IMU preintegration for 2D SLAM problems using Lie Groups

Idril Geer, Joan Vallvé and Joan Solà

***Abstract*— 2D SLAM is useful for mobile robots that are constrained to a 2D plane, for example in a warehouse, simplifying calculations in respect to the 3D case. The use of an IMU in such a context can enrich the estimation and make it more robust. In this paper we reformulate the IMU preintegration widely used in 3D problems for the 2D case, making use of Lie Theory. The Lie theory based formalization, first derived for a perfectly horizontal plane, allows us to easily extend it to problems where the plane is not orthogonal to the gravity vector. We implement the theory in a factor graph based estimation library, and carry out experiments to validate it on a mobile platform. Two experiments are carried out; on a horizontal and a sloped environment, and the sensor data is processed using our two 2D methods and a state-of-the-art 3D method.**

I. INTRODUCTION

Inertial Measurement Units (IMUs) can be useful as add-on sensors to enhance the richness of the information the robot has available for Simultaneous Localization And Mapping (SLAM), and also to add robustness to the estimation, especially during highly dynamic events such as slippage. They work at higher frequencies than most other sensors, which introduces the need for IMU preintegration. The problem of IMU preintegration in 3D is well studied [1]–[4] using Lie theory, but to our knowledge hasn't been developed for the 2D case. In this paper we explore the 2D formulation using Lie Theory and extend it to planar spaces on a slope. We start by considering the case where the 2D plane is perfectly horizontal, after that we move on to the case where the plane is on an arbitrary slope.

Inertial measurements have been incorporated to SLAM for some time now. One of the first examples is [5] which fuses IMU data with monocular vision using EKF. A related work [6] used stereovision visual odometry with IMU, and was fused in a loosely coupled manner using also EKF.

More modern works in SLAM use smoothing over a trajectory of keyframes for estimation, instead of EKF [7]. Incorporating IMU to these systems was possible thanks to the preintegration methods pioneered by Lupton [1], and improved later by Forster [2] with the use of Lie theory for the rotational part $SO(3)$. Indeed, the usage of Lie-theoretic approaches improves performances and yields more elegant designs. Barrau [8] proposed the $SE_2(3)$ matrix group which is well adapted to the IMU as it considers not only the orientation but also translation and velocity. Using the terminology introduced by Sola in [9], this $SE_2(3)$ group forms what we may call here a *compact* Lie group, as

The authors are with Mobile Robotics Laboratory, Institut de Robòtica i Informàtica Industrial, Universitat Politècnica de Catalunya, Barcelona, Spain {igeer,jvallve,jsola}@iri.upc.edu

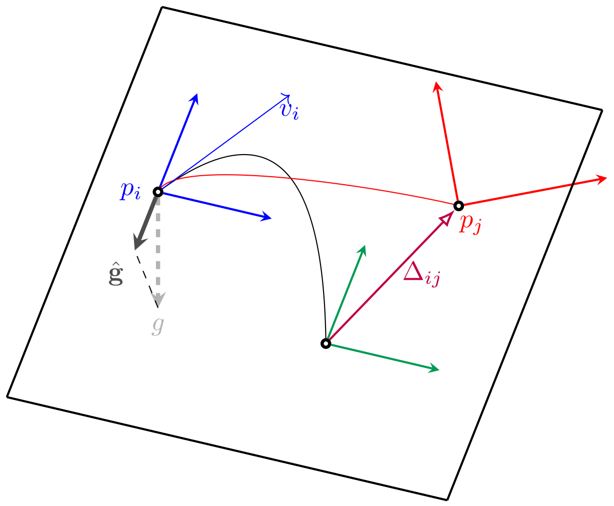

Fig. 1: In the sloped plane case, the free-falling, non-rotating frame follows a parabolic trajectory (grey) governed by the projected gravity $\hat{\mathbf{g}}$ and determined by the initial conditions p_i, v_i and R_i at time i (blue). The IMU delta Δ_{ij} between times i and j is defined as the state of the IMU at time j (red) expressed in the free-falling, non-rotating frame at time j (green).

opposed to the *composite* form which would be the one in [2]. Fourmy [4] proposed a similar matrix group for the IMU, which incorporates also the notion of time, thus leading to a more compact formulation of the algorithms.

Recently, other SLAM works such as [10] have considered a continuous-time representation of the trajectory using splines on the appropriate manifolds, thus circumventing the need for IMU preintegration. These systems come however at other costs and we do not contemplate them here.

Regarding planar or 2D implementations, inertial measurements have been used too but in a much simpler fashion. Some low-cost differential drive robots use wheel odometry only for measuring the translational motion, the rotational part being provided by a single-axis gyroscope. The bias of this gyro is calibrated each time the robot is stopped. Other works like [11] use the gyro to identify and correct non-systematic error sources like bumps and wheel slippage.

In this work, we present a preintegration scheme to incorporate 2D IMU data (accelerations in the plane and yaw rates) into a keyframe-based SLAM. We take for this a Lie-theoretic approach in the flavor of [9] and draw from the IMU works of Atchuthan [3] and Fourmy [4]. We propose a matrix Lie group comprising 2D translation, velocity, rotation and time. To study the validity and performance of this approach we incorporate the pre-integrated factors to a

Lidar, ICP-based, pose-graph SLAM. We compare it against our implementation of 3D preintegration [2], applied to 2D with the addition of planar constraints, observing a much quicker convergence of the biases for the 2D formulation, leading to better accuracy at a much lower implementation cost. We also present a variation of the system for non-horizontal terrain, where the gravity vector projects to non-zero values of acceleration in the plane, and show that we are this way able to observe the terrain's inclination.

II. SLAM AND IMU PREINTEGRATION

A SLAM smoothing problem can be modeled as the following non-linear least-squares optimization problem:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \|\mathbf{e}_k(\mathbf{x})\|^2_{\boldsymbol{\Omega}_k^{-1}} \tag{1}$$

where the tuples $\{\mathbf{e}(\mathbf{x}), \boldsymbol{\Omega}_k^{-1}\} := \Phi$ define the factors associated to each one of the sensor measurements.

In this context, IMU factors establish relative transformations between poses with velocity at consecutive keyframes. Here arises the need to preintegrate the IMU data, as a huge amount of measurements can be received between two consecutive keyframes, so we need to avoid having to reintegrate after each optimization step.

This section is devoted to the discussion of currently available methods for 3D IMU preintegration. It revolves around concepts proper to Lie theory, with key considerations regarding the representation of uncertainty and therefore the way to compute Jacobians for its propagation. We then propose our design for a 2D implementation.

A. *Generalized motion preintegration on Lie groups*

We follow the generalized preintegration theory described in [3, Chapter 4] and exploited in [12], [13], which is sketched below.

Let us consider a set of keyframes $\mathbf{x}_i$, each defined as the state of the robot at time t_i. We refer by deltas to the motion quantities $\boldsymbol{\Delta}_{im}$ linking two consecutive keyframes $\{\mathbf{x}_i, \mathbf{x}_m\}$. This relation is expressed through the operators $\boxminus$ and $\boxplus$,

$$\boldsymbol{\Delta}_{im} := \mathbf{x}_m \boxminus \mathbf{x}_i \tag{2}$$
$$\mathbf{x}_m := \mathbf{x}_i \boxplus \boldsymbol{\Delta}_{im}. \tag{3}$$

These operators must be wisely defined so that the integral of the motion data from time i to m depends only on the stream of data $\mathbf{U}_{im} = \{\mathbf{u}_{i+1}, \mathbf{u}_{i+2}, \cdots, \mathbf{u}_m\}$ and calibration parameters $\mathbf{b}_i$, but not on the initial state $\mathbf{x}_i$,

$$\boldsymbol{\Delta}_{im}(\mathbf{b}_i, \mathbf{U}_{im}) = f(\mathbf{b}_i, \mathbf{U}_{im}) \perp\!\!\!\perp \mathbf{x}_i. \tag{4}$$

This usually boils down to defining the delta $\boldsymbol{\Delta}_{im}$ as the motion between a frame at time m that has evolved from time i following a trajectory for which the sensor would have sensed no motion, and the current state at time m. In the case of the IMU in 3D, this reference frame is the free-falling, non rotating frame [3].

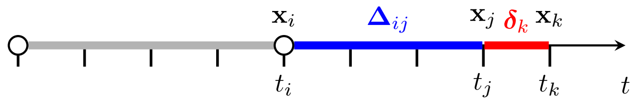

Fig. 2: The delta $\boldsymbol{\Delta}_{ij}$ contains the information of the movement from the keyframe $\mathbf{x}_i$ until time t_j, while the small delta $\boldsymbol{\delta}_k$ contains the information between $\mathbf{x}_j$ and $\mathbf{x}_k$. The IMU's sampling time is $\delta t = t_k - t_j$.

1) Front end: preintegration: Interestingly, the space of deltas has the structure of a Lie group. Motion preintegration can therefore be explained in terms of Lie groups, starting at the group identity $\boldsymbol{\Delta}_{ii} = \boldsymbol{\Delta}_{\mathcal{E}}$ and advancing incrementally at each new step from j to k via the group composition $\circ$ (see Fig. 2),

$$\boldsymbol{\Delta}_{ik} = \boldsymbol{\Delta}_{ij} \circ \boldsymbol{\delta}_k, \tag{5}$$

where $\boldsymbol{\delta}_k$ is referred to as the current delta at time k. This can be expressed as the exponential of a group velocity vector (a vector in the tangent space of the group) times the sampling time δt. This tangent vector is obtained from the measured data $\mathbf{u}_k$ calibrated using the calibration vector $\mathbf{b}$, which we sample at time i. The current delta is thus,

$$\boldsymbol{\delta}_k = \mathrm{Exp}((\mathbf{u}_k - \mathbf{b}_i)\delta t). \tag{6}$$

where $\mathrm{Exp}()$ is the retraction map for the Lie Algebra, using the notation found in [9].

It is important to remark that, given that this exponential is composed on the right side in (5), the involved tangent vectors $\mathbf{u}$ and $\mathbf{b}$ are expressed in the local frame represented by $\boldsymbol{\Delta}_{ij}$. This allows us to easily identify them with the sensor measurement and bias. Summarizing, the pre-integrated delta between times i and m can be expressed as

$$\boldsymbol{\Delta}_{im}(\mathbf{b}_i, \mathbf{U}_{im}) = \prod_{k=i+1}^{m} \boldsymbol{\delta}_k = \prod_{k=i+1}^{m} \mathrm{Exp}\left((\mathbf{u}_k - \mathbf{b}_i)\delta t\right), \tag{7}$$

where $\prod$ expresses repeated composition with $\circ$.

In practice, this integral is performed using the currently available value of the calibration parameters, $\mathbf{b}_i = \overline{\mathbf{b}}_i$, which is a constant, obtaining a pre-integrated delta that only depends on the data,

$$\overline{\boldsymbol{\Delta}}_{im} := \boldsymbol{\Delta}(\overline{\mathbf{b}}_i, \mathbf{U}_{im}). \tag{8}$$

To complete the preintegration, the delta is accompanied by its Jacobian with respect to the calibration parameters, $\mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}} := \partial\boldsymbol{\Delta}/\partial\mathbf{b}$, and the delta covariance, $\boldsymbol{\Sigma}_{im}$. These are also pre-integrated incrementally at each time k,

$$\mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\Delta}_{ik}} = \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\Delta}_{ij}} + \mathbf{J}_{\boldsymbol{\delta}_k}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\delta}_k} \tag{9}$$
$$\boldsymbol{\Sigma}_{ik} = \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}} \boldsymbol{\Sigma}_{ij} \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}\top} + \mathbf{J}_{\mathbf{u}_k}^{\boldsymbol{\Delta}_{ik}} \boldsymbol{\Sigma}_{\mathbf{u}} \mathbf{J}_{\mathbf{u}_k}^{\boldsymbol{\Delta}_{ik}\top}, \tag{10}$$

where $\boldsymbol{\Sigma}_{\mathbf{u}}$ is the IMU's measurements noise covariance, and the different Jacobians are obtained from the Jacobian blocks of (5–6), computed according to Lie theory [9], and applying the chain rule. See Section III-D further on for important considerations regarding these Jacobians.

2) Back-end: factor evaluation: The quantities $\overline{\mathbf{\Delta}}_{im}$, $\mathbf{\Sigma}_{im}$, $\mathbf{J}^{\mathbf{\Delta}}_{\mathbf{b}}$ and $\overline{\mathbf{b}}_i$ are used to define the preintegration factor, which will be repeatedly evaluated by the back-end or solver. At each evaluation the pre-integrated delta is corrected for updated values of the calibration $\mathbf{b}_i \neq \overline{\mathbf{b}}_i$ with the help of the pre-integrated Jacobian $\mathbf{J}^{\mathbf{\Delta}}_{\mathbf{b}}$,

$$\mathbf{\Delta}_{im} = \overline{\mathbf{\Delta}}_{im} \oplus \mathbf{J}^{\mathbf{\Delta}}_{\mathbf{b}}(\mathbf{b}_i - \overline{\mathbf{b}}_i), \tag{11}$$

where the operator $\oplus$ retracts vectors from the tangent space to the group as explained in [9]. The error is computed using the reciprocal operator $\ominus$ between this corrected delta and a predicted delta $\widehat{\mathbf{\Delta}}_{im} = \mathbf{x}_m \boxminus \mathbf{x}_i$, yielding the final expression

$$\mathbf{e}_{im}(\mathbf{x}_i, \mathbf{x}_m, \mathbf{b}_i) = \big(\overline{\mathbf{\Delta}}_{im} \oplus \mathbf{J}^{\mathbf{\Delta}}_{\mathbf{b}}(\mathbf{b}_i - \overline{\mathbf{b}}_i)\big) \ominus (\mathbf{x}_m \boxminus \mathbf{x}_i), \tag{12}$$

which is accompanied by the information matrix $\mathbf{\Omega}^I_{im} := (\mathbf{\Sigma}_{im})^{-1}$ to form the quadratic costs in (1).

B. IMU specializations

Any implementation of this algorithm must specify: the delta group and tangent structures; all the operators $\boxplus$, $\boxminus$, $\circ$, $\oplus$ and $\ominus$; the mapping Exp; and all their Jacobians.

In particular, [2] uses a composite definition of the group $\mathbf{\Delta} = [\Delta\mathbf{p}, \Delta\mathbf{v}, \Delta\mathbf{R}]$ with translation $\Delta\mathbf{p}$ and velocity $\Delta\mathbf{v}$ treated as vector spaces, and rotations $\Delta\mathbf{R}$ as members of $SO(3)$. Ref. [8] uses the compact matrix Lie group $SE_2(3)$ containing the same parts, thus improving the interplay between them. In both, the passage of time δt in the integration step (5) is introduced as an extra parameter out of the group. Finally, [4] also uses a compact Lie group including time, therefore containing all the parts needed for the preintegration. In this paper, we draw from this last design and adapt it to 2D, as described in the following section.

III. IMU PREINTEGRATION IN 2D

We consider here the case of a perfectly horizontal plane, and will extend it to sloped terrain in the next chapter. This constitutes a simple case because it eliminates the effects of gravity, as gravity is orthogonal to the plane. Notice that this does not assume a perfectly aligned IMU, as any (small) misalignment will be absorbed by the IMU's bias estimation.

A. Delta definition

We define the deltas $\mathbf{\Delta} = [\Delta\mathbf{p}, \Delta\mathbf{v}, \Delta\mathbf{R}]$ as follows: Consider keyframes $\mathbf{x}_i = [\mathbf{p}_i, \mathbf{v}_i, \mathbf{R}_i]$ and $\mathbf{x}_j$. Consider also a reference frame $\mathcal{F}_t$ that started at $\mathcal{F}_i = \mathbf{x}_i$ and evolved from times i to j following a trajectory that is such that a perfect IMU without bias and noise would produce null data. In the 3D case this trajectory is that of a free-falling, non rotating frame [3], [4]. Since in our 2D case the gravity effects are absent, this reduces to a constant velocity without rotation. Thus at time j we have $\mathcal{F}_j = [\mathbf{p}_i + \mathbf{v}_i\Delta t_{ij},\ \mathbf{v}_i,\ \mathbf{R}_i]$. The deltas are finally defined as the state $\mathbf{x}_j$ expressed with respect to $\mathcal{F}_j$, yielding the definition of the operators $\boxminus, \boxplus$ as:

$$\mathbf{\Delta}_{ij} = \mathbf{x}_j \boxminus \mathbf{x}_i := \begin{cases} \Delta\mathbf{p}_{ij} &= \mathbf{R}_i^\top(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij}) \\ \Delta\mathbf{v}_{ij} &= \mathbf{R}_i^\top(\mathbf{v}_j - \mathbf{v}_i) \\ \Delta\mathbf{R}_{ij} &= \mathbf{R}_i^\top\mathbf{R}_j \end{cases} \tag{13}$$

$$\mathbf{x}_j = \mathbf{x}_i \boxplus \mathbf{\Delta}_{ij} := \begin{cases} \mathbf{p}_j &= \mathbf{p}_i + \mathbf{v}_i\Delta t_{ij} + \mathbf{R}_i\Delta\mathbf{p}_{ij} \\ \mathbf{v}_j &= \mathbf{v}_i + \mathbf{R}_i\Delta\mathbf{v}_{ij} \\ \mathbf{R}_j &= \mathbf{R}_i\Delta\mathbf{R}_{ij} \end{cases} \tag{14}$$

The delta blocks $\Delta\mathbf{p}, \Delta\mathbf{v}, \Delta\mathbf{R}$ can be organized in different Lie group flavors by defining the group composition. For the sake of reference, the classical flavor 2D-equivalent to [2] would deal with the blocks directly, having composition $\circ$ defined as:

$$\mathbf{\Delta}_{ij} \circ \mathbf{\Delta}_{jk} := \begin{cases} \Delta\mathbf{p}_{ik} &= \Delta\mathbf{p}_{ij} + \Delta\mathbf{v}_{ij}\Delta t_{jk} + \Delta\mathbf{R}_{ij}\Delta\mathbf{p}_{jk} \\ \Delta\mathbf{v}_{ik} &= \Delta\mathbf{v}_{ij} + \Delta\mathbf{R}_{ij}\Delta\mathbf{v}_{jk} \\ \Delta\mathbf{R}_{ik} &= \Delta\mathbf{R}_{ij}\Delta\mathbf{R}_{jk} \end{cases} \tag{15}$$

identity $\mathbf{\Delta}_{\mathcal{E}} = [\mathbf{0}, \mathbf{0}, \mathbf{I_2}]$, and inverse $\mathbf{\Delta}^{-1}$ defined by:

$$\mathbf{\Delta}^{-1} := \begin{cases} \Delta\mathbf{p}^{-1} &= -\Delta\mathbf{R}^\top(\Delta\mathbf{p} - \Delta\mathbf{v}\Delta t) \\ \Delta\mathbf{v}^{-1} &= -\Delta\mathbf{R}^\top\Delta\mathbf{v} \\ \Delta\mathbf{R}^{-1} &= \Delta\mathbf{R}^\top \end{cases} \tag{16}$$

Further, it defines the current delta $\boldsymbol{\delta}$ from the unbiased 2D IMU data $(\mathbf{a}, \omega)$ through the mapping

$$\boldsymbol{\delta} = \begin{cases} \delta\mathbf{p} &= \frac{1}{2}\mathbf{a}\delta t^2 \\ \delta\mathbf{v} &= \mathbf{a}\delta t \\ \delta\mathbf{R} &= \exp\big([\omega\delta t]_\times\big) \end{cases} \tag{17}$$

leading when considering the bias to the recursive integration scheme,

$$\begin{aligned} \Delta\mathbf{p}_{ik} &= \Delta\mathbf{p}_{ij} + \Delta\mathbf{v}_{ij}\delta t + \frac{1}{2}\Delta\mathbf{R}_{ij}(\mathbf{a}_k - \mathbf{a}_{bi})\delta t^2 \\ \Delta\mathbf{v}_{ik} &= \Delta\mathbf{v}_{ij} + \Delta\mathbf{R}_{ij}(\mathbf{a}_k - \mathbf{a}_{bi})\delta t \\ \Delta\mathbf{R}_{ik} &= \Delta\mathbf{R}_{ij}\,\mathrm{Exp}((\boldsymbol{\omega}_k - \omega_{bi})\delta t) \end{aligned} \tag{18}$$

where $\mathbf{a}_{bi}$ and ω_{bi} are the accelerometer and gyroscope biases respectively.

With these definitions and the generic algorithm presented earlier, it is possible to build a functioning 2D preintegration scheme. However, we opt for a compact Lie group flavor as described below.

B. The 2D IMU matrix Lie Group

The derivation of the deltas is done in the same way as in [4], but in 2D instead of 3D.

We can instead think of the IMU deltas as matrices of the following form:

$$\mathbf{\Delta} = \begin{bmatrix} \Delta\mathbf{R} & \Delta\mathbf{v} & \Delta\mathbf{p} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{19}$$

It is easy to see that the previous composition rule $\circ$ (15) coincides with the matrix product when we add the Δt

element with a sum as composition. The inverse (16) and identity also coincide, with $\mathbf{\Delta}_{\mathcal{E}} = \mathbf{I}_4$.

To characterize the tangent space we follow the method described in [9]. For this, we define the instant rates $\mathbf{v} := \frac{\partial \Delta \mathbf{p}}{\partial t}$, $\mathbf{a} := \frac{\partial \Delta \mathbf{v}}{\partial t}$, $s := \frac{\partial \Delta t}{\partial t}$, and the skew-symmetric matrix $[\omega]_\times := \Delta \mathbf{R}^\top \dot{\Delta \mathbf{R}}$. With these we find that the tangent elements satisfy:

$$\boldsymbol{\tau}^\wedge = \mathbf{\Delta}^{-1}\dot{\mathbf{\Delta}} = \begin{bmatrix} [\omega]_\times & \Delta\mathbf{R}^\top \mathbf{a} & \Delta\mathbf{R}^\top(\mathbf{v} - s\Delta\mathbf{v}) \\ 0 & 0 & s \\ 0 & 0 & 0 \end{bmatrix} \tag{20}$$

where $[\omega]_\times$ is the 2×2 skew-symmetric matrix. By looking at (20) around the identity, $\mathbf{\Delta} = \mathbf{\Delta}_{\mathcal{E}}$, we find the Lie algebra's structure $\boldsymbol{\tau}^\wedge$ and its isomorphic Cartesian representation $\boldsymbol{\tau}$:

$$\boldsymbol{\tau}^\wedge = \dot{\mathbf{\Delta}}|_{\mathbf{\Delta}=\mathbf{I}} = \begin{bmatrix} [\omega]_\times & \mathbf{a} & \mathbf{v} \\ 0 & 0 & s \\ 0 & 0 & 0 \end{bmatrix} \quad , \quad \boldsymbol{\tau} = \begin{bmatrix} \mathbf{v} \\ \mathbf{a} \\ \omega \\ s \end{bmatrix} \tag{21}$$

We now have from (20) an ODE $\forall \mathbf{\Delta}$: $\dot{\mathbf{\Delta}}(t) = \mathbf{\Delta}(t)\boldsymbol{\tau}^\wedge$. For a small enough time lapse where we can take $\boldsymbol{\tau}^\wedge$ as constant this solves to:

$$\mathbf{\Delta}(t) = \mathbf{\Delta}(0)\exp(\boldsymbol{\tau}^\wedge \cdot t) = \mathbf{\Delta}(0)\,\mathrm{Exp}(\boldsymbol{\tau}\cdot t) \tag{22}$$

where $\mathbf{\Delta}(0) = \mathbf{I}$ and Exp is developed below.

Defining for convenience $[\theta]_\times := [\omega]_\times \Delta t$, $\nu := \mathbf{a}\Delta t$, $\rho := \mathbf{v}\Delta t$ we find the following closed form for the exponential:

$$\mathbf{\Delta}(\Delta t) = \begin{bmatrix} \mathcal{R} & \mathcal{Q}\nu & \mathcal{Q}\rho + s\mathcal{P}\nu\Delta t \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{23}$$

where $\mathcal{R} := \sum_{k\geq 0} \frac{1}{k!}[\theta]_\times^k = \exp([\theta]_\times)$, $\mathcal{Q} := \sum_{k\geq 1}\frac{1}{k!}[\theta]_\times^{k-1}$, and $\mathcal{P} := \sum_{k\geq 2}\frac{1}{k!}[\theta]_\times^{k-2}$. The closed forms for $\mathcal{R}$, $\mathcal{Q}$ and $\mathcal{P}$ can easily be found to be:

$$\mathcal{R} = \cos(\theta)I + \sin(\theta)[1]_\times = \mathbf{R}(\theta) \tag{24}$$

$$\mathcal{Q} = \frac{\sin(\theta)}{\theta}I + \frac{1-\cos(\theta)}{\theta}[1]_\times \tag{25}$$

$$\mathcal{P} = \frac{1-\cos(\theta)}{\theta^2}I + \frac{\theta - \sin(\theta)}{\theta^2}[1]_\times \tag{26}$$

where $\mathbf{R}(\theta)$ is the rotation matrix of angle θ.[1].

1) Incorporation of the IMU measurements: The IMU gives us its linear acceleration $\mathbf{a}$ and angular velocity ω, which are the same instant variations defined previously to characterize the tangent space. The previous closed form of the exponential depends on these two magnitudes, as well as on $\mathbf{v}$, which is not part of the IMU's measurements. Let's remember though that for the ODE integration we've supposed that $\boldsymbol{\tau}$ is constant, thus $\mathbf{v}$ is constant too. As the deltas are defined as the motion relative to the non-rotating constant velocity frame defined at the start of the integration period, and at this time the relative velocity is zero by definition, we conclude that $\mathbf{v} = 0$ during the full integration step.

Let us also consider the meaning of $s = \frac{\partial \Delta t}{\partial t}$: it gives us the speed at which Δt evolves, so if we take $s = 1$ we're saying that Δt evolves as fast as our time t.

We can thus write the tangent vectors measured by the IMU in the following way:

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{0} & \mathbf{a} & \omega & 1 \end{bmatrix}^\top \tag{27}$$

In this case, through the expression of the exponential our delta is:

$$\mathbf{\Delta}(\Delta t) = \mathrm{Exp}(\boldsymbol{\tau}\Delta t) = \begin{bmatrix} \mathcal{R} & \mathcal{Q}\nu & \mathcal{Q}\rho \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{28}$$

C. Incremental preintegration in the Lie group

As seen before, the incremental delta preintegration is given by the group composition. In terms of the Lie exponential:

$$\mathbf{\Delta}_{ik} = \mathbf{\Delta}_{ij} \circ \boldsymbol{\delta}_k = \mathbf{\Delta}_{jk} \circ \mathrm{Exp}(\boldsymbol{\tau}_k \boldsymbol{\delta} t) \tag{29}$$

Where $\boldsymbol{\tau}_k = [\mathbf{0}, \mathbf{a}_k - \mathbf{a}_{bi}, \omega_k - \omega_{bi}, 1]^\top$ is in the tangent space of $\mathbf{\Delta}_{ij}$ expressed locally. This preintegration depends only on the IMU's measurements $(\mathbf{a}_k, \omega_k)$, biases $(\mathbf{a}_{bi}, \omega_{bi})$ and on the sampling time $\boldsymbol{\delta} t$.

D. Jacobians

The trivial Jacobians are the ones related to the IMU measurements, biases and noise: $\mathbf{J}^{\mathbf{b}}_{\mathbf{b}_m} = I_3$ and $\mathbf{J}^{\mathbf{b}}_{\mathbf{b}_b} = \mathbf{J}^{\mathbf{b}}_{\mathbf{b}_n} = -I_3$, where $\mathbf{b}_m = (\mathbf{a}_m, \omega_m)$, $\mathbf{b}_b = (\mathbf{a}_b, \omega_b)$ and $\mathbf{b}_n = (\mathbf{a}_n, \omega_n)$.

The Jacobian of the current delta in respect to the tangent vector is

$$\mathbf{J}^{\mathbf{\Delta}}_{\boldsymbol{\tau}} = \begin{bmatrix} \mathcal{Q}\delta t & s\mathcal{P}\delta t^2 & \mathcal{Q}'[1]_\times \mathbf{v}\delta t^2 + \mathcal{P}'[1]_\times \mathbf{a}\delta t^3 & 0 \\ 0 & \mathcal{Q}\delta t & \mathcal{Q}'[1]_\times \mathbf{a}\delta t^2 & 0 \\ 0 & 0 & -\mathcal{R}\delta t^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{30}$$

and in respect to the body magnitudes vector it's

$$\mathbf{J}^{\mathbf{\Delta}}_{\mathbf{b}} = \begin{bmatrix} \mathcal{P}\delta t^2 & \mathcal{P}'[1]_\times \mathbf{a}\delta t^2 \\ \mathcal{Q}\delta t & \mathcal{Q}'[1]_\times \mathbf{a}\delta t \\ 0 & -\mathcal{R}\delta t \\ 0 & 0 \end{bmatrix} \tag{31}$$

For the composition $\mathbf{\Delta}_{ik} = \mathbf{\Delta}_{ij}\boldsymbol{\delta}_{jk}$, we have the following Jacobians:

$$\mathbf{J}^{\mathbf{\Delta}_{ik}}_{\mathbf{\Delta}_{ij}} = \begin{bmatrix} I_2 & I_2\delta t_{jk} & \boldsymbol{\delta}\mathbf{p}_{jk} & 0 \\ 0 & I_2 & \boldsymbol{\delta}\mathbf{v}_{jk} & 0 \\ 0 & 0 & \boldsymbol{\delta}\mathbf{R}_{jk} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{32}$$

$$\mathbf{J}^{\mathbf{\Delta}_{ik}}_{\boldsymbol{\delta}_{jk}} = \begin{bmatrix} \Delta\mathbf{R}_{ij} & 0 & 0 & \Delta\mathbf{v}_{ij} \\ 0 & \Delta\mathbf{R}_{ij} & 0 & 0 \\ 0 & 0 & \Delta\mathbf{R}_{ij} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{33}$$

[1]If we were to apply the small angle approximation on these functions we would get the current delta described in (17)

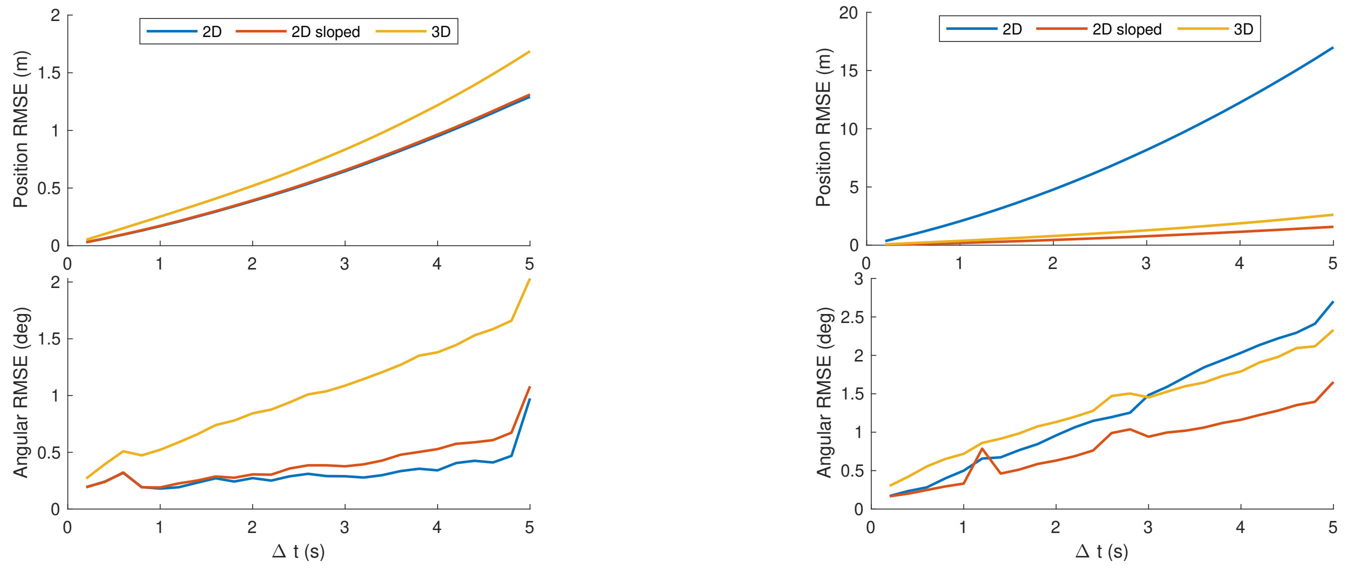

Fig. 3: Root Mean Squared Errors (RMSE) of the compared preintegration methods respect to the integration time δt in the horizontal test (left) and sloped test (right).

IV. IMU Preintegration in 2D on a Slope

The IMU delta definitions and equations for a sloped surface are very similar to the 3D case. We add to the state a gravity vector $\hat{\mathbf{g}}$ that corresponds to the real gravity projected on the plane in which the robot moves. The equations used then are the same as in the 3D case as described in [3], but with the 2D objects described in this article and using $\hat{\mathbf{g}}$ instead of $\mathbf{g}$. We now quickly describe how we derived this.

In the horizontal 2D case we made away with $\mathbf{g}$ as it was orthogonal to the plane where we projected. On a slope this is not true and the projection $\hat{\mathbf{g}}$ is nonzero. Our case can be thought of as a free-falling, non-rotating 2D frame (on our sloped plane) that follows an arbitrary $\hat{\mathbf{g}}$ 2D gravity vector contained in the plane (because its direction and magnitude depend on the slope's angles). Thus, the equations for this case are analogous to the 3D case, but use 2D objects and $\hat{\mathbf{g}}$ instead of $\mathbf{g}$. An illustration of this can be seen in Fig. 1

The $\boxminus$, $\boxplus$ operators are thus defined as:

$$\mathbf{\Delta}_{ij} = \mathbf{x}_j \boxminus \mathbf{x}_i := \begin{cases} \Delta\mathbf{p}_{ij} &= \mathbf{R}_i^\top\left(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\hat{\mathbf{g}}\Delta t_{ij}^2\right) \\ \Delta\mathbf{v}_{ij} &= \mathbf{R}_i^\top\left(\mathbf{v}_j - \mathbf{v}_i - \hat{\mathbf{g}}\Delta t_{ij}\right) \\ \Delta\mathbf{R}_{ij} &= \mathbf{R}_i^\top\mathbf{R}_j \end{cases} \tag{34}$$

$$\mathbf{x}_j = \mathbf{x}_i \boxplus \mathbf{\Delta}_{ij} := \begin{cases} \mathbf{p}_j &= \mathbf{p}_i + \mathbf{v}_i\Delta t_{ij} + \mathbf{R}_i\Delta\mathbf{p}_{ij} + \frac{1}{2}\hat{\mathbf{g}}\Delta t_{ij}^2 \\ \mathbf{v}_j &= \mathbf{v}_i + \mathbf{R}_i\Delta\mathbf{v}_{ij} + \hat{\mathbf{g}}\Delta t_{ij} \\ \mathbf{R}_j &= \mathbf{R}_i\Delta\mathbf{R}_{ij} \end{cases} \tag{35}$$

and the deltas form a Lie group with the same same composition, inverse and identity elements as our 2D case. The work done to find the tangent space, exponential, and Jacobians is identical here as the addition of the $\hat{\mathbf{g}}$ part to the delta has no effect on the procedure, only the dimensions of its parts, which are identical. The errors are calculated using the same equation described in (12).

V. Experiments

To validate the 2D preintegration methods presented we propose two different experiments:

- **Horizontal test:** This first experiment was performed inside our laboratory, where the floor can be assumed to be completely flat and orthogonal to gravity.
- **Sloped test:** The second experiment was carried out right outside of the building our institute is in, as there's a nice and steady slope there that was measured with an inclinometer as ranging from 3.1 to 3.8 degrees at different points.

To execute the experiments we used an skid-steer platform equipped with a 360 degrees field-of-view LIDAR sensor and a Microstrain IMU.

In each experiment, the trajectories are estimated in three different ways:

- As a 2D SLAM problem using the 2D IMU preintegration (Section III).
- As a 2D SLAM problem using the 2D IMU preintegration on a slope (Section IV).
- As a 3D SLAM problem using the state-of-art 3D IMU preintegration [2] implemented in WOLF [13].

In all three cases, one frame is added to the problem every 5 seconds. Along with the IMU preintegration factors, relative pose factors between consecutive frames are added using a Iterative Closest Point algorithm (ICP) [14]. The ICP measurements provide observability of the IMU biases.

To restrict the 3D case to the plane, we force the ICP-based relative pose factors to be horizontal transformations by imposing zero measurement and small covariance values on the displacement in Z, as well as on the rotations in roll and pitch.

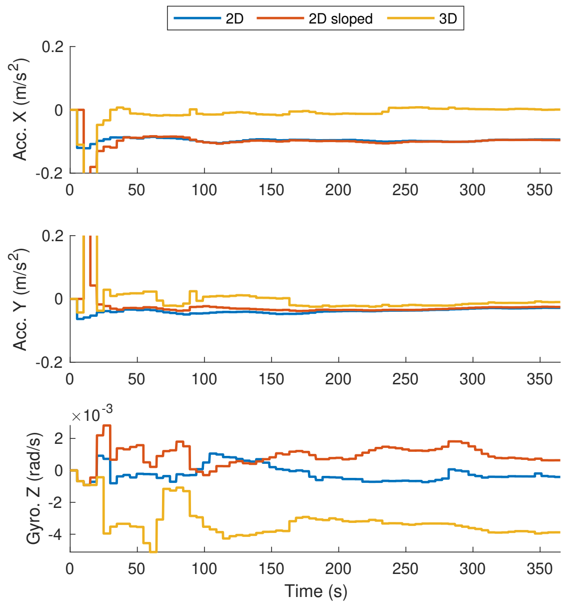

Fig. 4: Evolution of the estimation of the biases by the three different methods in the horizontal test.

A. Drift analysis

To validate the proposed formulation, we compare the drift of the state-of-art 3D preintegration against the two 2D preintegration methods proposed.

The ICP measurements give us a relative pose with respect to the last frame given by the ICP algorithm. Taking ICP result as baseline, we compare this pose with the one given by the corrected delta resulting from the IMU preintegration (11). This comparison gives us the IMU's drift. Trivially, when a new frame is added to the problem, the relative pose w.r.t. the last frame is zero, and so is the drift. For each experiment we will have several samples of the evolution of the drift along the 5 seconds between all consecutive frames.

For each value of time integration Δt, we compute the RMSE values of the translation and rotation drifts. Plotting these values with respect to Δt illustrates the average drift evolution of a preintegration method. The data of the first 5 frames is discarded to avoid errors due to unobserved bias.

1) Horizontal test: The average position and angular drifts of the three compared methods are depicted in Fig. 3. We observe that the 2D preintegration methods are equivalent to the 3D preintegration. In fact, the 2D methods slightly outperform the 3D method. This is expected, as firstly the 3D SLAM problem's dimension is twice the 2D problem's dimension. And secondly, in the 3D method the solver has to estimate the 2D nature of the solution while it is already imposed in the 2D method. The bias dynamics and the noise of the X and Y gyroscopes, as well as the noise of the Z accelerometer, cause the 3D preintegration to leave the plane.

Comparing the 2D preintegration with the 2D preintegration on a slope, we can observe that the results are practically identical, although the added necessity of estimating the slope causes the latter to take a little longer to correctly discriminate between the biases and gravity. This is shown in Fig. 4, where we can see the evolution of the biases estimation for the 3 methods. Only biases of the gyroscope Z and accelerometers X and Y are shown, since these are the only ones estimated during 2D preintegration.

2) Sloped test: The average evolution of position and angular drifts of the three compared methods are depicted in Fig. 3. We observe that in this case the horizontal 2D method is significantly worse than the other two, while the sloped 2D method outperforms the 3D method. When we use the horizontal method on a plane with a slope, there appears a *global* bias that corresponds to $\hat{\mathbf{g}}$ that can't be explained in the IMU's bias estimation, leading to higher errors. The same explanation as before applies again as to why the sloped 2D method outperforms the 3D method.

As we can see in Fig. 5 the 2D sloped biases converge faster than the rest and agree in value to the ones estimated by the 3D method, while they take longer for the horizontal 2D method.

B. Slope estimation

The slope of the environment and its orientation can be calculated from the resulting projected $\hat{\mathbf{g}}$ vector estimated by the 2D sloped preintegration method. The slope of the estimated trajectory produced in the 3D method can also be easily computed from the frames estimated orientation.

In Fig. 6 and Fig. 7 we can see the evolution of the estimation of the slope and its orientation for our two methods where orthogonal gravity is not assumed. The range of slope inclinations manually measured is depicted by the dotted line.

In the case of the experiment with a slope, both the 3D method and the 2D method on a slope manage to estimate a slope angle within the band measured by an inclinometer on multiple points. They also both converge towards a very

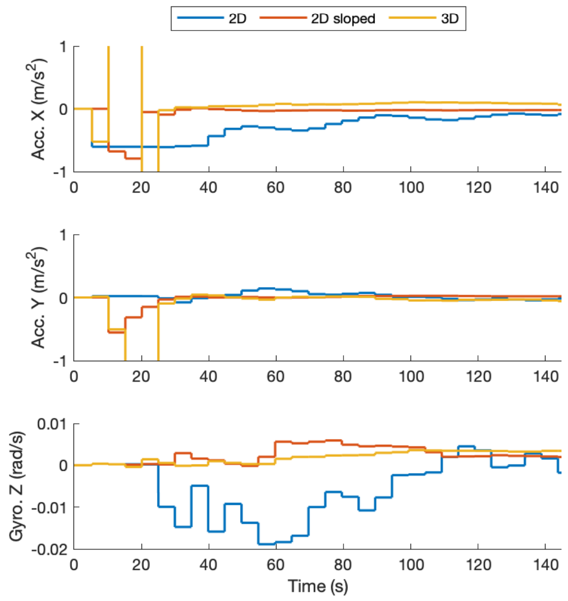

Fig. 5: Evolution of the estimation of the biases by the three different methods in the non-horizontal test.

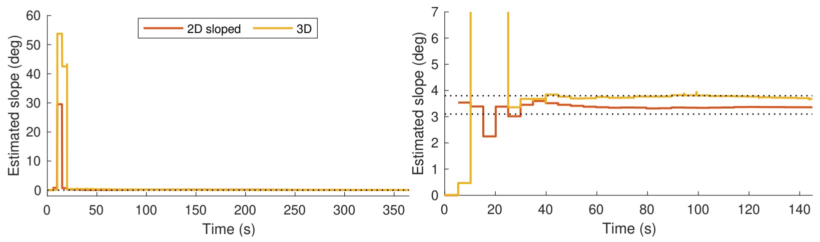

Fig. 6: Evolution of the estimation of the slope for the horizontal (left) and sloped (right) tests

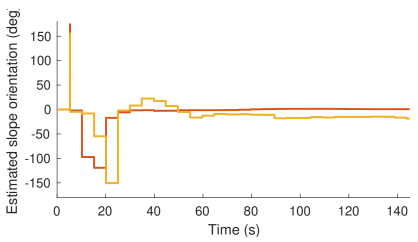

Fig. 7: Evolution of the estimation of the plane orientation in the sloped test

similar orientation of the slope, which is relative to the initial pose of the robot.

C. Computational cost

	Method	IMU capture process		Solver Time	
		Avg. (μs)	Max. (ms)	Avg. (ms)	Max. (ms)
Horiz.	2D	**28.9**	2.38	**0.800**	**14.9**
	2D sloped	31.1	**2.25**	0.875	18.4
	3D	34.2	4.23	1.79	67.4
Sloped	2D	**32.8**	**1.96**	**0.526**	32.6
	2D sloped	34.1	2.08	0.556	**21.8**
	3D	33.5	3.82	1.00	31.9

TABLE I: Computational cost of processing IMU measurements and solving the problem for all three compared alternatives on horizontal and sloped experiments.

One justification for considering the 2D problem instead of working directly with the 3D problem is that the reduced dimension impacts the computational cost favorably. We can observe that this is indeed the case in Tab. I. The average IMU capture process is practically the same in both experiments, but the peak slowest is significantly higher for the 3D method. When looking at the solver's iteration needed for convergence though, the picture is much clearer as the average iteration took double as long for both experiments, and the maximum iteration times paint a similar picture, as the 3D problem's dimension is double that of the 2D problem. Another thing to note is that, although the average number of iterations to reach convergence is 1 in all cases, the Horizontal 2D method boasted a 100% convergence rate. Similarly, the rest of the methods also had a near-100% convergence rate, with a negligible number of instances, of order $10^{-5}\%$, where the max number of solver iterations was reached without convergence.

VI. Conclusions

In this paper we applied the Lie formalism described in [9] to develop a preintegration pipeline for an IMU in a 2D problem, in the style of [4]. The resulting formulation was implemented in the WOLF [13] library, and experiments were carried out on a skid-steer platform to validate the method.

We found that both our 2D methods are equivalent to the 3D method when used on a plane orthogonal to gravity, and that our sloped 2D method outperforms the other two when applied to a sloped plane.

On the computational side, we observed that using the 2D methods on their appropriate case is more efficient than using the 3D method, thus justifying the use of 2D methods when we are solving a 2D problem.

Future work on the topic could include extending this work to multi-incline surfaces, as well as exploring the limits of the 2D method.

References

[1] T. Lupton and S. Sukkarieh, "Efficient integration of inertial observations into visual slam without initialization," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1547–1552.

[2] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[3] D. Atchuthan, "Towards new sensing capabilities for legged locomotion using real-time state estimation with low-cost IMUs," Theses, Université Paul Sabatier - Toulouse III, Oct. 2018. [Online]. Available: https://tel.archives-ouvertes.fr/tel-02088756

[4] M. Fourmy, D. Atchuthan, N. Mansard, J. Solà, and T. Flayols, "Absolute humanoid localization and mapping based on imu lie group and fiducial markers," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 237–243.

[5] P. Pinies, T. Lupton, S. Sukkarieh, and J. D. Tardos, "Inertial aiding of inverse depth slam using a monocular camera," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 2797–2802.

[6] K. Konolige, M. Agrawal, and J. Solà, "Large-scale visual odometry for rough terrain," in *Robotics Research*, M. Kaneko and Y. Nakamura, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 201–212.

[7] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Real-time monocular slam: Why filter?" in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 2657–2664.

[8] A. Barrau and S. Bonnabel, "A mathematical framework for imu error propagation with applications to preintegration," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 5732–5738.

[9] J. Solà, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," 2020.

[10] G. Cioffi, T. Cieslewski, and D. Scaramuzza, "Continuous-time vs. discrete-time vision-based slam: A comparative study," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2399–2406, 2022.

[11] J. Borenstein and L. Feng, "Gyrodometry: a new method for combining data from gyros and odometry in mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, 1996, pp. 423–428 vol.1.

[12] M. Fourmy, T. Flayols, P.-A. Léziart, N. Mansard, and J. Solà, "Contact forces preintegration for estimation in legged robotics using factor graphs," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1372–1378.

[13] J. Solà, J. Vallvé-Navarro, J. Casals, J. Deray, M. Fourmy, D. Atchuthan, and J. Andrade-Cetto, "Wolf: a modular estimation framework for robotics based on factor graphs," 2021, article sense publicar.

[14] A. Censi, "An icp variant using a point-to-line metric," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 19–25.