aided estimation, we integrated them into IN2LAAMA [23], a lidar-inertial framework for localisation and mapping. We quantitatively compared IN2LAAMA with PMs and GPMs on simulated data. The following results are computed over nine runs with trajectories of 95.6 m on average. Loop-closures has been deactivated to evaluate only the odometry part. The PM version showed an average final position error of 0.85 m with three failure cases (error above 5 m). Note that the failure cases are results of the accumulated drift of PMs that reach a level greater than the lidar-feature association distance threshold. This leads to unconstrained drifting of the estimate. On the other hand, GPMs performed well by displaying an average final error of 0.25 m.

We conducted real-world experiments with both the PM and GPM versions of IN2LAAMA. Figure 4 shows a trajectory estimate and its associated map created with GPMs. The data have been collected moving the system up-and-down while walking in our lab environment at the University of Technology Sydney. The hand-held sensor suite used is composed of a Velodyne VLP-16 3D lidar and an MTi3 Xsens IMU. Despite the jerky motion, one can see the map crispness as the walls are flat and edges are sharp. Average point-to-plane

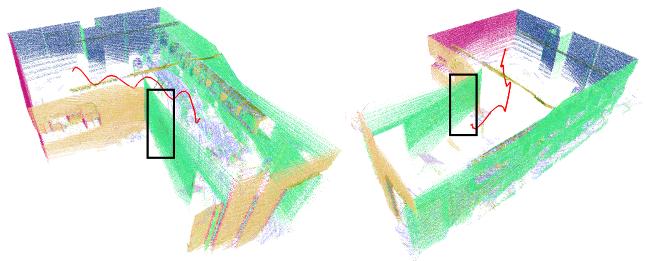


Fig. 4: Two viewpoints of a map created with IN2LAAMA and GPMs. The red line represents the 6.2-meter-long trajectory estimate. The maximum estimated velocity is $1.7\,\mathrm{m.s}^{-1}$. The black rectangles mark the part of the map used for quantitative comparison between PMs and GPMs.

distances have been computed for lidar points belonging to a flat wall (black rectangles in Figure 4) for quantitative comparison between the two preintegration methods. We manually segmented around 150k points in both maps and estimated the plane equation with a principal component analysis. The average point-to-plane distance is $9.0\,\mathrm{mm}$ when using PMs, but it gets as low as $6.3\,\mathrm{mm}$ with GPMs. Whether it is with simulated or real-data, the GPMs integrated seamlessly in IN2LAAMA improving accuracy compared to the original preintegration method.

VI. Conclusion

This paper presents a novel theory for preintegration based on continuous representations of inertial data. While GPs are used to model the IMU measurements, linear operators are applied to the covariance kernels to analytically infer integrals of the continuous signals over any time interval. In combination with a first-order expansion for postintegration bias and inter-sensor time-shift corrections, GPMs are especially suited for inertial-aided estimation frameworks.

Our experiments show that GPMs outperform both UPMs and PMs in terms of accuracy. The GPMs calculation suffers from the cubic computational complexity of GPs but is still suitable for close-to-real-time operations. We validate the use of GPMs in IN2LAAMA, a lidar-inertial localisation and mapping framework. On the theory side, future work includes releasing the assumption of constant bias during preintegration and developing an analytical approximation for the integration of gyroscope measurements in the 3D-rotation case. In terms of implementation for real-time operations, query rates above approximately $2\,\mathrm{Hz}$ (actual frequency subject to computer performance and code optimisation) do not allow the hyper-parameter training from scratch for each integration interval. While one can use heuristics, pre-training or sporadic training (every N integration intervals), a more appealing approach is the use of filtering methods to update the hyper-parameters online. Thus the computation cost per integration interval is substantially reduced while the hyper-parameter are tracked over time.

References

[1] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, 2012.

[2] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Tech. Rep. September, 2012.

[3] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3565–3572, 2007.

[4] R. Mur-Artal and J. D. Tardos, "Visual-Inertial Monocular SLAM with Map Reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[5] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1–17, 2018.

[6] M. Bosse, R. Zlot, and P. Flick, "Zebedee : Design of a spring-mounted 3-D range sensor with application to mobile mapping," *IEEE Transactions on Robotics*, vol. 28, no. October, pp. 1–15, 2012.

[7] J. Zhang and S. Singh, "LOAM : Lidar odometry and mapping in real-time," *Robotics: Science and Systems*, 2014.

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

[9] M. Bryson, M. Johnson-Roberson, and S. Sukkarieh, "Airborne smoothing and mapping using vision and inertial sensors," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2037–2042, 2009.

[10] J. Zhang and S. Singh, "Enabling aggressive motion estimation at low-drift and accurate mapping in real-time," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5051–5058, 2017.

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

[12] C. Le Gentil, T. Vidal-calleja, and S. Huang, "IN2LAMA : INertial Lidar Localisation And MApping," *IEEE International Conference on Robotics and Automation*, 2019.

[13] J. Delmerico and D. Scaramuzza, "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots," *IEEE International Conference on Robotics and Automation*, pp. 2502–2509, 2018.

[14] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," *Robotics: Science and Systems*, pp. 6–15, 2015.

[15] ——, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[16] K. Eckenhoff, P. Geneva, and G. Huang, "Closed-form preintegration methods for graph-based visualinertial navigation," *International Journal of Robotics Research*, vol. 38, no. 5, pp. 563–586, 2019.

[17] C. Le Gentil, T. Vidal-Calleja, and S. Huang, "3D Lidar-IMU Calibration based on Upsampled Preintegrated Measurements for Motion Distortion Correction," *IEEE International Conference on Robotics and Automation*, 2018.

[18] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-Time Batch Estimation using Temporal Basis Functions," *IEEE International Conference on Robotics and Automation*, pp. 2088–2095, 2012.

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

[20] C. E. Rasmussen, C. K. I. Williams, G. Processes, M. I. T. Press, and M. I. Jordan, *Gaussian Processes for Machine Learning*, 2006.

[21] M. Boyle, "The Integration of Angular Velocity," *Advances in Applied Clifford Algebras*, vol. 27, no. 3, pp. 2345–2374, 2017.

[22] V. M. Tereshkov, "A Simple Observer for Gyro and Accelerometer Biases in Land Navigation Systems," *Journal of Navigation*, vol. 68, no. 04, pp. 635–645, 2015.

[23] C. Le Gentil, T. Vidal-Calleja, and S. Huang, "IN2LAAMA: INertial Lidar Localisation Autocalibration And MApping," *Arxiv*, 2019. [Online]. Available: http://arxiv.org/abs/1905.09517