



RESEARCH ARTICLE

IMU as an input versus a measurement of the state in inertial-aided state estimation

Keenan Burnett¹ , Angela P. Schoellig² and Timothy D. Barfoot¹ 

¹University of Toronto Institute for Aerospace Studies, Toronto, ON, Canada

²Technical University of Munich, Munich, Germany

Corresponding author: Keenan Burnett; Email: keenan.burnett@robotics.utoronto.ca

Received: 9 March 2024; **Revised:** 6 November 2024; **Accepted:** 19 November 2024; **First published online:** 3 January 2025

Keywords: robot localization; continuous-time; Gaussian processes; inertial measurement units; lidar odometry

Abstract

Treating inertial measurement unit (IMU) measurements as inputs to a motion model and then preintegrating these measurements have almost become a de facto standard in many robotics applications. However, this approach has a few shortcomings. First, it conflates the IMU measurement noise with the underlying process noise. Second, it is unclear how the state will be propagated in the case of IMU measurement dropout. Third, it does not lend itself well to dealing with multiple high-rate sensors such as a lidar and an IMU or multiple asynchronous IMUs. In this paper, we compare treating an IMU as an input to a motion model against treating it as a measurement of the state in a continuous-time state estimation framework. We methodically compare the performance of these two approaches on a 1D simulation and show that they perform identically, assuming that each method's hyperparameters have been tuned on a training set. We also provide results for our continuous-time lidar-inertial odometry in simulation and on the Newer College Dataset. In simulation, our approach exceeds the performance of an imu-as-input baseline during highly aggressive motion. On the Newer College Dataset, we demonstrate state-of-the-art results. These results show that continuous-time techniques and the treatment of the IMU as a measurement of the state are promising areas of further research. Code for our lidar-inertial odometry can be found at: https://github.com/utiasASRL/steam_icp.

1. Introduction

Inertial measurement units (IMUs) are important sensors in the context of state estimation. A popular approach in robotics is to treat IMU measurements as inputs to a motion model and then to numerically integrate the motion model to form relative motion factors between pairs of estimation times in a process known as preintegration [1–3]. In this paper, we investigate the treatment of IMUs as a measurement of the state using continuous-time state estimation with a Gaussian process motion prior. We compare treating an IMU as an input versus a measurement on a simple 1D simulation problem. We then test our approach to lidar-inertial odometry using a simulated environment and compare to a baseline that represents the IMU-as-input approach. Finally, we provide experimental results for our lidar-inertial odometry on the Newer College Dataset.

Preintegration was initially devised as a method to avoid having to estimate the state at each measurement time in (sliding-window) batch trajectory estimation. We will show that by employing our continuous-time estimation techniques, we can achieve the same big-O complexity as classic preintegration, which is linear in the number of measurements. The contributions of this work are as follows:

- We provide a detailed comparison of treating an IMU as an input to a motion model versus as a measurement of the state on a 1D simulation problem. Such a comparison has not been previously presented in the literature.

- We show how to perform preintegration using heterogeneous factors (a combination of binary and unary factors) using continuous-time state estimation. To our knowledge, this has not been shown before in the literature.
- We present our novel approach to lidar-inertial odometry using a Singer prior which includes body-centric acceleration in the state. We provide experimental results on the Newer College Dataset and on a custom simulated dataset. On the Newer College Dataset, we demonstrate state-of-the-art performance.

2. Related work

Lupton and Sukkarieh [1] were the first to show that a temporal window of inertial measurements could be summarized using a single relative motion factor in a method known as preintegration. Forster et al. [2] then showed how to perform preintegration on the manifold $SO(3)$. Subsequently, Brossard et al. [3] demonstrated how to perform preintegration on the manifold of extended poses $SE_2(3)$, showing that this approach captures the uncertainty resulting from IMU measurements more consistently than $SO(3) \times \mathbb{R}^3$. All of these approaches treat the IMU as an input to a motion model. This approach has a few shortcomings. First, it conflates the IMU measurement noise with the underlying process noise. Second, it is unclear how the state and covariance should be propagated in the absence of IMU measurements. IMU measurement dropout is rare. However, it is worth considering the possibility in safety-critical applications. The third issue is that classic preintegration does not lend itself well to dealing with multiple high-rate sensors such as a lidar and an IMU or multiple asynchronous IMUs.

Previous work in continuous-time lidar-only odometry and lidar-inertial odometry include [4–9] all of which employed B-splines. In B-spline approaches, exact derivatives of the continuous-time trajectory can be computed allowing for unary factors to be created for each accelerometer and gyroscope measurement, removing the need for preintegration. However, the spacing of control points has a large impact on the smoothness of B-spline trajectories. Determining this spacing is an important engineering challenge in B-spline approaches. This can be avoided by working with Gaussian processes instead. For a detailed comparison between B-splines and Gaussian processes in continuous-time state estimation, we refer the reader to Johnson et al. [10].

Recently, Zheng and Zhu [11] demonstrated continuous-time lidar-inertial odometry using Gaussian processes where rotation is decoupled from translation. They use a white-noise-on-jerk (WNOJ) prior for position in a global frame, a white-noise-on-acceleration (WNOA) prior for rotation using a sequence of local Gaussian processes, and a white-noise-on-velocity prior for the IMU biases. Using this approach, they demonstrate competitive performance on the HILTI SLAM benchmark [12]. One consequence of estimating position in a global frame is that the power spectral density matrix \mathbf{Q} of the prior must typically be isotropic, whereas a body-centric approach allows for lateral-longitudinal-vertical dimensions to be weighted differently. In addition, as was shown by Brossard et al. [3], decoupling rotation from translation does not capture the uncertainty resulting from IMU measurements as accurately as keeping them coupled. However, one clear advantage of their approach is that all parts of the state are directly observable by the measurements, whereas in our approach angular acceleration is not directly observable.

In our prior work, we demonstrated continuous-time lidar-only odometry [13] using a WNOA motion prior. Lidar-only odometry using a WNOJ prior [14] and a Singer prior [15] have also been previously demonstrated. In this work, we choose to work with the Singer prior, which includes body-centric acceleration in the state. By including acceleration in the state, we can treat gyroscope and accelerometer measurements as measurements of the state rather than as inputs to a motion model.

Another approach based on Gaussian processes is that of Le Gentil and Vidal-Calleja [16, 17]. They employ six independent Gaussian processes, three for acceleration in a global frame, and three for angular velocity. They optimize for the state at several inducing points given a set of IMU measurements over a

preintegration window. They then analytically integrate these Gaussian processes to obtain preintegrated measurements that can be queried at any time of interest.

Prior lidar-inertial odometry methods include [18–21]. For a recent survey and comparison of open-source lidar-only and lidar-inertial odometry approaches, we refer the reader to [22]. For a more detailed literature review of lidar odometry, lidar-inertial odometry, and continuous-time state estimation, we refer the reader to our prior work [23]. For a recent survey on continuous-time state estimation, we refer the reader to Talbot et al. [24]. Extrinsic calibration of an IMU and calibration of the temporal offset between the IMU timestamps and the other sensors are both important areas of research. Recent works in this area include [25, 26]. By working with an existing dataset where extrinsic calibration and temporal synchronization has been taken care of, we can focus on the task of localization.

3. IMU as an input versus a measurement

In this section, we investigate an approach where IMU measurements are treated as direct measurements of the state using a continuous-time motion prior. We compare the performance of these two approaches on a simulated toy problem where we estimate the position and velocity of a 1D robot given noisy measurements of position and acceleration. Position measurements are acquired at a lower rate (10 Hz), while the acceleration measurements are acquired at a higher rate (100 Hz). For both approaches, our goal will be to reduce the number of states being estimated through preintegration.

3.1. IMU as an input

As a baseline, we consider treating IMU measurements as inputs to a discrete motion model:

$$\underbrace{\begin{bmatrix} \mathbf{p}_k \\ \dot{\mathbf{p}}_k \end{bmatrix}}_{\mathbf{x}_k} = \underbrace{\begin{bmatrix} \mathbf{1} & \Delta t_k \mathbf{1} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}}_{\Phi(t_k, t_{k-1})} \underbrace{\begin{bmatrix} \mathbf{p}_{k-1} \\ \dot{\mathbf{p}}_{k-1} \end{bmatrix}}_{\mathbf{x}_{k-1}} + \underbrace{\begin{bmatrix} \frac{1}{2} \Delta t_k^2 \\ \Delta t_k \end{bmatrix}}_{\mathbf{B}_k} \mathbf{u}_k, \quad (1)$$

where $\Delta t_k = t_k - t_{k-1}$, $\Phi(t_k, t_{k-1})$ is the transition function, and \mathbf{u}_k are acceleration measurements. A preintegration window is then defined between two endpoints, (t_{k-1}, t_k) , which includes times τ_0, \dots, τ_J . Following the approach of [1, 2], the preintegrated measurements $\Delta \mathbf{x}_{k,k-1}$ are computed as:

$$\Delta \mathbf{x}_{k,k-1} = \sum_{n=1}^J \Phi(\tau_J, \tau_n) \mathbf{B}_n \mathbf{u}_n. \quad (2)$$

These preintegrated measurements can be used to replace the acceleration measurements with a single relative motion factor between two endpoints:

$$J_{v,k} = \frac{1}{2} \mathbf{e}_k^T \Sigma_k^{-1} \mathbf{e}_k, \quad (3a)$$

$$\mathbf{e}_k = \mathbf{x}_k - \Phi(t_k, t_{k-1}) \mathbf{x}_{k-1} - \Delta \mathbf{x}_{k,k-1}, \quad (3b)$$

where

$$\Sigma_k = \sum_{n=1}^J \Phi(\tau_J, \tau_n) \mathbf{B}_n \mathbf{Q}_n \mathbf{B}_n^T \Phi(\tau_J, \tau_n)^T, \quad (4)$$

and \mathbf{Q}_n is the covariance of the acceleration input $\mathbf{u}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_n)$. Note that in this approach, uncertainty is propagated using the covariance on the acceleration input, \mathbf{Q}_n , which conflates IMU measurement noise and the underlying process noise. If the acceleration measurements were to drop out suddenly, it is unclear how the state and covariance should be propagated using this approach. It can be shown that preintegration is mathematically equivalent to marginalizing out the states between \mathbf{x}_{k-1} and \mathbf{x}_k .

The overall objective function that we seek to minimize is then

$$J(\mathbf{x}) = \sum_{k=0}^K (J_{v,k}(\mathbf{x}) + J_{y,k}(\mathbf{x})), \quad (5)$$

where

$$J_{y,k} = \frac{1}{2} (\mathbf{y}_k - \mathbf{C}_k \mathbf{x}_k)^T \mathbf{R}_k^{-1} (\mathbf{y}_k - \mathbf{C}_k \mathbf{x}_k) \quad (6)$$

are the measurement factors, and \mathbf{R}_k is the associated measurement covariance.

3.2. Continuous-time state estimation

In order to incorporate potentially asynchronous position and acceleration measurements, we propose to carry out continuous-time trajectory estimation as exactly sparse Gaussian process regression [27–29]. We consider systems with a Gaussian process (GP) prior and a linear measurement model:

$$\mathbf{x}(t) \sim \mathcal{GP}(\check{\mathbf{x}}(t), \check{\mathbf{P}}(t, t')), \quad (7a)$$

$$\mathbf{y}_k = \mathbf{C}_k \mathbf{x}(t_k) + \mathbf{n}_k, \quad (7b)$$

where $\mathbf{x}(t)$ is the state, $\check{\mathbf{x}}(t)$ is the mean function, $\check{\mathbf{P}}(t, t')$ is the covariance function, and \mathbf{y}_k are measurements corrupted by zero-mean Gaussian noise $\mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$. In this section, we restrict our attention to a class of GP priors resulting from linear time-invariant (LTI) stochastic differential equations (SDEs) of the form:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{L}\mathbf{w}(t), \quad (8)$$

$$\mathbf{w}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}\delta(t - t')),$$

where $\mathbf{w}(t)$ is a white-noise Gaussian process, \mathbf{Q} is a power spectral density matrix, and $\mathbf{u}(t)$ is a known exogenous input. The general solution to this differential equation is

$$\mathbf{x}(t) = \Phi(t, t_0)\mathbf{x}(t_0) + \int_{t_0}^t \Phi(t, s)(\mathbf{B}\mathbf{u}(s) + \mathbf{L}\mathbf{w}(s))ds, \quad (9)$$

where $\Phi(t, s) = \exp(\mathbf{A}(t - s))$ is the transition function. The mean function is

$$\check{\mathbf{x}}(t) = E[\mathbf{x}(t)] = \Phi(t, t_0)\check{\mathbf{x}}_0 + \int_{t_0}^t \Phi(t, s)\mathbf{B}\mathbf{u}(s)ds. \quad (10)$$

Over a sequence of estimation times, $t_0 < t_1 < \dots < t_K$, the mean function can be written as:

$$\check{\mathbf{x}}(t_k) = \Phi(t_k, t_0)\check{\mathbf{x}}_0 + \sum_{n=1}^k \Phi(t_k, t_n)\mathbf{B}_n\mathbf{u}_n, \quad (11)$$

assuming piecewise-constant input \mathbf{u}_n . This can be rewritten in a lifted form as:

$$\check{\mathbf{x}} = \mathbf{A}\mathbf{B}\mathbf{u}, \quad (12)$$

where \mathbf{A} is the lifted lower-triangular transition matrix, the inverse of which is

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{1} & & & \\ -\Phi(t_1, t_0) & \ddots & & \\ & \ddots & \mathbf{1} & \\ & & -\Phi(t_K, t_{K-1}) & \mathbf{1} \end{bmatrix}, \quad (13)$$

$\mathbf{B} = \text{diag}(\mathbf{1}, \mathbf{B}_1, \dots, \mathbf{B}_K)$, and $\mathbf{u} = [\tilde{\mathbf{x}}_0^T \mathbf{u}_1^T \dots \mathbf{u}_K^T]^T$. See [29] for further details on the formulations above. The covariance function is then

$$\begin{aligned} \check{\mathbf{P}}(t, t') &= E[(\mathbf{x}(t) - E[\mathbf{x}(t)])(\mathbf{x}(t') - E[\mathbf{x}(t')])^T] \\ &= \Phi(t, t_0) \check{\mathbf{P}}_0 \Phi(t', t_0)^T + \int_{t_0}^{\min(t, t')} \Phi(t, s) \mathbf{L} \mathbf{Q} \mathbf{L}^T \Phi(t', s)^T ds. \end{aligned} \quad (14)$$

The covariance can also be rewritten in a lifted form using the same set of estimation times as above:

$$\check{\mathbf{P}} = \mathbf{A} \mathbf{Q} \mathbf{A}^T, \quad (15)$$

where $\mathbf{Q} = \text{diag}(\check{\mathbf{P}}_0, \mathbf{Q}_1, \dots, \mathbf{Q}_K)$, and

$$\mathbf{Q}_k = \int_0^{\Delta t_k} \exp(\mathbf{A}(\Delta t_k - s)) \mathbf{L} \mathbf{Q} \mathbf{L}^T \exp(\mathbf{A}(\Delta t_k - s))^T ds. \quad (16)$$

Our prior over the entire trajectory can then be written as:

$$\mathbf{x} \sim \mathcal{N}(\check{\mathbf{x}}, \check{\mathbf{P}}), \quad (17)$$

where $\check{\mathbf{P}}$ is the kernel matrix. Note that the inverse kernel matrix $\check{\mathbf{P}}^{-1}$ is block-tridiagonal thanks to the Markovian nature of the state. This sparsity property also holds for linear time-varying (LTV) SDEs, provided that they are also Markovian [28]. The exact sparsity of $\check{\mathbf{P}}^{-1}$ is what allows us to perform efficient Gaussian process regression. This fact can be observed more easily by inspecting the following linear system of equations:

$$\underbrace{(\check{\mathbf{P}}^{-1} + \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C})}_{\hat{\mathbf{P}}^{-1}} \hat{\mathbf{x}} = \mathbf{A}^{-T} \mathbf{Q}^{-1} \mathbf{B} \mathbf{u} + \mathbf{C}^T \mathbf{R}^{-1} \mathbf{y}, \quad (18)$$

where the Hessian is on the left-hand side, $\hat{\mathbf{P}}^{-1}$ is block-tridiagonal since $\check{\mathbf{P}}^{-1}$ is block-tridiagonal, and $\mathbf{C}^T \mathbf{R}^{-1} \mathbf{C}$ is block-diagonal. Thus, this linear system of equations can be solved in $O(K)$ time using a sparse Cholesky solver. The exact sparsity of $\hat{\mathbf{P}}^{-1}$ also enables us to perform efficient Gaussian process interpolation. The standard GP interpolation formulas are given by:

$$\hat{\mathbf{x}}(\tau) = \check{\mathbf{x}}(\tau) + \check{\mathbf{P}}(\tau) \check{\mathbf{P}}^{-1} (\hat{\mathbf{x}} - \check{\mathbf{x}}), \quad (19a)$$

$$\hat{\mathbf{P}}(\tau, \tau) = \check{\mathbf{P}}(\tau, \tau) + \check{\mathbf{P}}(\tau) \check{\mathbf{P}}^{-1} (\hat{\mathbf{P}} - \check{\mathbf{P}}) \check{\mathbf{P}}^{-T} \check{\mathbf{P}}(\tau)^T, \quad (19b)$$

where

$$\check{\mathbf{P}}(\tau) = \begin{bmatrix} \check{\mathbf{P}}(\tau, t_0) & \check{\mathbf{P}}(\tau, t_1) & \dots & \check{\mathbf{P}}(\tau, t_K) \end{bmatrix}. \quad (20)$$

The key to performing efficient interpolation relies on the sparsity of

$$\check{\mathbf{P}}(\tau) \check{\mathbf{P}}^{-1} = \begin{bmatrix} \mathbf{0} & \dots & \mathbf{0} & \Lambda(\tau) & \Psi(\tau) & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}, \quad (21)$$

where

$$\Psi(\tau) = \mathbf{Q}_\tau \Phi(t_{k+1}, \tau)^T \mathbf{Q}_{k+1}^{-1}, \quad (22a)$$

$$\Lambda(\tau) = \Phi(\tau, t_k) - \Psi(\tau) \Phi(t_{k+1}, t_k), \quad (22b)$$

are the only nonzero block columns at indices $k+1$ and k , respectively. Thus, each interpolation query of the posterior trajectory is an $O(1)$ operation.

3.3. A generalization to preintegration

In Section 3.1, we showed how to perform preintegration when considering acceleration measurements as inputs to a motion model following closely from [1, 2]. In this section, we generalize the concept of

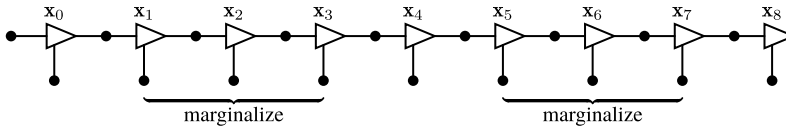


Figure 1. In this factor graph, we consider a case where we would like to marginalize several states out of the full Bayesian posterior. The triangles represent states and the black dots represent factors. This factor graph could potentially correspond to doing continuous-time state estimation with binary motion prior factors, unary measurement factors, and a unary prior factor on the initial state \mathbf{x}_0 .

preintegration to support heterogeneous factors (a combination of binary factors and unary factors). An example factor graph is shown in Figure 1. As a motivating example, consider having measurements of position such as from a GPS and measurements of acceleration coming from an accelerometer. These are unary measurement factors, called such because they only involve a single state. The binary factors here are motion prior factors derived from our Gaussian process motion prior, called binary because they are between two states. In classic preintegration, the only factors that are preintegrated are binary factors. Here, we show that we can simply use the formulas for querying a Gaussian process posterior at the endpoints of the preintegration window to form a preintegration factor that summarizes all the measurements contained therein. First, we consider the joint density of the state at a set of query times ($\tau_0 < \tau_1 < \dots < \tau_J$) and the measurements:

$$p\left(\begin{bmatrix} \mathbf{x}_\tau \\ \mathbf{y} \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \check{\mathbf{x}}_\tau \\ \check{\mathbf{C}}\check{\mathbf{x}}_\tau \end{bmatrix}, \begin{bmatrix} \check{\mathbf{P}}_{\tau,\tau} & \check{\mathbf{P}}_\tau \mathbf{C}^T \\ \mathbf{C}\check{\mathbf{P}}_\tau^T & \mathbf{R} + \mathbf{C}\check{\mathbf{P}}_\tau \mathbf{C}^T \end{bmatrix}\right). \quad (23)$$

We then perform the usual factoring using a Schur complement to obtain the posterior:

$$p(\mathbf{x}_\tau | \mathbf{y}) = \mathcal{N}\left(\check{\mathbf{x}}_\tau + \check{\mathbf{P}}_\tau \mathbf{C}^T (\mathbf{C}\check{\mathbf{P}}_\tau \mathbf{C}^T + \mathbf{R})^{-1} (\mathbf{y} - \mathbf{C}\check{\mathbf{x}}), \check{\mathbf{P}}_{\tau,\tau} - \check{\mathbf{P}}_\tau \mathbf{C}^T (\mathbf{C}\check{\mathbf{P}}_\tau \mathbf{C}^T + \mathbf{R})^{-1} \mathbf{C}\check{\mathbf{P}}_\tau^T\right), \quad (24)$$

where we obtain expressions for $\hat{\mathbf{x}}_\tau$ and $\hat{\mathbf{P}}_{\tau,\tau}$. We rearrange this further by inserting $\check{\mathbf{P}}^{-1}\check{\mathbf{P}}$ after the first instance of $\check{\mathbf{P}}_\tau$ and by applying the Sherman–Morrison–Woodbury identities to obtain

$$\hat{\mathbf{x}}_\tau = \check{\mathbf{x}}_\tau + \check{\mathbf{P}}_\tau \check{\mathbf{P}}^{-1} (\check{\mathbf{P}}^{-1} + \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C})^{-1} \mathbf{C}^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{C}\check{\mathbf{x}}), \quad (25a)$$

$$\hat{\mathbf{P}}_{\tau,\tau} = \check{\mathbf{P}}_{\tau,\tau} - \check{\mathbf{P}}_\tau \check{\mathbf{P}}^{-1} (\check{\mathbf{P}}^{-1} + \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C})^{-1} \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C}\check{\mathbf{P}}_\tau^T, \quad (25b)$$

where we note that $(\check{\mathbf{P}}^{-1} + \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C})$ is block-tridiagonal, and so the product $(\check{\mathbf{P}}^{-1} + \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C})^{-1} \mathbf{C}^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{C}\check{\mathbf{x}})$ can be evaluated in $O(K)$ time using a sparse Cholesky solver. When we encounter products resembling $\mathbf{A}^{-1}\mathbf{b}$ where \mathbf{A} is block-tridiagonal, we can instead solve $\mathbf{A}\mathbf{z} = \mathbf{b}$ for \mathbf{z} using an efficient solver that takes advantage of the sparsity of \mathbf{A} . Finally, the product $\check{\mathbf{P}}_\tau \check{\mathbf{P}}^{-1}$ is quite sparse, having only two nonzero block columns per block row as shown earlier in (21). It follows that $\hat{\mathbf{x}}_\tau$ and $\hat{\mathbf{P}}_{\tau,\tau}$ can be computed in time that scales linearly with the number of measurements. Now, we consider the case where the query times consist of the beginning and end of a preintegration window, (t_k, t_{k+1}) . The queried mean and covariance can be treated as pseudomeasurements summarizing the measurements contained in the preintegration window. We adjust our notation to make it clear that these are now being treated as measurements by using $\tilde{\mathbf{x}}$ instead of $\hat{\mathbf{x}}$. What we obtain is a joint Gaussian factor for the states at times t_k and t_{k+1} :

$$J = \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \tilde{\mathbf{x}}_k \\ \mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1} \end{bmatrix}^T \begin{bmatrix} \tilde{\mathbf{P}}_{k,k} & \tilde{\mathbf{P}}_{k,k+1} \\ \tilde{\mathbf{P}}_{k,k+1} & \tilde{\mathbf{P}}_{k+1,k+1} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{x}_k - \tilde{\mathbf{x}}_k \\ \mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1} \end{bmatrix}. \quad (26)$$

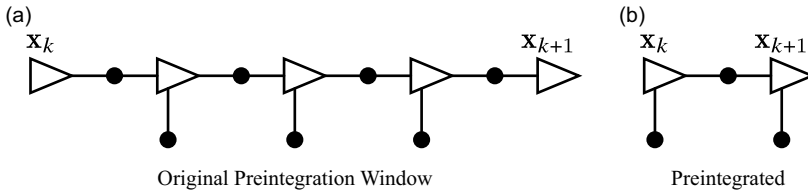


Figure 2. This figure depicts the results of our preintegration, which can incorporate heterogeneous factors. The resulting joint Gaussian factor in (26) can be thought of as two unary factors, one each for \mathbf{x}_k and \mathbf{x}_{k+1} , and an additional binary factor between \mathbf{x}_k and \mathbf{x}_{k+1} .

A diagram depicting how this affects the resulting factor graph is shown in Figure 2. Using this approach, we can ‘preintegrate’ heterogeneous factors between pairs of states. One clear advantage of this approach is that it offers a tidy method for bookkeeping measurement costs and uncertainties. In the supplementary materials, we provide an alternative formulation using a Schur complement with the same linear time complexity. Indeed, marginalization with a Schur complement is equivalent to the presented marginalization approach using a Gaussian process. However, the Gaussian process still serves a useful purpose in creating motion prior factors. Furthermore, the Gaussian process provides methods for interpolating the posterior.

It is unclear how to extend this marginalization approach to $SE(3)$ due to our choice to approximate $SE(3)$ trajectories using sequences of local Gaussian processes [28]. It is possible that this marginalization approach could be applied using a global GP formulation such as the one presented by Le Gentil and Vidal-Calleja [17]. However, their approach must contend with rotational singularities. We leave the extension to $SE(3)$ as an area of future work. In our implementation of lidar-inertial odometry, we instead use the posterior Gaussian process interpolation formula as in (19) to form continuous-time measurement factors. This is actually an approximation, as it is not exactly the same as marginalization. However, we have found the interpolation approach to work well in practice. Furthermore, the interpolation approach lends itself quite easily to parallelization enabling a highly efficient implementation.

3.4. IMU as a measurement

In our proposed approach, we treat IMU measurements as direct measurements of the state within a continuous-time estimation framework. For the 1D toy problem, we chose to use a Singer prior, which is defined by the following LTI SDE:

$$\begin{aligned} \ddot{\mathbf{p}}(t) &= -\alpha \ddot{\mathbf{p}}(t) + \mathbf{w}(t), \\ \mathbf{w}(t) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_c \delta(t - t')), \end{aligned} \quad (27)$$

where $\mathbf{w}(t)$ is a white-noise Gaussian process and $\mathbf{Q}_c = 2\alpha\sigma^2$ is the power spectral density matrix [30]. By varying α and σ^2 , the Singer prior can model motion priors ranging from WNOA ($\alpha \rightarrow \infty, \tilde{\sigma}^2 = \alpha^2\sigma^2$) to WNOJ ($\alpha \rightarrow \mathbf{0}$). The discrete-time motion model is given by:

$$\mathbf{x}_k = \Phi(t_k, t_{k-1})\mathbf{x}_{k-1} + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k), \quad (28)$$

where \mathbf{w}_k is the process noise, $\mathbf{x}_k = [\mathbf{p}_k^T \dot{\mathbf{p}}_k^T \ddot{\mathbf{p}}_k^T]^T$, $\Phi(t_k, t_{k-1})$ is the state transition function, and \mathbf{Q}_k is the discrete-time covariance. Expressions for $\Phi(t_k, t_{k-1})$ and \mathbf{Q}_k for the Singer prior are provided by Wong et al. [30] and are repeated in the supplementary materials. The binary motion prior factors are given by:

$$J_{v,k} = \frac{1}{2} \mathbf{e}_k^T \mathbf{Q}_k^{-1} \mathbf{e}_k, \quad (29a)$$

$$\mathbf{e}_k = \mathbf{x}_k - \Phi(t_k, t_{k-1})\mathbf{x}_{k-1}. \quad (29b)$$

The unary measurement factors have the same form as in (6) except that now accelerations are treated as direct measurements of the state. In addition, the overall objective is the same as in (5). We arrive at the linear system of equations from (18). By default, this approach would require that we estimate the state at each measurement time. In order to reduce the size of the state space, as in Section 3.1, we build preintegrated factors using the approach presented in Section 3.3 in order to bundle together both the unary measurement factors as well as the binary motion prior factors into a single factor. In the exact same fashion as the IMU-as-input approach, we preintegrate between pairs of states associated with the low-rate measurement times.

3.5. Simulation results

In this section, we compare two different approaches to combining high-rate acceleration measurements with low-rate position measurements. We refer to these two different approaches as IMU-as-input and IMU-as-measurement. The comparison is conducted on a toy 1D simulation problem. We sample trajectories from a GP motion prior as defined in (17) by using standard methods for sampling from a multidimensional Gaussian [29].

In our first simulation, we sample trajectories from a WNOJ motion prior with $Q_c = 1.0$. Figure 3 provides a qualitative comparison of the IMU-as-input and IMU-as-measurement approaches for a single sampled simulation trajectory. Note that the performance of the two estimators including their $3\text{-}\sigma$ confidence bounds appears to be nearly identical. Figure 4 depicts 1000 trajectories sampled from this motion prior. In order to simulate noisy sensors, we also corrupt both position and acceleration measurements using zero-mean Gaussian noise where $y_{\text{pos},k} = [1 \ 0 \ 0] \mathbf{x}_k + n_{\text{pos},k}$, $n_{\text{pos},k} \sim \mathcal{N}(0, \sigma_{\text{pos}}^2)$, $y_{\text{acc},k} = [0 \ 0 \ 1] \mathbf{x}_k + n_{\text{acc},k}$, and $n_{\text{acc},k} \sim \mathcal{N}(0, \sigma_{\text{acc}}^2)$. In the simulation, we set $\sigma_{\text{pos}} = 0.01m$ and $\sigma_{\text{acc}} = 0.01m/s^2$.

Both IMU-as-input and IMU-as-measurement have parameters that need to be tuned on the dataset. For this purpose, we use a separate training set of 100 sampled trajectories. For the IMU-as-input approach, we learn the covariance of the acceleration input Q_k using maximum likelihood over the training set where we have the benefit of using noiseless ground-truth states in simulation. The objective that we seek to minimize is

$$J = \frac{1}{2} \ln |\check{\mathbf{P}}| + \frac{1}{2T} \sum_{t=1}^T (\mathbf{x}_t - \check{\mathbf{x}})^T \check{\mathbf{P}}^{-1} (\mathbf{x}_t - \check{\mathbf{x}}), \quad (30)$$

where \mathbf{x}_t are validation set trajectories, $\check{\mathbf{x}} = \mathbf{A}\mathbf{v}$ is the full trajectory prior for the IMU-as-input method, $\check{\mathbf{P}} = \mathbf{A}\mathbf{Q}\mathbf{A}^T$ is the prior covariance over the whole trajectory, \mathbf{A} is the lifted transition matrix as in (13), $\mathbf{v} = [\check{\mathbf{x}}_0^T \ \Delta \mathbf{x}_{1,0}^T \ \cdots \ \Delta \mathbf{x}_{K,K-1}^T]^T$, and $\mathbf{Q} = \text{diag}(\check{\mathbf{P}}_0, \boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K)$.

Using a numerical optimizer, we found that $Q_k \approx 0.00338$ given a WNOJ motion prior with $Q_c = 1.0$, and acceleration measurement noise of $\sigma_{\text{acc}}^2 = 0.0001m^2/s^4$. Note that the input covariance Q_k is clearly much greater than the simulated noise on the acceleration measurements σ_{acc}^2 . This is because, for the IMU-as-input approach, the covariance on the acceleration input Q_k is conflating two sources of noise, the IMU measurement noise and the underlying process noise. It also means that Q_k has to be trained and adapted to new datasets in order to maintain a consistent estimator.

For the IMU-as-measurement approach, we need to train the parameters of our Gaussian process (GP) motion prior. In order to highlight the versatility of our proposed approach, we employ a Singer motion prior [30], which has the capacity to model priors ranging from WNOA to WNOJ. The Singer prior is parametrized by an inverse length scale matrix $\boldsymbol{\alpha}$ and a variance σ^2 , both of which are diagonal. For values of $\boldsymbol{\alpha}$ close to zero, numerical optimizers encounter difficulties due to numerical instabilities of \mathbf{Q}_k and its Jacobians. Instead, we derive the analytical gradients to learn $\{\boldsymbol{\alpha}, \sigma^2\}$ using gradient descent following the approach presented by Wong et al. [30]. The objective that we seek to minimize is

$$J = \frac{1}{2} \sum_{t=1}^T \sum_{k=1}^K (\mathbf{e}_{k,t}^T \mathbf{Q}_{k,t}^{-1} \mathbf{e}_{k,t} + \ln |\mathbf{Q}_{k,t}|), \quad (31)$$

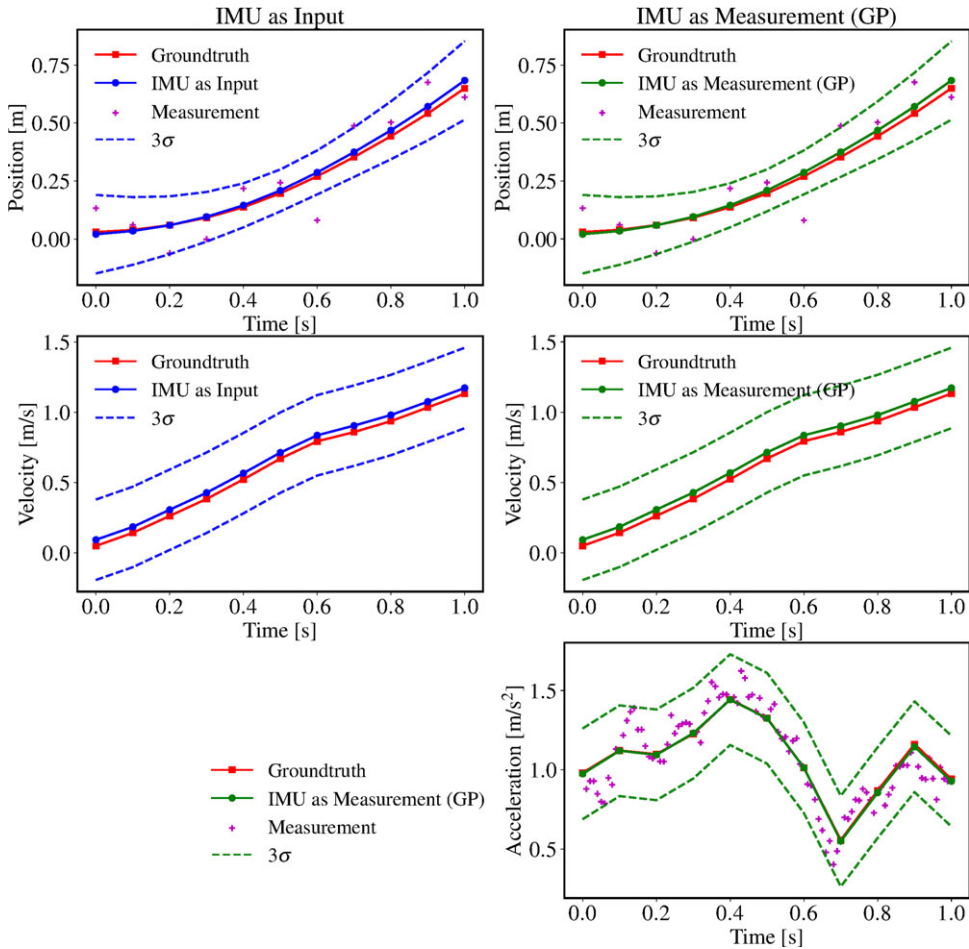


Figure 3. The estimated trajectories of IMU-as-input and IMU-as-measurement are plotted alongside the ground-truth trajectory, which is sampled from white-noise-on-jerk motion prior with $Q_c = 1.0$. Both methods were pretrained on a hold-out validation set of simulated trajectories.

where both $\mathbf{e}_{k,t}$ and $\mathbf{Q}_{k,t}$ are functions of the Singer prior parameters $\{\boldsymbol{\alpha}, \boldsymbol{\sigma}^2\}$. Further details on the analytical gradients are provided in the supplementary materials. Note that the approach of Wong et al. [30] supports learning the parameters of the Singer prior even with noisy ground truth; however, it requires that we first estimate the measurement covariances and then keep them fixed during the optimization. In order to learn both the GP parameters and the measurement covariances simultaneously, Wong et al. leverage exactly sparse Gaussian variational inference [15].

Figure 5 shows the results of our first simulation experiment with the WNOJ prior. Each row in Figure 5 is a box plot of a metric computed independently for each of the 1000 simulated trajectories. The blue boxes represent the interquartile range, the red lines are the medians, the whiskers correspond to the 2.5 and 97.5 percentiles, and the red dots denote outliers (data points beyond the whiskers). The black dashed lines in the first, third, and fourth rows corresponds to the target value that is 0 for the mean error and 1 for the normalized estimation error squared (NEES). Underneath each box plot, we also compute the mean value of the metrics across all data points.

In the first row of Figure 5, we can see that the mean error in both position and velocity is close to zero for both estimators. The gray lines in the first row denote a 95% two-sided confidence interval, a statistical test to check that the estimators are unbiased. We expect to see the whiskers of the box plots

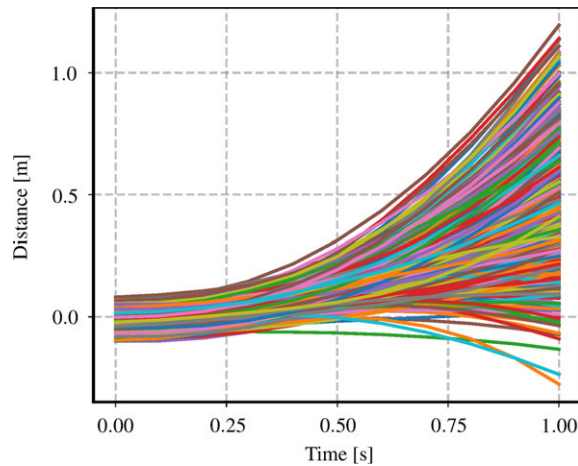


Figure 4. This figure depicts 1000 simulated trajectories sampled from a white-noise-on-jerk (WNOJ) prior where $\check{\mathbf{x}}_0 = [0.0 \ 0.0 \ 1.0]^T$, $\check{\mathbf{P}}_0 = \text{diag}(0.001, 0.001, 0.001)$, $Q_c = 1.0$.

lie within the 95% confidence interval in order to confirm that the estimator is unbiased, which is the case.

The third row displays a commonly used method for computing the NEES. This method uses the marginals of the posterior covariance and relies on the ergodic hypothesis in order to treat the error from each timestep as being independent. In this case, we compute the marginal covariance at each timestep for position and velocity only so that the results of the two estimators can be compared directly. The gray lines denote a 95% chi-squared bound, a statistical test for checking that the estimators are consistent. Interestingly, we observe that neither estimator passes the statistical test in this case, even though the mean and median NEES are close to 1. It appears that, in this case, the ergodic hypothesis is not valid.

In the fourth row, we present an alternative formulation of the NEES that uses the full posterior covariance over the entire trajectory. In this case, we are satisfied to find that both estimators pass the statistical test for confirming that they are consistent. The main difference between this version of the NEES and the previous one is that we have retained the cross-covariance terms between timesteps.

In summary, we observe that the two approaches achieve nearly identical performance. Both estimators are unbiased and consistent so long as their parameters are trained on a training set.

Figures 6, 7 depict the results of our second experiment where the ground-truth trajectories are sampled from a Singer prior with $\alpha = 10.0$, $\sigma^2 = 1.0$. This large value of α is intended to approximate a WNOA prior. Our results show that both estimators are capable of adapting to a dataset with a different underlying motion prior while remaining unbiased and consistent.

3.6. Discussion

As mentioned previously, the big-O time complexity of classic preintegration and our approach are the same. In practice, our approach is slightly slower but not by much. Using a modern CPU, either approach can be considered real-time capable. Note that the number of preintegration windows could be adjusted and each preintegration window could be computed in parallel to make the approach more efficient. In this way, we could parallelize the solving of some estimation problems. We are motivated by sensor configurations that cannot be easily handled by classic preintegration such as multiple asynchronous high-rate sensors. This could include a lidar and an IMU or multiple asynchronous IMUs.

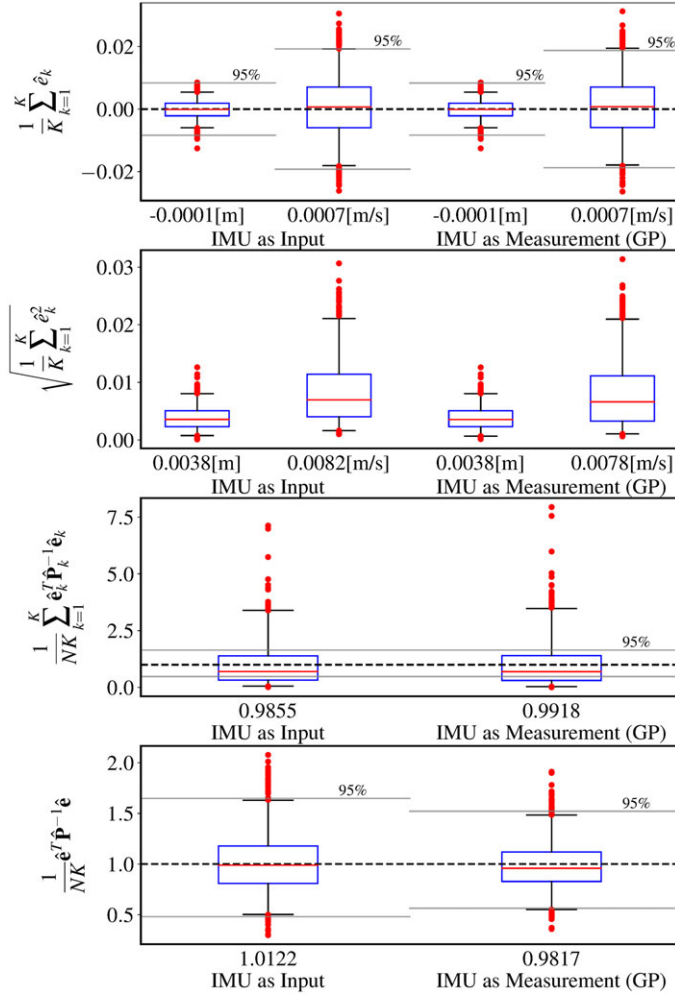


Figure 5. This figure compares the performance of the baseline IMU-as-input approach versus our proposed IMU-as-measurement approach that leverages a Gaussian process (GP) motion prior. The ground-truth trajectories are sampled from a white-noise-on-jerk (WNOJ) prior as shown in Figure 4. The IMU input covariance Q_k for the IMU-as-input method was trained on a validation set, with a value of 0.00338. The parameters of our proposed method was also trained on the same validation set, with resulting values of $\sigma^2 = 1.0069$ and $\alpha = 0.0$. R_{pos} for both methods was chosen to match the simulated noise added to the position measurements. Similarly, R_{acc} for the IMU-as-measurement approach was set to match the simulated noise added to the acceleration measurements. In the bottom row, the chi-squared bounds have a different size because the dimension of the state in IMU-as-measurement is greater (it includes acceleration), and so the dimension of the chi-squared distribution increases, resulting in a tighter chi-squared bound.

4. Lidar-inertial odometry

Our lidar-inertial odometry is implemented as sliding-window batch trajectory estimation. The factor graph corresponding to our approach is depicted in Figure 8. The state $\mathbf{x}(t) = \{\mathbf{T}(t), \boldsymbol{\omega}(t), \dot{\boldsymbol{\omega}}(t), \mathbf{b}(t)\}$ consists of the $SE(3)$ pose $\mathbf{T}_v(t)$, the body-centric velocity $\boldsymbol{\omega}_v^{vi}(t)$, the body-centric acceleration $\dot{\boldsymbol{\omega}}_v^{vi}(t)$, and the IMU biases $\mathbf{b}(t)$. $\boldsymbol{\omega}_v^{vi}$ is a 6×1 vector containing the body-centric linear velocity \mathbf{v}_v^{vi} and angular velocity $\boldsymbol{\omega}_v^{vi}$. We approximate the $SE(3)$ trajectory using a sequence of local Gaussian processes as in

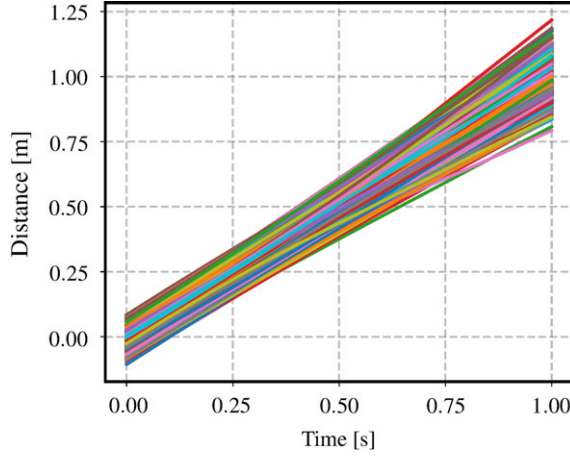


Figure 6. This figure depicts 1000 simulated trajectories sampled from a singer prior where $\check{\mathbf{x}}_0 = [0.0 \ 1.0 \ 0.0]^T$, $\check{\mathbf{P}}_0 = \text{diag}(0.001, 0.001, 0.001)$, $\alpha = 10.0$, and $\sigma^2 = 1.0$. A large value of α is intended to approximate a white-noise-on-acceleration (WNOA) prior.

[28]. Between pairs of estimation times, the local variable $\xi_k(t)$ is defined as:

$$\xi_k(t) = \ln(\mathbf{T}(t)\mathbf{T}(t_k)^{-1})^\vee. \quad (32)$$

We use (32) and the following to convert between global and local variables:

$$\dot{\xi}_k(t) = \mathcal{J}(\xi_k(t))^{-1} \dot{\mathbf{w}}(t), \quad (33)$$

$$\ddot{\xi}_k(t) \approx -\frac{1}{2}(\mathcal{J}(\xi_k(t))^{-1} \dot{\mathbf{w}}(t))^\top \dot{\mathbf{w}}(t) + \mathcal{J}(\xi_k(t))^{-1} \ddot{\mathbf{w}}(t), \quad (34)$$

where the approximation for $\ddot{\xi}_k(t)$ was originally derived by Tang et al. [14]. We use a Singer prior, introduced in [30], which is defined by the following Gaussian process:

$$\ddot{\xi}_k(t) \sim \mathcal{GP}(\mathbf{0}, \sigma^2 \exp(-\ell^{-1}|t - t'|)), \quad (35)$$

and which can equivalently be represented using the following LTI SDE:

$$\dot{\boldsymbol{\gamma}}_k(t) = \mathbf{A}\boldsymbol{\gamma}_k(t) + \mathbf{L}\mathbf{w}(t), \quad (36)$$

where

$$\begin{aligned} \mathbf{w}(t) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_c \delta(t - t')), \\ \boldsymbol{\gamma}_k(t) &= \begin{bmatrix} \xi_k(t) \\ \dot{\xi}_k(t) \\ \ddot{\xi}_k(t) \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\alpha \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}, \end{aligned}$$

σ^2 is a variance, ℓ is a length scale, $\alpha = \ell^{-1}$, and $\mathbf{w}(t)$ is a white-noise Gaussian process where $\mathbf{Q}_c = 2\alpha\sigma^2$ is the associated power spectral density matrix. (36) can be stochastically integrated to arrive at a local Gaussian process:

$$\boldsymbol{\gamma}_k(t) \sim \mathcal{GP}(\boldsymbol{\Phi}(t, t_k) \check{\boldsymbol{\gamma}}_k(t_k), \boldsymbol{\Phi}(t, t_k) \check{\mathbf{P}}(t_k) \boldsymbol{\Phi}(t, t_k)^T + \mathbf{Q}_t), \quad (37)$$

where the formulation for the transition function $\boldsymbol{\Phi}(t, t_k)$ and the covariance \mathbf{Q}_t can be found in [30]. In order to convert our continuous-time formulation into a factor graph, we build a sequence of motion

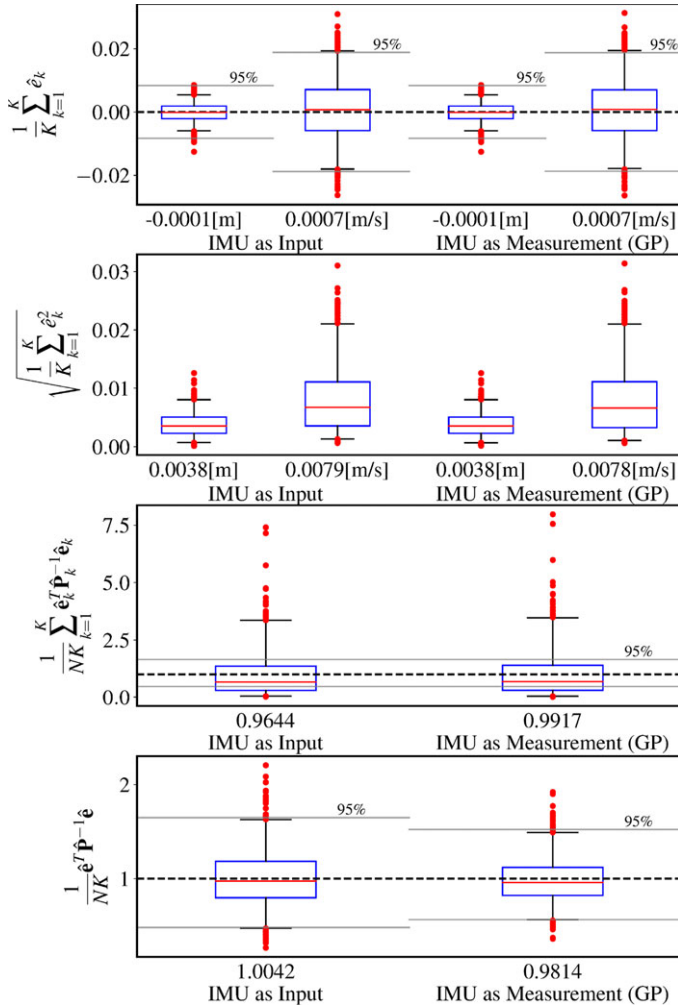


Figure 7. This figure summarizes the results of our second simulation experiment where the ground-truth trajectories were sampled from a singer prior with $\alpha = 10.0$, $\sigma^2 = 1.0$. The trained acceleration input covariance for IMU-as-input was $\mathbf{Q}_k \approx 0.00283$. The trained singer prior parameters were $\alpha = 10.2442$ and $\sigma^2 = 1.0074$. Note that even though the underlying prior changed by a lot from the first simulation to the second, from a white-noise-on-jerk prior to an approximation of a white-noise-on-acceleration prior, both estimators were able to remain unbiased and consistent.

prior factors between pairs of estimation times using

$$J_{v,k} = \frac{1}{2} \mathbf{e}_{v,k}^T \mathbf{Q}_k^{-1} \mathbf{e}_{v,k}, \quad (38a)$$

$$\mathbf{e}_{v,k} = \boldsymbol{\gamma}_k(t_{k+1}) - \boldsymbol{\Phi}(t_{k+1}, t_k) \boldsymbol{\gamma}_k(t_k). \quad (38b)$$

Our IMU measurement model is

$$\begin{bmatrix} \tilde{\mathbf{a}} \\ \tilde{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} \mathbf{a}_v^{vi} - \mathbf{C}_v \mathbf{g}_i \\ \boldsymbol{\omega}_v^{vi} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_a \\ \mathbf{b}_\omega \end{bmatrix} + \begin{bmatrix} \mathbf{w}_a \\ \mathbf{w}_\omega \end{bmatrix}, \quad (39)$$

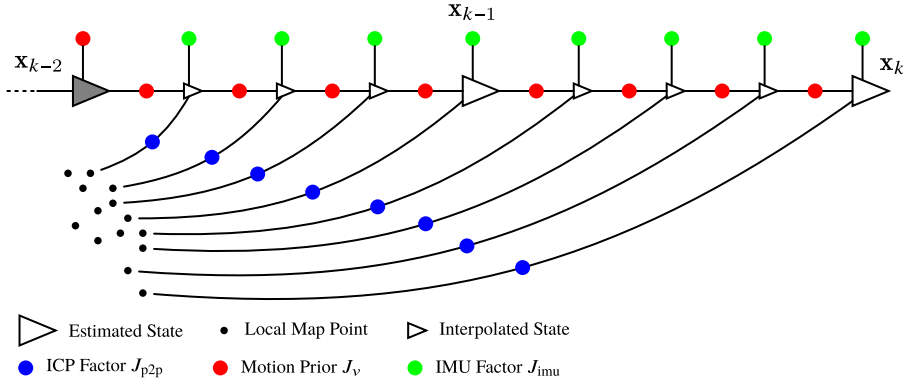


Figure 8. This figure depicts a factor graph of our sliding-window lidar-inertial odometry using a continuous-time motion prior. The larger triangles represent the estimation times that form our sliding window. The state $\mathbf{x}(t) = \{\mathbf{T}(t), \bar{\mathbf{w}}(t), \bar{\mathbf{w}}(t), \mathbf{b}(t)\}$ includes the pose $\mathbf{T}(t)$, the body-centric velocity $\bar{\mathbf{w}}(t)$, the body-centric acceleration $\bar{\mathbf{w}}(t)$, and the IMU biases $\mathbf{b}(t)$. The gray-shaded state \mathbf{x}_{k-2} is next to be marginalized and is held fixed during the optimization of the current window. The smaller triangles are interpolated states that we do not directly estimate during the optimization process. Instead, we construct continuous-time measurement factors using the posterior Gaussian process interpolation formula. We include a unary prior on \mathbf{x}_{k-2} to denote the prior information from the sliding-window filter.

where \mathbf{b}_a and \mathbf{b}_ω are the accelerometer and gyroscope biases, $\mathbf{w}_a \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_a)$ and $\mathbf{w}_\omega \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_\omega)$ are zero-mean Gaussian noise. Due to angular velocity and acceleration being a part of the state, the associated IMU error function is straightforward:

$$J_{\text{imu},\ell} = \frac{1}{2} \begin{bmatrix} \mathbf{e}_{a,\ell} \\ \mathbf{e}_{\omega,\ell} \end{bmatrix}^T \begin{bmatrix} \mathbf{R}_a & \\ & \mathbf{R}_\omega \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{e}_{a,\ell} \\ \mathbf{e}_{\omega,\ell} \end{bmatrix}, \quad (40a)$$

$$\mathbf{e}_{a,\ell} = \tilde{\mathbf{a}}_\ell - \dot{\mathbf{v}}(\tau_\ell) + \mathbf{C}_{vi}(\tau_\ell)\mathbf{g}_i - \mathbf{b}_a(\tau_\ell), \quad (40b)$$

$$\mathbf{e}_{\omega,\ell} = \tilde{\boldsymbol{\omega}}_\ell - \boldsymbol{\omega}(\tau_\ell) - \mathbf{b}_\omega(\tau_\ell), \quad (40c)$$

where we rely on forming measurement factors using the posterior Gaussian process interpolation formula. For each of these continuous-time measurement factors, we compute Jacobians of the perturbation to the state at the interpolated times with respect to the bracketing estimation times. This is an approximation, as it is not exactly the same as marginalization. However, we have found it to work well in practice. We also include motion prior factors for the IMU biases:

$$J_{v,b,k} = \frac{1}{2} \mathbf{e}_{v,b,k}^T \mathbf{Q}_{b,k}^{-1} \mathbf{e}_{v,b,k}, \quad (41a)$$

$$\mathbf{e}_{v,b,k} = \mathbf{b}(t_{k+1}) - \mathbf{b}(t_k), \quad (41b)$$

where $\mathbf{Q}_{b,k} = \mathbf{Q}_b \Delta t_k$ is the covariance resulting from a white-noise-on-velocity motion prior and \mathbf{Q}_b is the associated power spectral density matrix. We use point-to-plane factors. The associated error function is

$$J_{p2p,j} = \mathbf{e}_{p2p,j}^T \mathbf{R}_{p2p,j}^{-1} \mathbf{e}_{p2p,j}, \quad (42a)$$

$$\mathbf{e}_{p2p,j} = \alpha_j \mathbf{n}_j^T \mathbf{D}(\mathbf{p}_j - \mathbf{T}_{vi}(\tau_j)^{-1} \mathbf{T}_{vs} \mathbf{q}_j), \quad (42b)$$

where \mathbf{q}_j is the query point, \mathbf{p}_j is the matched point in the local map, \mathbf{n}_j is an estimate of the surface normal at \mathbf{p}_j given neighboring points in the map, \mathbf{D} is a constant matrix removing the homogeneous component, \mathbf{T}_{vs} is a known extrinsic calibration between the lidar frame and the vehicle frame, and $\alpha_j = (\sigma_2 - \sigma_3)/\sigma_1$ [31] is a heuristic weight to favor planar neighborhoods. The objective function that we seek to minimize is

$$J = \sum_k J_{v,k} + \sum_j J_{p2p,j} + \sum_\ell J_{imu,\ell}. \quad (43)$$

We solve this nonlinear least squares problem for the optimal perturbation to the state using Gauss-Newton. Once the solver has converged, we update the pointcloud correspondences and iterate this two-step process to convergence. In practice, we typically limit the maximum number of inner-loop Gauss-Newton iterations to 10, and the number of outer-loop iterations to 10 in order to enable real-time operation.

In our approach, we estimate the orientation of the gravity vector relative to the initial map frame at startup. We perform sliding-window batch trajectory estimation where the length of the sliding window is 200 ms. We output the pose at the middle of the newest lidar scan.

5. IMU-as-input lidar-inertial baseline

As a baseline where IMU measurements are treated as an input, we consider a lidar-inertial odometry approach where IMU measurements are used to de-skew the lidar pointcloud and classic preintegration is used as a prior. The baseline is implemented as sliding-window batch trajectory estimation and the factor graph corresponding to the baseline approach is depicted in Figure 9. The state $\mathbf{x}(t_k) = \{\mathbf{C}_{iv}(t_k), \mathbf{r}_i^{vi}(t_k), \mathbf{v}_i^{vi}(t_k), \mathbf{b}(t_k)\}$ consists of the orientation $\mathbf{C}_{iv}(t_k)$, position $\mathbf{r}_i^{vi}(t_k)$, velocity $\mathbf{v}_i^{vi}(t_k)$, and IMU biases. All variables are expressed in a global frame. We use classic preintegration to form binary factors between pairs of estimated states in the sliding window [2]. At each iteration of the optimization, we integrate the IMU measurements to extrapolate for the state at each IMU measurement time using

$$\mathbf{C}_j = \mathbf{C}_i \prod_{k=i}^{j-1} \exp(\Delta t_k (\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_\omega(t_k))^\wedge), \quad (44a)$$

$$\mathbf{v}_j = \mathbf{v}_i + \mathbf{g} \Delta t_{ij} + \sum_{k=i}^{j-1} \mathbf{C}_k (\tilde{\mathbf{a}}_k - \mathbf{b}_a(t_k)) \Delta t_k, \quad (44b)$$

$$\mathbf{r}_j = \mathbf{r}_i + \sum_{k=i}^{j-1} \left[\mathbf{v}_k \Delta t_k + \frac{1}{2} \mathbf{g} \Delta t_k^2 + \frac{1}{2} \mathbf{C}_k (\tilde{\mathbf{a}}_k - \mathbf{b}_a(t_k)) \Delta t_k^2 \right]. \quad (44c)$$

The position at a given lidar point time can then be obtained by linearly interpolating between the positions at the IMU measurement times. The orientation at a given lidar point time can be obtained using the following formula:

$$\mathbf{C}(\tau_j) = \mathbf{C}(t_\ell) \left(\mathbf{C}(t_\ell)^T \mathbf{C}(t_{\ell+1}) \right)^\alpha \quad (45)$$

where $\alpha = (\tau_j - t_\ell)/(t_{\ell+1} - t_\ell)$. Using these interpolated states, we can write the point-to-plane error function as:

$$\mathbf{e}_{p2p,j} = \mathbf{n}_j^T \left(\mathbf{p}_j - \mathbf{C}_{iv}(\tau_j) (\mathbf{C}_{vs} \mathbf{q}_j + \mathbf{r}_v^{sv}) - \mathbf{r}_i^{vi}(\tau_j) \right). \quad (46)$$

The Jacobians of this error function with respect to perturbations to the state variables are provided in the appendix.

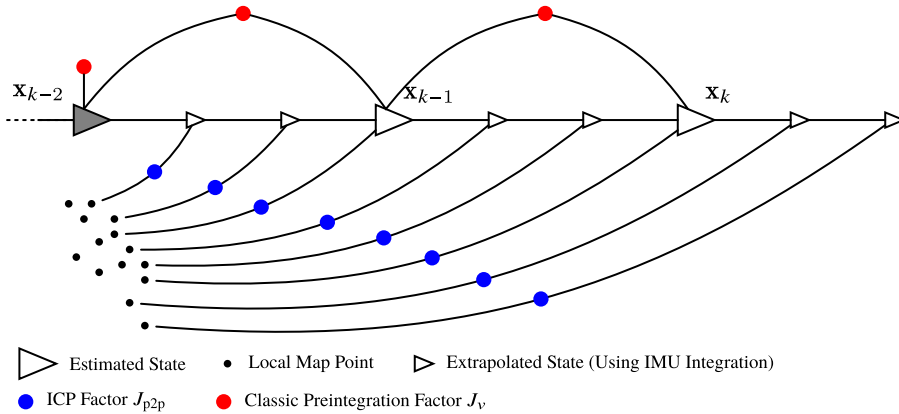


Figure 9. This figure depicts a factor graph of our baseline approach that uses IMU measurements to de-skew the pointcloud and to form relative motion priors using classic preintegration. The larger triangles represent the estimation times that form our sliding window. The state $\mathbf{x}(t_k) = \{\mathbf{C}_{iv}(t_k), \mathbf{r}_i^v(t_k), \mathbf{v}_i^v(t_k), \mathbf{b}(t_k)\}$ includes the orientation and position in a global frame, the velocity in a global frame, and the IMU biases. The gray-shaded state \mathbf{x}_{k-2} is next to be marginalized. The smaller triangles are extrapolated states that we do not directly estimate during the optimization process. Instead, we extrapolate for these states using IMU integration starting at an estimated state. The factor graph includes a unary prior on \mathbf{x}_{k-2} to denote the prior information from the sliding-window filter.

6. Lidar-inertial simulation

In this section, we compare the performance of our lidar-inertial odometry to the baseline imu-as-input approach in a simulated environment. The simulated environment is a rectangular room, and we simulate trajectories using sinusoidal body-centric velocities:

$$[\boldsymbol{\omega}(t)]_j = A_j \sin(2\pi f_j t), \quad (47)$$

where A_j and f_j are configurable amplitude and frequency parameters. The resulting body-centric acceleration can be obtained via differentiation:

$$[\dot{\boldsymbol{\omega}}(t)]_j = A_j 2\pi f_j \cos(2\pi f_j t). \quad (48)$$

We then step through the simulation so as to replicate the lidar firing sequence of a Velodyne Alpha-Prime 128-beam lidar, obtaining the pose of the sensor for each firing sequence. Starting with $\mathbf{T}_0 = \mathbf{T}_{vi}(t_0) = \mathbf{1}$,

$$\mathbf{T}_{k+1} \approx \exp \left(\left(\boldsymbol{\omega}(t_k) \Delta t_k + \frac{1}{2} \Delta t_k^2 \dot{\boldsymbol{\omega}}(t_k) \right)^\wedge \right) \mathbf{T}_k, \quad (49)$$

where Δt_k is very small ($53.3\mu\text{s}$). By generating the trajectories in this way, it is straightforward to extract the body-centric angular velocity and linear acceleration to simulate IMU measurements. We use the measurement model in (39) to simulate biases and gravity components. We simulate a constant nonzero bias on each gyroscope and accelerometer axis. We include zero-mean Gaussian noise on IMU measurements as well as Gaussian noise on the range measurement of each lidar point. We chose measurement noises to be close to what we experience on our experimental platform. Figure 10 depicts an example pointcloud produced in our simulation environment where the points are colored based off which wall they are reflected. Figure 11 compares the trajectory estimated by our lidar-inertial odometry with the ground truth.

For the simulation parameters, we use an IMU rate of 200 Hz, a simulation length of 20s, and three motion regimes denoted *slow*, *medium*, and *fast*. Where for each of these motion regimes, we randomly

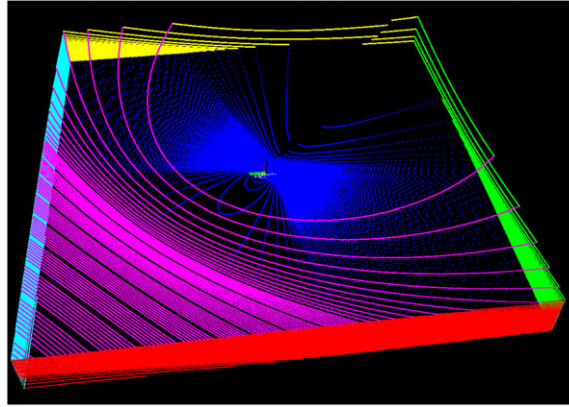


Figure 10. This figure depicts an example lidar pointcloud produced by our simulation, which contains motion distortion. The pointcloud is colored based off which wall the lidar point is reflected.

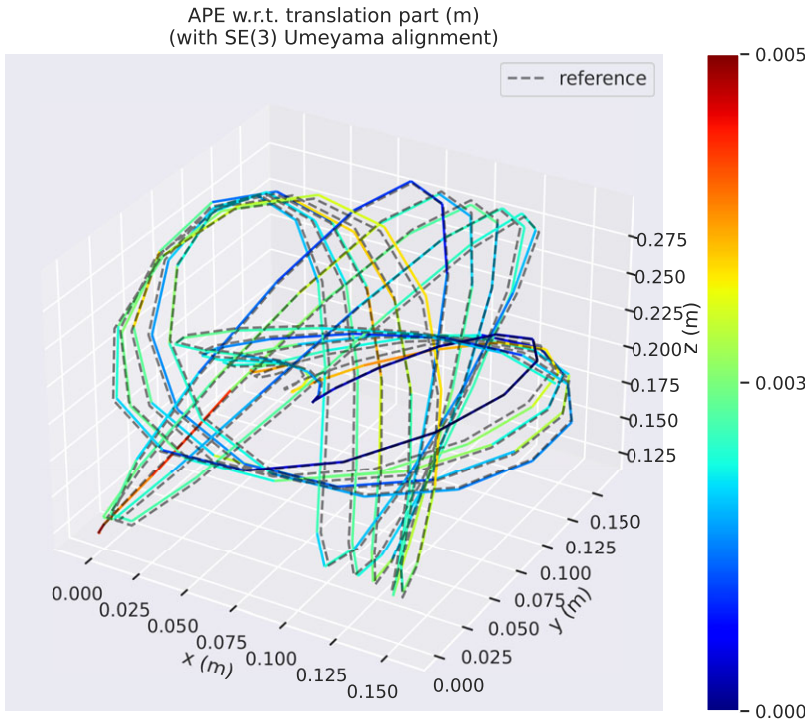


Figure 11. This figure depicts the results of our lidar-inertial simulation where the ground-truth position (dashed line) is compared to the position estimated by Singer-LIO colored by the absolute position error. This trajectory is an example of one of the slow sequences.

sample for the amplitudes and frequencies of the body-centric velocities used in the simulation of a sequence. The ranges for these parameters is given in Table I. One set of amplitudes and frequencies is sampled for each of the 20 sequences simulated for the three motion regimes. We set the standard deviation of the accelerometer measurement noise to 0.02 m/s^2 , the standard deviation of the gyroscope measurement noise to 0.01 rad/s , and the standard deviation of the lidar range measurements to 0.02 m .

Table I. Simulation parameter ranges for the different motion regimes.

| | Slow | Medium | Fast |
|------------------------------------|--------------------|--------------------|--------------------|
| Linear velocity amplitude [m/s] | $A \in [0.1, 0.5]$ | $A \in [0.5, 1.0]$ | $A \in [1.0, 2.0]$ |
| Angular velocity amplitude [rad/s] | $A \in [0.1, 0.5]$ | $A \in [0.5, 1.0]$ | $A \in [1.0, 2.0]$ |
| Linear velocity frequency [Hz] | $f \in [0.5, 1.0]$ | $f \in [1.0, 2.0]$ | $f \in [2.0, 4.0]$ |
| Angular velocity frequency [Hz] | $f \in [1.0, 2.0]$ | $f \in [2.0, 4.0]$ | $f \in [4.0, 8.0]$ |

Table II. Simulation results. Root mean squared absolute trajectory error (m). For each speed category (slow, medium, and fast), 20 randomized sequences were created. The results in this table are the overall root mean squared absolute position error across 20 sequences, for each approach.

| | Slow | Medium | Fast |
|-------------------------|---------------|---------------|---------------|
| Baseline (IMU as input) | 0.0026 | 2.1734 | Failed |
| Singer-LIO | 0.0026 | 0.0025 | 0.0208 |
| Singer-LO + Gyro | 0.0052 | 0.0085 | 0.0445 |
| Singer-LO | 0.0012 | 12.34 | Failed |

The accelerometers were given a constant bias of 0.05 m/s^2 , and the gyroscopes were given a constant bias of 0.05 rad/s .

We compare the performance of our lidar-inertial odometry against the baseline in Table II where we also show the performance of our approach using only the lidar and the gyroscope, and lidar only. We obtained the results by computing the absolute trajectory error between our estimated trajectories and the ground truth using the evo evaluation tool.¹ The results in the table are the overall root mean squared error obtained by concatenating the error from each individual sequence. The results show that in the low speed regime, the imu-as-input baseline approach and our imu-as-measurement approach based on the Singer prior achieve nearly identical results. This is unsurprising as it appears to replicate the results from Section 3.5. Interestingly, our lidar-only approach performs the best on the slow regime. However, in the medium and fast regime, the advantage of our lidar-inertial approach becomes apparent. In the medium regime, the baseline imu-as-input approach begins to break down. This is possibly due to the fact that the motion is no longer approximately constant acceleration and constant angular rate. On the other hand, our lidar-inertial approach performs roughly the same in the medium regime. Our lidar-only approach also breaks down in the medium regime. In the fast regime, both our lidar-inertial and lidar with gyro approaches achieve respectable results, while the lidar-only approach and the imu-as-input baseline fail completely.

7. Experimental results

In this section, we provide experimental results on the Newer College Dataset [32]. This dataset features a 64-beam Ouster lidar and provides the internal IMU measurements of the Ouster lidar. Ground-truth poses were obtained by matching live lidar poses to a map of the environment created using a survey-grade lidar at several stationary poses. This dataset is somewhat unique in that it features several sequences with highly dynamic motions. In Table III, we compare the performance of using our continuous-time Singer prior using lidar only (Singer-LO), lidar and a gyroscope only (Singer-LO + Gyro), and a full lidar-inertial setup including an accelerometer (Singer-LIO). We also compare the performance of our approach to some comparable works in the literature such as CT-ICP [31], a lidar-only approach, FAST-LIO2 [20] and DLIO [21], which can be considered the prior state of the

Table III. *Newer College Dataset results. Root mean squared absolute trajectory error (m). Estimated trajectory aligned with ground truth using Umeyama algorithm. * uses loop closures, † results obtained from [21], ‡ uses camera images.*

| Newer College Dataset | 01-Short | 02-Long | 05-Quad w/ Dynamics | 06-Dynamic Spinning | 07-Parkland Mound | Overall |
|-------------------------|---------------|---------------|---------------------|---------------------|-------------------|---------------|
| CT-ICP* [31] | 0.36 | | | | | |
| KISS-ICP† [34] | 0.6675 | 1.5311 | 0.1040 | Failed | 0.2027 | |
| FAST-LIO2† [20] | 0.3775 | 0.3324 | 0.0879 | 0.0771 | 0.1483 | 0.3152 |
| DLIO [21] | 0.3606 | 0.3268 | 0.0837 | 0.0612 | 0.1196 | 0.3048 |
| SLICT* [33] | 0.3843 | 0.3496 | 0.1155 | 0.0844 | 0.1290 | 0.3263 |
| CLIO*‡ [8] | 0.408 | 0.381 | | 0.091 | | |
| Singer-LO (Ours) | 0.4543 | Failed | 0.1120 | 0.0804 | Failed | |
| Singer-LO + Gyro (Ours) | 0.3044 | 0.3267 | 0.1092 | 0.0818 | 0.1457 | 0.2887 |
| Singer-LIO (Ours) | 0.3020 | 0.3186 | 0.1091 | 0.0821 | 0.1411 | 0.2832 |

art for this dataset, and SLICT [33] and CLIO [8], which are two continuous-time approaches that use linear interpolation and B-splines, respectively.

Our approach, Singer-LIO, demonstrates the best performance on the 01-Short and 02-Long sequences and also demonstrates the best overall performance. Interestingly, the sequences in which we expected the IMU to make the most difference were 05-Quad w/ Dynamics and 06-Dynamic Spinning due to their dynamic motions. However, we observe that in these sequences, our lidar-only approach performs similarly or even better, replicating the results of our lidar-inertial simulation. It appears that, in this dataset, the addition of an IMU mainly helps in areas where there are geometric degeneracies rather than the areas with dynamic motions. Sequences 05-Quad w/ Dynamics and 06-Dynamic Spinning are very similar to our lidar-inertial simulation in the slow regime, as they are conducted in a rectangular quad at New College, Oxford. FAST-LIO2 and DLIO can be considered state-of-the-art IMU-as-input approaches, and our approach demonstrates a clear advantage over these methods.

8. Conclusions

In this work, we compared treating an IMU as an input to a motion model versus treating it as a measurement of the state. On a 1D simulation problem, we showed that these two approaches performed identically when the data are sampled from either a constant velocity or constant acceleration prior and both methods are trained on a hold-out set. We demonstrated our approach to continuous-time lidar-inertial odometry using the Singer prior where body-centric acceleration is included in the state. In our simulated environment, we showed that our lidar-inertial odometry outperformed lidar-only odometry and an IMU-as-input baseline approach. On the Newer College Dataset, we demonstrated state-of-the-art results. There is still plenty of work to be done in treating IMU measurements as measurements of the state. Similar to nonuniform B-splines, it would be interesting to investigate a setup where the parameters of the Singer prior are adjusted on the fly so as to adjust between periods of smooth versus highly dynamic motion. When the IMU is treated as a measurement of the state, this allows us to now incorporate exogenous control inputs into our Gaussian process motion prior. This could be a promising area of research for estimating the state of drones where the torque commanded to the motors is often known. Our approach to combine multiple asynchronous high-rate sensors may prove beneficial in other sensor configurations such as multiple asynchronous IMUs.

Author contributions. Keenan Burnett carried out the research and wrote the article, Angela Schoellig provided feedback on research, and Tim Barfoot conceived of the study and provided feedback on research.

Financial support. This work was supported in part by the National Sciences and Engineering Research Council of Canada (NSERC) and by an Ontario Research Fund: Research Excellence (ORF-RE) grant.

Competing interests. The authors declare none.

Data availability. Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Supplementary material. The supplementary material for this article can be found at <https://doi.org/10.1017/S0263574724002121>.

References

- [1] T. Lupton and S. Sukkarieh, “Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions,” *IEEE T. Robot.* **28**(1), 61–76 (2012).
- [2] C. Forster, L. Carlone, F. Dellaert and D. Scaramuzza, “On-manifold preintegration for real-time visual-inertial odometry,” *IEEE T. Robot.* **33**(1), 1–21 (2017).
- [3] M. Brossard, A. Barrau, P. Chauchat and S. Bonnabel, “Associating uncertainty to extended poses for on lie group imu preintegration with rotating earth,” *IEEE T. Robot.* **38**(2), 998–1015 (2022).
- [4] D. Droschel and S. Behnke, “Efficient continuous-time slam for 3d lidar-based online mapping,” In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE (2018) pp. 5000–5007.
- [5] J. Quenzel and S. Behnke, “Real-time multi-adaptive-resolution-surfel 6d lidar odometry using continuous-time trajectory optimization,” In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2021) pp. 5499–5506.
- [6] C. Park, P. Moghadam, J. L. Williams, S. Kim, S. Sridharan and C. Fookes, “Elasticity meets continuous-time: Map-centric dense 3d lidar slam,” *IEEE T. Robot.* **38**(2), 978–997 (2021).
- [7] Y. Z. Ng, B. Choi, R. Tan and L. Heng, “Continuous-time radar-inertial odometry for automotive radars,” In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2021) pp. 323–330.
- [8] J. Lv, X. Lang, J. Xu, M. Wang, Y. Liu and X. Zuo, “Continuous-time fixed-lag smoothing for lidar-inertial-camera slam,” *IEEE/ASME Trans. Mechatron.* **28**(4), 2259–2270 (2023).
- [9] X. Lang, C. Chen, K. Tang, Y. Ma, J. Lv, Y. Liu and X. Zuo, “Coco-lic: Continuous-time tightly-coupled lidar-inertial-camera odometry using non-uniform b-spline,” *IEEE Robot. Autom. Lett.* **8**(11), 7074–7081 (2023).
- [10] J. Johnson, J. Mangelson, T. Barfoot and R. Beard, “Continuous-time trajectory estimation: A comparative study between Gaussian process and spline-based approaches,” (2024). arXiv preprint [arXiv:2402.00399](https://arxiv.org/abs/2402.00399).
- [11] X. Zheng and J. Zhu, “Traj-lio: A resilient multi-lidar multi-imu state estimator through sparse gaussian process,” (2024). arXiv preprint [arXiv:2402.09189](https://arxiv.org/abs/2402.09189).
- [12] M. Helmberger, K. Morin, B. Berner, N. Kumar, G. Cioffi and D. Scaramuzza, “The hilti slam challenge dataset,” *IEEE Robot. Autom. Lett.* **7**(3), 7518–7525 (2022).
- [13] K. Burnett, Y. Wu, D. J. Yoon, A. P. Schoellig and T. D. Barfoot, “Are we ready for radar to replace lidar in all-weather mapping and localization?,” *IEEE Robot. Autom. Lett.* **7**(4), 10328–10335 (2022).
- [14] T. Y. Tang, D. J. Yoon and T. D. Barfoot, “A white-noise-on-jerk motion prior for continuous-time trajectory estimation on se(3),” *IEEE Robot. Autom. Lett.* **4**(2), 594–601 (2019).
- [15] J. N. Wong, D. J. Yoon, A. P. Schoellig and T. D. Barfoot, “Variational inference with parameter learning applied to vehicle trajectory estimation,” *IEEE Robot. Autom. Lett.* **5**(4), 5291–5298 (2020).
- [16] C. L. Gentil, T. Vidal-Calleja and S. Huang, “In2laama: Inertial lidar localization autocalibration and mapping,” *IEEE T. Robot.* **37**(1), 275–290 (2020).
- [17] C. L. Gentil and T. Vidal-Calleja, “Continuous latent state preintegration for inertial-aided systems,” *Int. J. Robot. Res.* **42**(10), 874–900 (2023).
- [18] Y. Hu, Q. Zhou, Z. Miao, H. Yuan and S. Liu, “Outdoor lidar-inertial slam using ground constraints,” *Robotica* **42**(4), 1246–1261 (2024).
- [19] R. Li, X. Zhang, S. Zhang, J. Yuan, H. Liu and S. Wu, “Ba-liom: Tightly coupled laser-inertial odometry and mapping with bundle adjustment,” *Robotica* **42**(3), 684–700 (2024).
- [20] W. Xu, Y. Cai, D. He, J. Lin and F. Zhang, “Fast-lio2: Fast direct lidar-inertial odometry,” *IEEE T. Robot.* **38**(4), 2053–2073 (2022).
- [21] K. Chen, R. Nemiroff and B. T. Lopez, “Direct lidar-inertial odometry: Lightweight lio with continuous-time motion correction,” In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE (2023) pp. 3983–3989.
- [22] D. T. Fasiolo, L. Scalera and E. Maset, “Comparing lidar and imu-based slam approaches for 3d robotic mapping,” *Robotica* **41**(9), 2588–2604 (2023).
- [23] K. Burnett, A. P. Schoellig and T. D. Barfoot, “Continuous-time radar-inertial and lidar-inertial odometry using a gaussian process motion prior,” (2024). arXiv preprint [arXiv:2402.06174](https://arxiv.org/abs/2402.06174).
- [24] W. Talbot, J. Nubert, T. Tuna, C. Cadena, F. Dümbsen, J. Tordesillas, T. D. Barfoot and M. Hutter, “Continuous-time state estimation methods in robotics: A survey,” (2024). arXiv preprint [arXiv:2411.03951](https://arxiv.org/abs/2411.03951).
- [25] S. Li, J. Nie, C. Guo, Y. Yang and L. Mei, “A novel online time calibration framework via double-stage ekf for visual-inertial odometry,” *Robotica*, 1–17 (2024).
- [26] J. M. Ferguson, T. E. Ertop, S. D. Herrell and R. J. Webster, “Unified robot and inertial sensor self-calibration,” *Robotica* **41**(5), 1590–1616 (2023).

- [27] S. Anderson, T. D. Barfoot, C. H. Tong and S. Särkkä, “Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression,” *Auton. Robot.* **39**(3), 221–238 (2015).
- [28] S. Anderson and T. D. Barfoot, “Full steam ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on se(3),” *In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2015) pp. 157–164.
- [29] T. D. Barfoot, *State Estimation for Robotics*. 2nd edition, Cambridge University Press, (2024).
- [30] J. N. Wong, D. J. Yoon, A. P. Schoellig and T. D. Barfoot, “A data-driven motion prior for continuous-time trajectory estimation on se(3),” *IEEE Robot. Autom. Lett.* **5**(2), 1429–1436 (2020).
- [31] P. Dellenbach, J.-E. Deschaud, B. Jacquet and F. Goulette, “Ct-icp: Real-time elastic lidar odometry with loop closure,” *In 2022 International Conference on Robotics and Automation (ICRA)*, IEEE (2022) pp. 5580–5586.
- [32] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala and M. Fallon, “The newer college dataset: Handheld lidar, inertial and vision with ground truth,” *In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2020) pp. 4353–4360.
- [33] T.-M. Nguyen, D. Duberg, P. Jensfelt, S. Yuan and L. Xie, “SlicT: Multi-input multi-scale surfel-based lidar-inertial continuous-time odometry and mapping,” *IEEE Robot. Autom. Lett.* **8**(4), 2102–2109 (2023).
- [34] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley and C. Stachniss, “Kiss-icp: In defense of point-to-point icp—simple, accurate, and robust registration if done the right way,” *IEEE Robot. Autom. Lett.* **8**(2), 1029–1036 (2023).