2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

IN2LAMA: INertial Lidar Localisation And MApping

Cedric Le Gentil, Teresa Vidal-Calleja and Shoudong Huang

***Abstract*— In this paper, we introduce a probabilistic framework for INertial Lidar Localisation And MApping (IN2LAMA). Most of today's lidars are based on spinning mechanisms that do not capture snapshots of the environment. As a result, movement of the sensor can occur while scanning. Without a good estimation of this motion, the resulting point clouds might be distorted. In the lidar mapping literature, a constant velocity motion model is commonly assumed. This is an approximation that does not necessarily always hold. The key idea of the proposed framework is to exploit preintegrated measurements over upsampled inertial data to handle motion distortion without the need for any explicit motion-model. It tightly integrates inertial and lidar data in a batch on-manifold optimisation formulation. Using temporally precise upsampled preintegrated measurement allows frame-to-frame planar and edge features association. Moreover, features are re-computed when the estimate of the state changes, consolidating front-end and back-end interaction. We validate the effectiveness of the approach through simulated and real data.**

I. Introduction

The past few years saw the emergence of a new kind of business: mapping as a service. Companies like *Kaarta*[1] or *GeoSLAM*[2] use multi-sensor localisation and mapping algorithms to map various environments and provide their customers with detailed 3D models of the areas of interest. One can imagine the use of such services for applications in architecture, archaeology, structure surveillance, etc. This work presents INertial Lidar Localisation And MApping (IN2LAMA), a probabilistic framework for localisation and mapping based on a 3D-lidar range scanner and a 6-DoF-Inertial Measurement Unit (IMU), which aims at contributing to the automation of such services.

Unlike outdoor scenarios, where a GPS can provide substantial spatial information, an accurate position is not readily available for indoor localisation systems, in particular without the help of any additional infrastructure. In various fields, lidars proved to be the most appropriate tool for estimating the real world geometry. Despite delivering reliable range measurements, most of today's lidars have spinning mechanisms. If the motion is not properly handled, any movement of the sensor during a scan collection will introduce motion-distortion. For example, when considering scans as snapshots, any movement will distort the resulting point clouds. In other words, it corresponds to the assumption of no motion during sweeps. This problem has been addressed in the literature. For instance in [1], the authors proposed a 2D registration of 2D laser scans assuming constant velocity during the sweep. Using such a motion model, the scope of the standard Iterative Closest Point (ICP) [2] is extended to estimate both the pose and the velocity of the system. This makes the system able to correct the motion distortion partially.

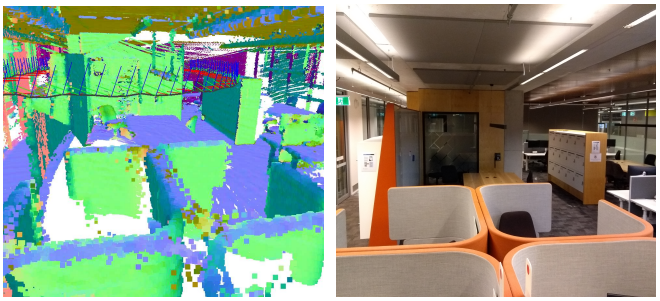

Fig. 1: Estimated map and trajectory in a real office environment versus the actual picture of the place. The map is colourised according to the post-processed normal vectors of the point cloud.

A constant velocity model is a common approach in the lidar mapping literature as it can be seen in [3] and [4]. On the other hand, to work around the motion distortion issue, our method does not consider the lidar measurements as snapshots of the environment, but as extremely high frequency individual 3D-point measurements.

Moreover, the method in [4] proposes a continuous state representation based on control-points and interpolates linearly in between. More than handling motion distortion in this kind of *rolling-shutter* sensors, continuous state estimation can be used to fuse data from multiple non-synchronised sensors. In [5], the trajectory is modelled as a linear combination of temporal basis functions and allows fusion of visual and inertial measurements. While these methods provide greater representability compared to traditional discrete models, their performances rely on the veracity of the models assumed.

Another approach that relies on continuous state estimation is presented in [6]. In this method, the state can be queried at any point in time. The key idea behind this approach is the use of a computationally efficient Gaussian Process (GP) regression over a discrete maximum a posteriori estimation. One could think that a solution for the IMU-lidar pair could rely on estimating a continuous trajectory from the slowest sensor (IMU) and querying the pose at the frequency of the fastest one (lidar). Unfortunately for us, an accurate position cannot be recovered by using only IMU readings.

Inertial sensors have been extensively used in combination with visual sensors for localisation. Originally proposed

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

[1]http://www.kaarta.com/

[2]https://geoslam.com/

for visual-inertial fusion in [7] and [8], the preintegrated measurements allow the pre-processing of IMU readings to be independent from the initial pose and velocity. The aim is to prevent repetitive integration of inertial readings every time the linearisation point changes. In a calibration context, our earlier work [9] extended this concept to handle non-synchronised sensor readings through the continuous representation of inertial measurements. Using the preintegration over upsampled IMU readings provides inertial information for each of the lidar 3D-points during a sweep. We named these new measurements Upsampled Preintegrated Measurements (UPMs). The present paper reuses this paradigm in a localisation and mapping context to handle motion distortion by tightly coupling inertial and laser data, thus, without the need for an explicit motion-model.

Similar to visual-inertial systems, lidar-inertial systems require a front-end that handles the exteroceptive sensor data for mapping and data association. For instance, the maps generated in [10] and [11] make use of surfels to represent the environment. Surfels provide rich information of surfaces given dense enough point clouds. The method proposed in this work provides frame-to-frame feature extraction and matching techniques for sparse data collected with lidars such as the Velodyne VLP-16[3]. Given the low vertical resolution of such devices, our front-end was designed using a channel-by-channel feature extraction in a similar fashion to the one developed in [3].

Other front-end methods, such as the ones in [3] and [12], aim at undistorting the incoming point clouds before registering them into the map. Prior knowledge of the actual motion is used to perform the undistortion. Although necessary for real-time operations, such assumptions carry the risk of propagating the errors of inaccurate initial conditions. The proposed method does not address the problem of real-time operation. Instead, our framework considers the full trajectory in a batch-optimisation, reducing the sensitivity to initial condition errors.

The main contribution of this work is a probabilistic formulation for lidar-inertial localisation and mapping. It tightly integrates IMU and lidar data in a batch on-manifold optimisation formulation. It is based on the IMU's UPMs [9] to characterise motion in lidar sweeps without the explicit need for a motion model. Using temporally precise UPMs allows frame-to-frame feature matching in the presence of motion distortion through the manipulation of planar and edge features. Moreover, there is a strong back and forth interaction between the front-end and back-end; features are re-computed when required based on the current solution during the optimisation.

The structure of this paper is as follows. Section II gives an overview of the proposed localisation and mapping method. Section III provides the technical details of the back-end. Section IV explains the front-end part, considering both the feature extraction and the data association. The performance of the proposed method is presented in Section V through simulated and real data experiments. Finally, Section VI presents the conclusion and future work.

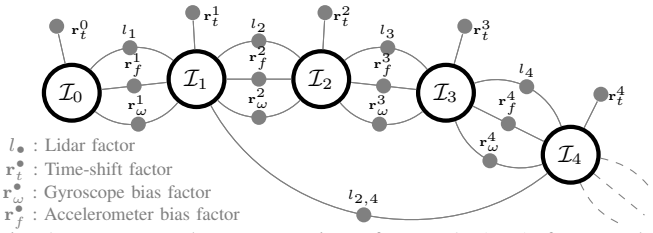

Fig. 2: Factor graph representation of an IN2LAMA framework example. $\mathcal{I}_m = \{\mathbf{R}_W^m,\ \mathbf{v}_W^m,\ \mathbf{p}_W^m,\ \hat{\mathbf{b}}_f^m,\ \hat{\mathbf{b}}_\omega^m,\ \hat{\delta_t}^m\}$ represents the IMU pose, velocity, biases and time-shift correction associated to the lidar frame $\mathcal{X}^m$ at τ_m. The factor $l_{2,4}$ represents a loop closure.

II. IN2LAMA OVERVIEW

Let us consider a system with a rigidly mounted 3D lidar and a 6-DoF IMU, where $\mathbf{R}_c$ and $\mathbf{p}_c$ respectively represent the relative rotation and translation from the IMU frame to the lidar frame. The system moves in the environment and the lidar provides 3D-points $\mathbf{x}_i$ at time t_i, grouped into M frames. We denote $\mathcal{X}^m$ the set of points contained in the m^{th} frame[4]. $\mathcal{F}^m$ is a subset of $\mathcal{X}^m$ that represents lidar feature-points. A feature is a point belonging to a distinctive type of surface (e.g. plane or edge). The set of feature associations $\mathcal{A}$ contains tuples of 3 or 4 lidar feature-points depending on whether they are edges or planes.

The inertial data include a 3-axis accelerometer and a 3-axis gyroscope, that provide respectively the raw readings $\mathbf{f}_q$ and ω_q at time t_q $(q = 1, \ldots, Q)$. To associate individual lidar points with IMU readings, GP regression is used to infer inertial readings on each IMU DoF independently at any given time t: continuous $\hat{\mathbf{f}}(t)$ and $\hat{\omega}(t)$ readings are estimated using GPs.

The proposed method aims to estimate the IMU orientation $\mathbf{R}_W^m$, position $\mathbf{p}_W^m$ and velocity $\mathbf{v}_W^m$ for each lidar frame, as well as the IMU biases ($\mathbf{b}_f^m$, $\mathbf{b}_\omega^m$) and time-shifts δ_t^m between the two sensors. The subscript $_W$ represents the earth-fixed world reference frame $\mathfrak{F}_W$. The superscript ^{m} denotes the m^{th} frame from the lidar and τ_m corresponds to the timestamp at the beginning of the m^{th} lidar frame.

In the following, $\mathcal{S}$ indicates the state to be estimated: $\mathcal{S} = (\mathbf{R}_W^0, \cdots, \mathbf{R}_W^M,\ \mathbf{p}_W^1, \cdots, \mathbf{p}_W^M,\ \mathbf{v}_W^0, \cdots, \mathbf{v}_W^M,\ \hat{\mathbf{b}}_f^0, \cdots, \hat{\mathbf{b}}_f^M,\ \hat{\mathbf{b}}_\omega^0, \cdots, \hat{\mathbf{b}}_\omega^M,\ \hat{\delta}_t^0, \cdots, \hat{\delta}_t^M)$ with $\hat{\mathbf{b}}_f^m$, $\hat{\mathbf{b}}_\omega^m$, and $\hat{\delta}_t^m$ the biases and time-shift corrections associated to the m^{th} lidar frame (more details are given in Section III). Note that $\mathbf{p}_W^0$ is not part of the state as one IMU position needs to be set arbitrarily to define the world frame.

The localisation and mapping problem is formulated as a Maximum Likelihood Estimation (MLE):

$$\mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ -\log(p(\mathcal{S}|\mathcal{Z})) = \underset{\mathcal{S}}{\operatorname{argmin}}\ C(\mathcal{S}), \qquad (1)$$

with $\mathcal{Z}$ representing the available measurements and C the optimisation cost function. Represented as the factor graph in Fig. 2, and under the assumption of zero-mean Gaussian

[3]https://velodynelidar.com/vlp-16.html

[4]A frame does not necessarily correspond to a 360-degree scan.

noise, it can be solved by minimising geometric distances $d_{\mathfrak{a}}$ associated with lidar features, biases factors, and time-shift factors. That is

$$C(\mathcal{S}) = \sum_{\mathfrak{a}\in\mathcal{A}} \|d_{\mathfrak{a}}\|^2_{\Sigma_{d_{\mathfrak{a}}}} + \sum_{m=0}^{M} \|r_t^m\|^2_{\Sigma_{r_t^m}} + \sum_{m=1}^{M} \left(\|\mathbf{r}_f^m\|^2_{\Sigma_{\mathbf{r}_f^m}} + \|\mathbf{r}_\omega^m\|^2_{\Sigma_{\mathbf{r}_\omega^m}}\right), \tag{2}$$

with $\mathbf{r}_f^m$, $\mathbf{r}_\omega^m$, and r_t^m corresponding respectively to the m^{th} accelerometer biases factor, gyroscope biases factor, and time-shift factor. Note that $\Sigma_\bullet$ is the covariance matrix of the variable $\bullet$.

A. Upsampled Preintegrated Measurements

The proposed method relies on the use of UPMs, which have been introduced in [9] based on principles originally presented in [7] and [8]. UPMs are used to constrain the motion in lidar scans. The original preintegrated measurements are

$$\begin{aligned}
\Delta\mathbf{p}_m^i &= \sum_{k=\kappa}^{i-1} \left(\Delta\mathbf{v}_m^k \Delta t_k + \frac{\Delta\mathbf{R}_m^k}{2}(\mathbf{f}(t_k - \delta_t^m) - \mathbf{b}_f^m)\Delta {t_k}^2\right) \\
\Delta\mathbf{v}_m^i &= \sum_{k=\kappa}^{i-1} \Delta\mathbf{R}_m^k(\mathbf{f}(t_k - \delta_t^m) - \mathbf{b}_f^m)\Delta t_k \\
\Delta\mathbf{R}_m^i &= \prod_{k=\kappa}^{i-1} \mathrm{Exp}\big((\boldsymbol{\omega}(t_k - \delta_t^m) - \mathbf{b}_\omega^m)\Delta t_k\big),
\end{aligned} \tag{3}$$

with $\{\kappa \in \mathbb{N} | t_\kappa = \tau_m\}$. Then

$$\mathbf{p}_W^i = \mathbf{p}_W^m + \Delta\varsigma_m^i \mathbf{v}_W^m + \frac{1}{2}\Delta {\varsigma_m^i}^2 \mathbf{g} + \mathbf{R}_W^m \Delta\mathbf{p}_m^i \tag{4}$$

$$\mathbf{v}_W^i = \mathbf{v}_W^m + \Delta\varsigma_m^i \mathbf{g} + \mathbf{R}_W^m \Delta\mathbf{v}_m^i \tag{5}$$

$$\mathbf{R}_W^i = \mathbf{R}_W^m \Delta\mathbf{R}_m^i, \tag{6}$$

where:

- $\mathbf{g}$ is the known gravity vector in $\mathfrak{F}_W$.
- $\Delta t_k = t_{k+1} - t_k$ and $\Delta\varsigma_m^i = t_i - \tau_m$.
- $\mathrm{Exp}(.)$ is the exponential mapping from axis-angle representations ($\mathfrak{so}(3)$) to rotation matrices ($SO(3)$)[5].

These measurements become UPMs when they are computed over interpolated IMU readings. The interpolation allows the computation of preintegrated measurements for any given time and therefore tackling the non-synchronisation of the sensor readings. The interpolation method employed in this paper is the GP regression [13] with constant mean functions and isometric Matern covariance functions for each independent IMU DoF.

III. IN2LAMA BACK-END

The cost function associated with the MLE consists of three terms or factor types; lidar, IMU biases, and inter-sensor time-shift.

A. Lidar factor

Lidar factors correspond to distance residuals computed between lidar feature-points and their corresponding feature-points from other lidar frames. As we will explain in the front-end section, the set of feature associations $\mathcal{A}$ contains tuples of 3 (point-to-edge constraints) or 4 feature-points (point-to-plane constraints).

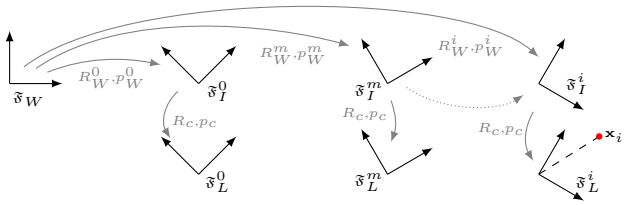

Fig. 3: Frames and transformations during a sequence of measurements. $\mathfrak{F}_I^\bullet$ and $\mathfrak{F}_L^\bullet$ respectively represent the IMU and lidar frames at time $t_\bullet$. The continuous line arrows represent the transformations between the different frames. $\mathfrak{F}_W$ is the world fixed frame. The dotted line shows the use of upsampled preintegrated measurements to reproject the point $\mathbf{x}_i$.

For the lidar factors, point-to-line or point-to-plane distances are used. The matched points found in $\mathcal{A}$ are projected in the world frame $\mathfrak{F}_W$ using the calibration parameters, UPMs for each of the points and the current estimates of the IMU poses and velocities (Fig. 3). Therefore a point $\mathbf{x}_i \in \mathcal{X}^m$ is projected into $\mathfrak{F}_W$ using (4) and (6)

$$\mathbf{x}_W^i = \mathbf{R}_W^i(\mathbf{R}_c\mathbf{x}_i + \mathbf{p}_c) + \mathbf{p}_W^i \tag{7}$$

Let us denote an edge association $\mathfrak{a}_3 \in \mathcal{A}$. $\mathfrak{a}_3 = \{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\}$ with $\mathbf{x}_i \in \mathcal{F}^m$, $\mathbf{x}_j \in \mathcal{F}^n$, $\mathbf{x}_k \in \mathcal{F}^o$ and $n, o \neq m$. These points are projected in $\mathfrak{F}_W$ via (7) to get $\mathbf{x}_W^i$, $\mathbf{x}_W^j$ and $\mathbf{x}_W^k$. The point-to-line distance

$$d_{\mathfrak{a}_3} = \frac{\|(\mathbf{x}_W^i - \mathbf{x}_W^j) \times (\mathbf{x}_W^i - \mathbf{x}_W^k)\|_2}{\|(\mathbf{x}_W^j - \mathbf{x}_W^k)\|_2} \tag{8}$$

is used as an edge feature residual.

Let us denote a plane association $\mathfrak{a}_4 \in \mathcal{A}$. $\mathfrak{a}_4 = \{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \mathbf{x}_l\}$ with $\mathbf{x}_i \in \mathcal{F}^m$, $\mathbf{x}_j \in \mathcal{F}^n$, $\mathbf{x}_k \in \mathcal{F}^o$, $\mathbf{x}_l \in \mathcal{F}^p$ and $n, o, p \neq m$. These points are projected in $\mathfrak{F}_W$ via (7) to get $\mathbf{x}_W^i$, $\mathbf{x}_W^j$, $\mathbf{x}_W^k$ and $\mathbf{x}_W^l$. The point-to-plane distance

$$d_{\mathfrak{a}_4} = \frac{(\mathbf{x}_W^i - \mathbf{x}_W^j)^\top \left((\mathbf{x}_W^j - \mathbf{x}_W^k) \times (\mathbf{x}_W^j - \mathbf{x}_W^l)\right)}{\|(\mathbf{x}_W^j - \mathbf{x}_W^k) \times (\mathbf{x}_W^j - \mathbf{x}_W^l)\|_2} \tag{9}$$

is used as a plane feature residual. As in [9], the variance of lidar residuals depends on the state. Therefore, the noise covariance propagation through the Jacobians needs to be executed regularly during the optimisation.

B. IMU biases and inter-sensor time-shift

The accelerometer and gyroscope biases, $\mathbf{b}_f$ and $\mathbf{b}_\omega$ respectively, are modelled by Brownian motion as in [14]. The computation of the UPMs also takes into account a time-shift to compensate potential inter-sensor timestamping inaccuracies. Unfortunately, the biases and time-shift are not perfectly known at the time of preintegration. Hence we adopted the first-order expansion presented in [8] to include

[5]The expressions for Exp mapping transformation can be found in [8].

a bias and time-shift correction into our method:

$$\begin{aligned}
\Delta\mathbf{R}^i_{m(\mathbf{b}_\omega,\delta_t)} \approx& \Delta\mathbf{R}^i_{m(\bar{\mathbf{b}}^m_\omega,\bar{\delta}^m_t)}\mathrm{Exp}\big(\tfrac{\partial\Delta\mathbf{R}^i_m}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}^m_\omega + \tfrac{\partial\Delta\mathbf{R}^i_m}{\partial\delta_t}\hat{\delta}^m_t\big)\\
\Delta\mathbf{v}^i_{m(\mathbf{b}_f,\mathbf{b}_\omega,\delta_t)} \approx& \Delta\mathbf{v}^i_{m(\bar{\mathbf{b}}^m_f,\bar{\mathbf{b}}^m_\omega,\bar{\delta}^m_t)} + \tfrac{\partial\Delta\mathbf{v}^i_m}{\partial\mathbf{b}_f}\hat{\mathbf{b}}^m_f\\
&+ \tfrac{\partial\Delta\mathbf{v}^i_m}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}^m_\omega + \tfrac{\partial\Delta\mathbf{v}^i_m}{\partial\delta_t}\hat{\delta}^m_t\\
\Delta\mathbf{p}^i_{m(\mathbf{b}_f,\mathbf{b}_\omega,\delta_t)} \approx& \Delta\mathbf{p}^i_{m(\bar{\mathbf{b}}^m_f,\bar{\mathbf{b}}^m_\omega,\bar{\delta}^m_t)} + \tfrac{\partial\Delta\mathbf{p}^i_m}{\partial\mathbf{b}_f}\hat{\mathbf{b}}^m_f\\
&+ \tfrac{\partial\Delta\mathbf{p}^i_m}{\partial\mathbf{b}_\omega}\hat{\mathbf{b}}^m_\omega + \tfrac{\partial\Delta\mathbf{p}^i_m}{\partial\delta_t}\hat{\delta}^m_t,
\end{aligned} \tag{10}$$

with $\mathbf{b}^m_f = \bar{\mathbf{b}}^m_f + \hat{\mathbf{b}}^m_f$, $\mathbf{b}^m_\omega = \bar{\mathbf{b}}^m_\omega + \hat{\mathbf{b}}^m_\omega$, and $\delta^m_t = \bar{\delta}^m_t + \hat{\delta}^m_t$. Note that $\bar{\bullet}$ denotes the prior knowledge of the value and $\hat{\bullet}$ represents the correction. The residuals

$$\mathbf{r}^m_f = \bar{\mathbf{b}}^m_f + \hat{\mathbf{b}}^m_f - \bar{\mathbf{b}}^{m-1}_f - \hat{\mathbf{b}}^{m-1}_f \tag{11}$$

$$\mathbf{r}^m_\omega = \bar{\mathbf{b}}^m_\omega + \hat{\mathbf{b}}^m_\omega - \bar{\mathbf{b}}^{m-1}_\omega - \hat{\mathbf{b}}^{m-1}_\omega \tag{12}$$

are used in the biases factors to impose the Brownian motion constraint. We consider Gaussian noise around the prior time-shift knowledge. Therefore the residual $r^m_t = \hat{\delta}^m_t$ constitutes the time-shift factors.

IV. IN2LAMA FRONT-END

The front-end of the proposed method aims at populating a set $\mathcal{A}$ of lidar feature-point associations to allow lidar frame matching.

A. Feature extraction

The features used here are of a similar nature to the ones developed in [3]: planar and edge points. The features proposed by the authors in [3] have been designed for fast computation, but they suffer from a lack of score consistency. For example, in [3], a point belonging to a planar surface perpendicular to the laser viewpoint will have a lower smoothness score than the same planar surface observed from a different viewpoint angle. The planar surfaces, however, should have the same score to encourage good feature association from frame-to-frame. As the proposed method does not aim for real-time computation, we propose a feature extraction technique based on linear regression. We also introduce inward and outward edges as opposed to the unique edge feature in [3].

The vertical resolution of most of today's lidars have steered the design of our feature extraction algorithm towards a channel-by-channel method in a similar way than [3]. Given an N-channel lidar, each lidar frame $\mathcal{X}^m$ is split into N "lines", $\mathcal{N}^m_l$ ($l = 1, \cdots, N$), according to the 3D-points' elevation. All the points are given a curvature score. The curvature computation aims at fitting lines to two subsets of points adjacent to the point under examination $\mathbf{x}_i \in \mathcal{N}^m_l$, and then to retrieve the cosine of the angle between these two lines. The subsets, $\mathcal{L}_i$ and $\mathcal{R}_i$ contain the D previous and following measurements in $\mathcal{N}^m_l$.

First, the points need to be reprojected into the lidar frame at τ_m ($\mathfrak{F}^m_L$) to remove motion distortion according to the best current estimate of the state $\mathcal{S}$

$$\mathbf{x}^i_{L_m} = \mathbf{R}_c^\top\Big(\mathbf{R}^{m\top}_W\big(\mathbf{R}^i_W(\mathbf{R}_c\mathbf{x}_i + \mathbf{p}_c) + \mathbf{p}^i_W - \mathbf{p}^m_W\big) - \mathbf{p}_c\Big). \tag{13}$$

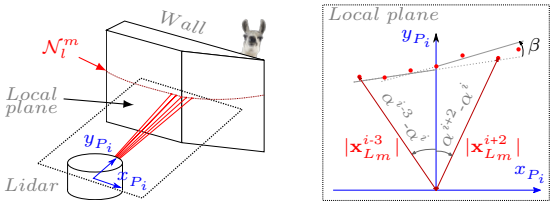

Fig. 4: Geometric feature extraction based on linear regression. The points around a given azimuth are assumed to belong to a local plane. On that local plane, linear regression is performed considering points in $\mathcal{N}^m_l$ on both sides of the point under examination $\mathbf{x}_i \in \mathcal{N}^m_l$. The curvature score is equal to $\cos(\beta)$ with β the angle between the two fitted lines.

The curvature scores are computed under the approximation that around a certain azimuth the consecutively measured 3D-points belong to the same plane. As shown in Fig. 4, and given α^i the new azimuth of $\mathbf{x}^i_{L_m}$; the points in $\mathcal{L}_i$ and $\mathcal{R}_i$ are projected on a planar space around α_i

$$x^k_{P_i} = |\mathbf{x}^{i+k}_{L_m}|\sin(\alpha^{i+k} - \alpha^i), \tag{14}$$

$$y^k_{P_i} = |\mathbf{x}^{i+k}_{L_m}|\cos(\alpha^{i+k} - \alpha^i), \tag{15}$$

with $k = -D, \cdots, D$.

$$\mathbf{X}_{\mathcal{L}_i} = \begin{bmatrix} 1 & x^{-D}_{P_i} \\ \vdots & \vdots \\ 1 & x^0_{P_i} \end{bmatrix}, \mathbf{X}_{\mathcal{R}_i} = \begin{bmatrix} 1 & x^0_{P_i} \\ \vdots & \vdots \\ 1 & x^D_{P_i} \end{bmatrix}, \tag{16}$$

$$\mathbf{Y}_{\mathcal{L}_i} = \begin{bmatrix} y^{-D}_{P_i} & \cdots & y^0_{P_i} \end{bmatrix}^\top \text{ and } \mathbf{Y}_{\mathcal{R}_i} = \begin{bmatrix} y^0_{P_i} & \cdots & y^D_{P_i} \end{bmatrix}^\top,$$

group the projected points coordinates according to the two adjacent subsets $\mathcal{L}_i$ and $\mathcal{R}_i$. In the rest of this section, $\bullet$ represents either $\mathcal{L}_i$ or $\mathcal{R}_i$. A line of slope $s_\bullet$ and y-intercept $q_\bullet$ can be fitted to the subset $\bullet$ with

$$\begin{bmatrix} q_\bullet & s_\bullet \end{bmatrix}^\top = \big(\mathbf{X}^\top_\bullet\mathbf{X}_\bullet\big)^{-1}\mathbf{X}^\top_\bullet\mathbf{Y}_\bullet, \tag{17}$$

and an associated unit direction vector can be obtained as

$$\mathbf{v}_\bullet = \begin{bmatrix} \frac{1}{\sqrt{1+s^2_\bullet}} & \frac{s_\bullet}{\sqrt{1+s^2_\bullet}} \end{bmatrix}^\top. \tag{18}$$

The score $c_i = \mathbf{v}^\top_{\mathcal{L}_i}\mathbf{v}_{\mathcal{R}_i}$ represents the cosine of the angle between the two fitted lines. As a consequence, c_i is close to 1 when the underlying surface is planar and decreases with the sharpness of edges. The error values

$$e^i_\bullet = \frac{1}{D+1}\sum\nolimits_{k|\mathbf{k}_i\in\bullet}\Big|y^k_{P_i} - q_\bullet - s_\bullet x^k_{P_i}\Big| \tag{19}$$

are used to reject points or to detect border of occlusions.

As in [3], surfaces close to being parallel to the laser beams are rejected as features. We also use a system of bins and a maximum number of features per bin on each laser line to ensure the features are spread over the whole scan. The points with the highest scores in $\mathcal{N}^m_l$ are classified as planar points and the lowest scores as edges. The edge orientation, inward (pointing toward the lidar) or outward (pointing away from the lidar), can be defined by looking at the value of the regressed line parameters. All the planar features in $\mathcal{N}^m_l$ with $l = 1, \cdots, N$, are grouped into a set $\mathcal{P}^m$, the inward edges in $\mathcal{E}^m_I$ and outward edges in $\mathcal{E}^m_O$.

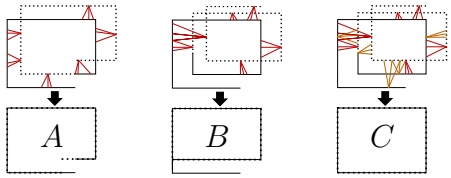

Fig. 5: Different data association strategies between a frame $\mathcal{X}^m$ (dashed line) and its previous frame $\mathcal{X}^{m-1}$ (plain line). The top row represents the data association. The bottom is the results after minimising point-to-plane distances. A uses 360° frames with back-association. B uses frames greater than 360°, with back-association. C extends B with back-and-forth-association. C ensure consistency of the lidar scans whereas A and B do not.

B. Data association

Scan registration requires matching features from frame-to-frame. For a pair of lidar frames i and k, after reprojecting both frames into $\mathfrak{F}_W$ and for each point of $\mathcal{P}^i$, the method looks for the 3 nearest neighbours in $\mathcal{P}^k$. For points in $\mathcal{E}_I^i$ and $\mathcal{E}_O^i$, only the 2 nearest neighbours are searched respectively in $\mathcal{E}_I^k$ and $\mathcal{E}_O^k$. The Kd-tree implementation in PCL [15] is used for the nearest neighbour searches. Thresholds are applied on matching distances to validate a data association and therefore remove some outliers. The collinearity of the 3 closest feature-points of a plane feature is checked for the same reasons. All the valid associations are included in $\mathcal{A}$ as tuples of 3 or 4 depending on the type of feature matched.

We consider scans greater than 360° (520° in our implementation) and do the data association both from i to k and from k to i, to ensure consistency of the lidar scans. Fig. 5 shows the motivation for such a choice through a simplified 2D example.

C. Back-end/front-end integration

The computation of the feature scores, therefore the data association, depends on the knowledge of the IMU poses and velocities to undistort the point clouds. These poses and velocities are part of $\mathcal{S}$. As a consequence, if the state changes, the data association might not be relevant any more. Hence, the proposed method relies on a strong integration between front-end and back-end in order to re-compute the lidar features and data association as the state changes over time during the batch optimisation.

V. Experiments and Results

Experiments on simulated and real data have been conducted to demonstrate the performances of the proposed method. The full framework has been implemented in C++ using Ceres [16] for the on-manifold optimisation.

A. Simulation

A sensing system (IMU and lidar with given extrinsic calibration) moving according to predefined trajectories in a virtual room (constituted of 7 planes) has been simulated. All the simulated trajectories have a duration of 14.5s and have been generated from sine functions with random frequencies and amplitudes.

The simulated sensing system has been modelled to match the characteristics of the system used in our experimental results (Section V-B):

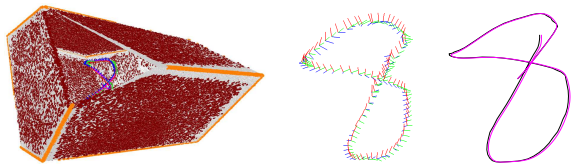

Fig. 6: Estimated map and trajectory in the simulated environment. Left: Estimated map of the 7-sided room with a corner cropped out to show the trajectory inside. Centre: Estimated trajectory (position and rotation). Right: Estimated position in black against ground truth in pink. The average distance travelled in each simulation is $27.32m$.

- 16-channel ($\pm 15°$) lidar rotating at 10Hz with a density of 300k point per second and noise of $\pm 3cm$.
- 3-axis accelerometer and 3-axis gyroscope sampling at 100Hz with noise of $0.02m/s^2$ and $0.097°/s$.

The extrinsic calibration between sensors is randomly generated for each run. The results are evaluated over a 10-run Monte Carlo simulation.

For this evaluation, the framework uses feature-matching with 4 previous frames. In other words, each $\mathcal{X}^m$ is matched individually with $\mathcal{X}^{m-i}$ with $i = 1, \cdots, 4$. Although there are multiple links in between the factor graph nodes, we call this set-up "odometry", as there are no explicit loop closures.

Among all the trials, the average velocity is $1.88m/s$ and the maximum velocity is $4.96m/s$. The Monte Carlo runs show an average final position error of $0.32m \pm 0.31$ and an average final orientation error of $0.39° \pm 0.28$. Divided by the distance travelled in each of the simulations, the relative position error is $1.33\% \pm 1.2$. Fig. 6 shows one of the simulation trials. These results show the robustness of our estimation method in the presence of fairly aggressive motion, both regarding linear and angular velocities. Through this set-up, we see a small drift as in general odometry systems. Although it is a simulated setting, data association is not given. Therefore the front-end might contain outliers. Planar features close to edges can be misassociated with the planar points from the neighbouring plane. This set-up shows the need for a more robust outlier rejection mechanism but demonstrates the rightfulness of our back-end and the observability of the state.

B. Real data

The hardware used for the real data experiments comprises a Velodyne VLP-16 and a low-cost Xsens MTi3 IMU. The *snark* driver[6] and the ROS Xsens driver[7] were used to collect the lidar and IMU data respectively. Lidar points and IMU measurements were logged with their associated timestamps. Note that there is no explicit mechanism for synchronisation between lidar and IMU data.

1) Odometry: This set-up aims to benchmark our method against a constant velocity feature matching framework. The method in [3] was chosen due to its top performance for lidar systems in the odometry benchmark of Kitti dataset [17]. Multiple environments are used for this evaluation:

[6]https://github.com/acfr/snark
[7]http://wiki.ros.org/xsens_driver

	Mean point-to-plane distance (mm)	
	[3]	IN2LAMA
a)	20.7	13.0
b)	69.8	12.3

TABLE I: Quantitative comparison on datasets a) and b). The errors displayed correspond to the mean point-to-plane distances between the floor points and the floor plane.

a) Lab environment with smooth trajectory
b) Lab environment with "dynamic" trajectory
c) Staircase in between two floors

All the datasets were collected walking at around $1.2m/s$.

In the first dataset (Fig. 7a)), both [3] and our method perform relatively well producing similar results. This dataset was collected in such a way that the constant velocity assumption made in [3] is fairly respected. In the second dataset, as the movement alternates between acceleration and deceleration, the constant velocity assumption is not respected. Fig. 7b) shows that our method can handle such movement where the map generated by [3] becomes blurry.

The staircase dataset is very challenging for both methods as per the nature of the sensor used. At a point in the recording, the information contained in the lidar scans is very little. It creates a lack of geometrical constraints in between two consecutive scans. Therefore, as shown in Fig. 7c), both methods drift.

From the map generated on datasets a) and b), we manually segmented the floor points. From these floor point clouds, we ran a RANSAC-based plane fitting algorithm [18]. The values shown in Table I correspond to the mean point-to-plane distance between each floor points and the fitted plane model.

The results presented here can be explained by two main factors. On one hand, [3] is designed for real-time operations and uses a constant velocity model, on the other hand, our method does not formulate any assumptions about the sensors' motion and both the trajectory and map are simultaneously estimated through a unique batch optimisation. These results show the advantage provided by the UPMs and a batch optimisation to build accurate maps. In fairness, however, our framework uses the extra information given by the IMU, as opposed to only the information provided by the constant velocity assumption in [3].

2) Loop-closure: The aim of this experiment is to show the ability of our framework to be used in a full simultaneous localisation and mapping configuration and not only in an odometry-like one. The sensing system was hand-held while walking around an office environment with an average velocity of $1.3m/s$ and peaks at $1.7m/s$. The trajectory followed a loop in which the starting and ending points were approximately at the same location. We manually set a loop closure between the last and the first pose estimated. Fig. 8 shows the effectiveness of the loop closure by removing "double-walls" from the map, highlighting the importance of loop closure for consistent map building.

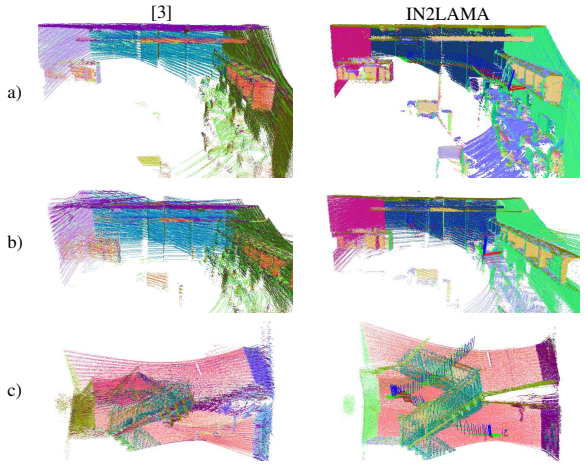

Fig. 7: Comparison between [3] and our method IN2LAMA on three different datasets. Once the maps are created, normals are computed using the 100 nearest points. These normals are used to colourise the points for the sake of visibility.

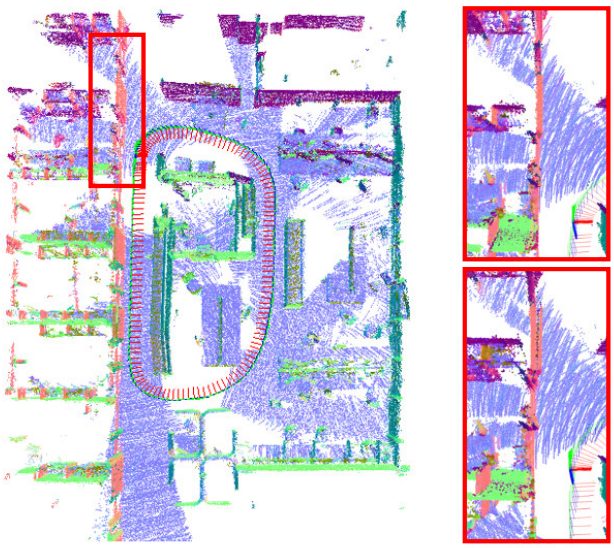

Fig. 8: Loop-closure demonstration. The image on the left represents the full map with a loop closure. Images on the right correspond to the area designated with the red rectangle in the full map after (top) and before (bottom) loop closure. The loop closure removes the "double-wall".

VI. Conclusion

This paper presents a novel probabilistic framework for INertial Lidar Localisation And MApping. It uses preintegration over upsampled IMU readings to characterise the motion distortion naturally present in spinning lidar scans. The off-line estimation of accurate maps is done via a full batch on-manifold optimisation without the need for an explicit motion-model. The effectiveness of the proposed method is demonstrated using simulated and real data.

Future work includes the improvement of the front-end with more robust data associations and automatic loop detections. We are also interested in including frame-to-map constraints. Therefore we will consider the use of surfels or other map representations to be incorporated into IN2LAMA framework.

REFERENCES

[1] S. Hong, H. Ko, and J. Kim, "VICP: Velocity updating iterative closest point algorithm," *Proceedings - IEEE International Conference on Robotics and Automation*, no. Section 3, pp. 1893–1898, 2010.

[2] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," *Robotics: Science and Systems*, vol. 5, pp. 168–176, 2009.

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

[4] M. Bosse and R. Zlot, "Continuous 3D Scan-Matching with a Spinning 2D Laser," *IEEE International Conference on Robotics and Automation*, 2009.

[5] 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.

[6] S. Anderson and T. D. Barfoot, "Full STEAM ahead: Exactly sparse Gaussian process regression for batch continuous-time trajectory estimation on SE(3)," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2015-Decem, no. 3, pp. 157–164, 2015.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan, "Elastic LiDAR Fusion: Dense Map-Centric Continuous-Time SLAM," *IEEE International Conference on Robotics and Automation*, 2018.

[12] D. Droeschel and S. Behnke, "Efficient Continuous-time SLAM for 3D Lidar-based Online Mapping," *IEEE International Conference on Robotics and Automation (ICRA)*, no. May, pp. 5000–5007, 2018.

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

[14] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1280–1286, 2013.

[15] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[16] S. Agarwal and K. Mierle, "Ceres Solver," *http://ceres-solver.org*.

[17] 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.

[18] P. H. S. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 1996.