On-manifold Probabilistic ICP : Application to Underwater Karst Exploration

Yohan Breux, André Mas, Lionel Lapierre

Abstract

This paper proposes an on-manifold derivation of the probabilistic ICP (pICP) and its adaptation in the context of autonomous underwater karst exploration. As vision-based methods may fail due to water turbidity, we have to rely on acoustic sonar measurements. This work leverages previous results on elevation angle estimations of wide-beam profiling sonar and adapts the pICP to the obtained point distributions.

On-manifold Probabilistic ICP : Application to Underwater Karst Exploration.

Yohan Breux, André Mas and Lionel Lapierre

March 24, 2021

1 Introduction

Iterative Closest Point (ICP) [3] is a well-known technique widely used in domains such as 3D Data registration or SLAM (Simultaneous Localization And Mapping). It aims at computing the relative displacement between two given 2D or 3D point clouds acquired with some sensors (3D scanner, LIDAR, acoustic sonar). The basic algorithm consists in iterating over the two following steps until convergence or for a fixed number of iterations. First, compute the correspondences (associations) between the two input clouds. Then find the displacement minimizing the sum of euclidean distance error over the associated pair of points. Generally, the minimization is done with least square methods. As only a local optimum can be found, this algorithm is limited to relatively small displacement or requires a good initial estimate for larger displacement.

In this paper, we are interested in its probabilistic extension first proposed by [18] that we note pIC. It takes in account uncertainties in the points positions and in the initial displacement estimate by modeling them as Gaussian random variables. This is particularly important when using sensors with strong noise such as underwater acoustic sonar. Furthermore, unlike LIDAR which provides full scans almost instantaneously, mechanically scanning sonar systems (MSIS) can take several seconds for a full 360° scan due to higher time-of-flight underwater. Besides, as the robot is moving during the acquisition, its uncertainty is also propagated to the measurements.

In the context of underwater karst exploration, we have to rely on acoustic sonar as water turbidity affects the efficiency of vision-based and LIDAR approaches. pIC is then the natural candidate in a SLAM framework to estimate the displacement between two overlapping scan. To guarantee overlapping between two successive scans, we have to rely on wide-beam sonar. The drawback of such sonar is that the elevation angle is unknown, leading to high uncertainty to the measured 3D points. Previous systems based on MSIS [15][16] deals with this problem by considering a 2D SLAM. In our previous work [5], we propose a method to estimate the probability distribution of elevation angles. We first compute a probabilistic surface model from the measurements of a vertically mounted narrow-beam sonar which is then leveraged to estimate the elevation angles from an horizontally mounted wide-beam sonar. We thus obtain 3D measurements following non-Gaussian distributions.

SLAM algorithms' localization accuracy is directly related to the ICP/pIC performance. Our first contribution in this paper is to propose the derivation of on-manifold probabilistic ICP (noted MpIC) as well as the derivation of the estimated transformation covariance. We show that it leads to superior performance compared to estimation based on Euler angles or quaternion representations. To the best of our knowledge, the literature [18][15][7][19] exploiting the pIC algorithms implicitly suppose an independent error covariance to obtain their closed-form solution for the transformation estimation. In this work, we also take into account the dependence of the error covariance on the transformation to estimate.

We then apply it in the context of underwater karst exploration based on an extension of our previous work [5]. In particular, we propose a Gaussian approximation to our estimated sonar measurements. In order to use a point-to-plane association scheme, we need to compute normals at each point. As we can expect sparse point cloud, we propose to compute the normals and their covariances based on the environment probabilistic model as obtained in [5]. In order to quantitatively assess our algorithm, we need to have access to ground truth displacement which are difficult to obtain in natural karst environment. We evaluate our algorithm on a simulated karst environment and compare it to state-of-the-art 2D approach successfully used in real underwater cave [16][17].

The structure of the paper is as follows. Section 2 gives an overview on ICP approaches with a focus on underwater robotics application. Section 3 briefly introduces generalities on Lie groups and related notations used throughout this paper. The description of pIC and its on-manifold derivation is given in Section 4. Section 5 succinctly introduces our previous elevation angle estimation method and its extensions with notably normal computations and Gaussian approximation. Finally, Section 6 evaluates our approach through experiments on a simulated karst environment. Conclusion and future working directions are left in Section 7.

2 Related Works

The original ICP algorithm comes from the seminary works by [3] and [10] for point-to-plane associations. A lot of improvements have been proposed by exploiting additional features such as surface orientations [9][23] or color [11][24], improving the association step [18][1] and/or the objective function [18][21][12]. The range of ICP applications is large and we are interested here on its usage in mobile robotics. A comprehensive overview can be found in [20].

In order to take into account sensor uncertainties, Montesano et al. [18] propose pIC, a probabilistic approach of the original ICP algorithm. They consider both points and the relative transformation between the point clouds as Gaussian random variables. The Euclidean distance used in the ICP association step and in the objective function is replaced by the Mahalanobis distance. Doing so, each association error is weighted by its covariance so that uncertain data contribute less to the objective function.

Segal et al. [23] also introduce a probabilistic approach with their generalized ICP (G-ICP). The association step is done as in the original ICP using Euclidean distances. However, they consider points as Gaussian random variables with fixed uncertainties in the plane orthogonal to the surface at the considered point. While this does not directly modelized uncertainties related to the measurement, it takes account of the sampling error of the surface e.g. corresponding points are not exactly the same point on the surface. Note that unlike pIC, the transformation is not considered as a random variable.

Agamennoni et al. [1] recently propose another probabilistic variant without explicit association phase. More precisely, they search to maximize the likehood of points from the target cloud conditioned on the associated point in the reference cloud. Each possible association is weighted by a latent variable. They use an Expectation-Maximization (EM) procedure to iteratively estimate the transformation and the associations weights.

In the field of underwater exploration, localization with SLAM approaches are often based on acoustic sonar scan matching. The level of noise of such sensors is higher than scan laser used in terrestrial robotics. Furthermore, due to the propagation time of wave in water, a full scan can take several seconds to complete while the robot is moving. We thus have to consider uncertainties in the robot position during the scan acquisition on top of measurement errors.

Palomer et al [19] develop an EKF SLAM framework for seafloor mapping. They use a multibeam echosounder and the resulting scans are matched using pIC in 3D. However, the closed-form expression for the estimated transformation in the pIC error minimization step is obtained by implicitly considering an error covariance independent of the transformation. In fact, this covariance is dependent on the transformation through the jacobians involved in its definition. Furthermore, optimization is made using Euler angle representation which can be prone to gimbal lock [4].

In this paper, we are interested in methods exploiting mechanically scanned imaging sonar (MSIS). Mallios et al. [15][17] use the pIC algorithm [18] in an Extended Kalman Filter (EKF) SLAM framework for underwater cave exploration in a 2D setting. A similar approach is proposed by Burguera et al. [7] [6]. Similarly to this work, they are based on mechanically scanned imaging sonar (MSIS).

In order to estimate 3D transformations, we propose in our previous work [5] a method for estimating the distribution of elevation angles from wide-beam sonar measurements by leveraging measurements from a secondary narrow-beam sonar. In this paper, we propose a generic on-manifold pIC algorithm MpIC with the derivation of its estimated transformation covariance. We then propose a practical application where it is used to match 3D scan obtained by MSIS acoustic sonars where elevation angles are estimated using the method in [5].

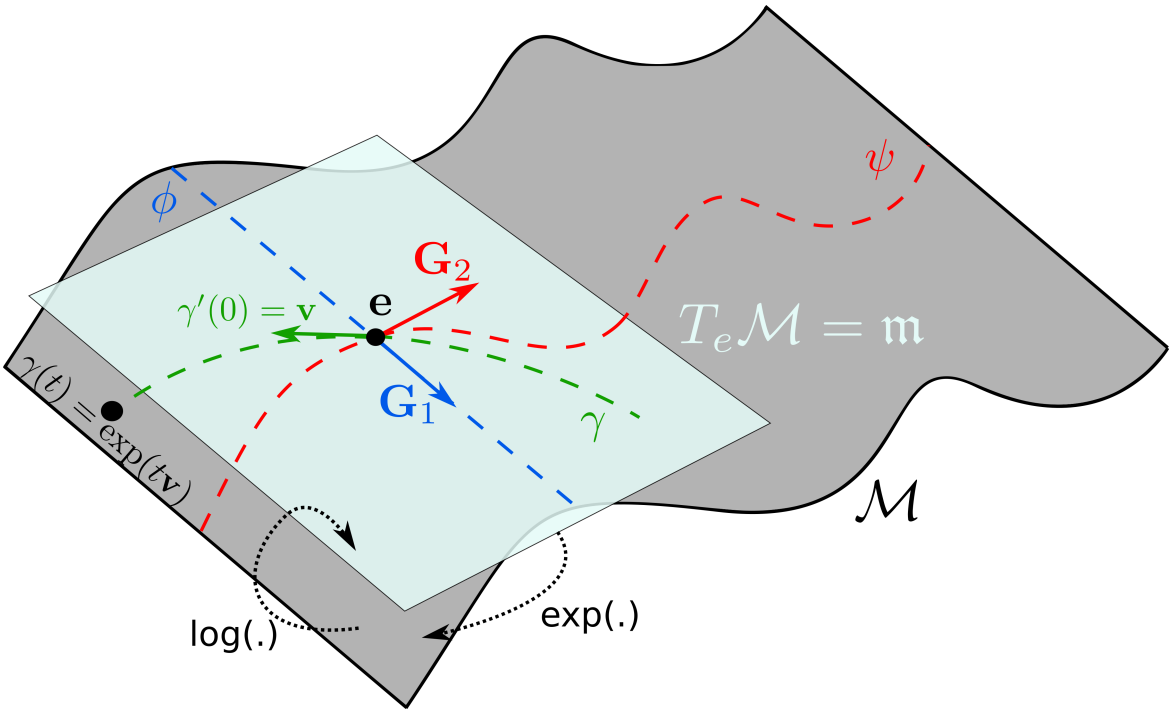


Figure 1: Representation of a 2-manifold $\mathcal{M}$ in $\mathbb{R}^3$ with its tangent space $T_e\mathcal{M}$ at a point $\mathbf{e} \in \mathcal{M}$. ϕ and ψ are two curves of $\mathcal{M}$ passing through e such that their tangent vector G_1, G_2 at $\mathbf{e}$ generate a basis of $T_e\mathcal{M}$. If $\mathcal{M}$ is a Lie group and $\mathbf{e}$ its neutral element, then $T_e\mathcal{M} = \mathfrak{m}$ is its associated Lie algebra. The transition from one to another is done through the applications $exp : \mathfrak{m} \mapsto \mathcal{M}$ and $log : \mathcal{M} \mapsto \mathfrak{m}$. γ is an example of path with velocity $\mathbf{v}$.

3 Generalities on Lie groups

In this section, we introduce the notations and some differential geometry concepts relevant to the derivations in the following sections. In particular, we focus on Lie Groups and more particularly on the 3D rigid transformation group SE(3). A good introduction to Lie groups for application in Robotics can be found in [25]. A more detailed coverage of differential geometry and Lie groups can be found in [14]

A N-dimensional manifold $\mathcal{M}$ is a topological space where every neighboorhood V_p of a point $p \in \mathcal{M}$ is homeomorphic to $\mathbb{R}^N$.

A N-dimensional manifold embedded in $\mathbb{R}^{\mathbb{D}}$, $N < D$, is said to be smooth (or differentiable) if every point $p \in \mathcal{M}$ can be locally parametrized by a C^∞-diffeomorphism $\phi : \Omega \to U$ with $p \in U$, $\mathbf{O}_N \in \Omega$ and (Ω, U) being respectively open subsets of $\mathbb{R}^{\mathbb{N}}$ and $\mathcal{M}$. The tangent plane at a point p on $\mathcal{M}$ is designed by $T_p\mathcal{M}$.

A Lie group $\mathcal{G}$ is defined as a smooth manifold with a group structure such that its group product and its inverse are differentiable. We can then define its Lie algebra $\mathfrak{g} = T_e\mathcal{G}$ (a vector space equipped with a bilinear product called here the Lie bracket) which corresponds to the tangent space of G at its identity element e. Generally, computations are easier on the Lie Algebra. In particular, it is used for on-manifold non-linear optimizations such as Gauss-Newton or Levenberg-Marquardt as they are designed to work on vector spaces and not on general manifolds.

In the following, we only consider matrix Lie groups. By noting $GL(n)$ the general linear group in dimension n (set of inversible real $n \times n$ inversible matrices), the Cartan theorem shows that every closed subgroup of $GL(n)$ is a Lie group.

The exponential map $exp : \mathfrak{g} \to \mathcal{G}$ maps elements of the algebra to the group. It is defined, in the case of matrix Lie group, as the exponential of a matrix such that

$$e^A = \sum_{k=0}^{+\infty} \frac{1}{k!} A^k$$

We note its inverse the logarithm map $log : G \to \mathfrak{g}$. Note that there is no general results on the injectivity/surjectivity of those applications.

We define the isomorphism $^\wedge$ which maps elements of $\mathbb{R}^N$ (local parametrization) to elements of the Lie algebra. We also define its inverse operator vee $^\vee$.

$$\begin{aligned} ^\wedge &: \mathbb{R}^N \mapsto \mathfrak{g} \\ \mathbf{g} &\to (\mathbf{g})^\wedge \equiv \widehat{\mathbf{g}} = \sum_{i=1}^{N} g_i G_i \end{aligned}$$

where $\{G_i\}$ is a basis (called generators) of the Lie algebra.

Smooth manifolds, and then Lie groups, are also Riemaniann manifolds ([14], Theorem 13.2 p410). Thus, we can define a smooth inner product $< \cdot, \cdot >_p$ on the tangent space $T_p\mathcal{M}$ at each point $p \in \mathcal{M}$. The metric induced by this inner product is used to measure the length of piecewise-C^1 curves $\gamma : [a, b] \subset \mathbb{R} \mapsto \mathcal{M}$ on $\mathcal{M}$ such that

$$L(\gamma) = \int_a^b \langle \gamma'(t), \gamma'(t) \rangle_{\gamma(t)}^{\frac{1}{2}} dt \tag{1}$$

with the velocity $\gamma'(t) \in T_{\gamma(t)}\mathcal{M}, \forall t \in [a, b]$. Geodesics are an important class of curves. They are the curves which locally minimize the length between two points and are equivalent to the straight line in Euclidean geometry (null curvature). In fact, by definition, geodesics are curves with constant velocity. This means that $\forall t, \gamma'(0) = \gamma'(t) = v \in T_{\gamma(0)}\mathcal{M}$. We can then define for a given velocity v a geodesic γ going through a point $p \in \mathcal{M}$

$$\gamma(t) = e^{tv}$$

It is illustrated in Figure **??**.

In this paper, we are interested in two particular matrix Lie groups : the Special Orthogonal group $SO(3)$ representing the rotations and the Special Euclidian group $SE(3)$ representing the rigid transformations in R^3.

3.1 SO(3) and SE(3)

$SO(3)$ is a closed subgroup of $GL(3)$ defined as

$$SO(3) = \left\{ R \in GL(3) |\ RR^T = I_3 \text{ and } \det R = 1 \right\}$$

The corresponding Lie algebra $\mathfrak{so}(3)$ is defined by the space of skew-symmetric matrices

$$\forall W \in \mathfrak{so}(3), W = \widehat{\boldsymbol{\omega}} = [\boldsymbol{\omega}]_\times\ ,\ \boldsymbol{\omega} \in \mathbb{R}^3$$

The generators $G_i^{\mathfrak{so}(3)}$ of $\mathfrak{so}(3)$ are given by

$$G_i^{\mathfrak{so}(3)} = \widehat{e_i} \tag{2}$$

where $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ is the canonical basis of $\mathbb{R}^3$.

$SE(3)$ is defined as the semidirect product $SO(3) \ltimes \mathbb{R}^3$ and is isomorphic to the affine group $GA(3)$ (a subgroup of $GL(4)$) through the application

$$\begin{aligned} SO(3) \times \mathbb{R}^3 &\mapsto GA(3) \\ R, \mathbf{t} &\to \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \end{aligned}$$

Its Lie algebra $\mathfrak{se}(3)$ is given by

$$\begin{aligned} &\forall \Xi \in \mathfrak{se}(3),\ \Xi = \widehat{\boldsymbol{\xi}} = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{\tau} \\ \mathbf{0}^T & 0 \end{bmatrix} \\ &\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\omega}^T & \boldsymbol{\tau}^T \end{bmatrix}^T, \boldsymbol{\omega} \in \mathbb{R}^3,\ \boldsymbol{\tau} \in \mathbb{R}^3 \end{aligned}$$

The generators $G_i^{\mathfrak{se}(3)}$ of $\mathfrak{se}(3)$ are given by

$$G_i^{\mathfrak{se}(3)} = \begin{bmatrix} G_i^{\mathfrak{so}(3)} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix},\ i \in \{1,2,3\}$$
$$G_i^{\mathfrak{se}(3)} = \begin{bmatrix} 0 & \mathbf{e}_i \\ \mathbf{0}^T & 0 \end{bmatrix},\ i \in \{4,5,6\} \tag{3}$$

Expressions for the exponential / log maps of $SO(3)$ and $SE(3)$ can be found in [25].

Throughout this paper, we use the left-invariant Riemannian metric induced by the point-independent Riemanian metric on $GA(3)$ ([2], Section 3.2)

$$\begin{aligned} \langle u, v \rangle_A &= Tr(\mathbf{u}^T \mathbf{v}) \\ &= \langle A^{-1}\mathbf{u}, A^{-1}\mathbf{v} \rangle_e \\ &= \langle \mathbf{s}_u, \mathbf{s}_v \rangle_e \\ &= \mathbf{s}_u^T G \mathbf{s}_v \end{aligned} \tag{4}$$

$$G = \begin{bmatrix} I_3 & 0 \\ 0 & 2I_3 \end{bmatrix}, A \in SE(3),\ \mathbf{u}, \mathbf{v} \in T_A SE(3)$$

where e is the identity element of $SE(3)$, $T_A SE(3)$ the tangent plane at A and $\mathbf{s}_u, \mathbf{s}_v \in \mathfrak{se}(3)$.

Pose-pose and pose-point compositions (resp. inverse compositions) are represented by the operator $\oplus$ (resp. $\ominus$). More precisely, for 3D pose $q \in SE(3)$ with $SE(3)$ seen as a matrix group, the pose-pose composition corresponds to the group multiplication and the pose-point composition to the group action on $\mathbb{R}^3$.

We also define the $\boxplus$ operator for directly composed increments of Lie group expressed in the tangent vector space (local parametrization). It is given by

$$\begin{aligned} \boxplus : \mathcal{G} \times \mathbb{R}^n &\mapsto \mathcal{G} \\ G,\ \boldsymbol{\xi} &\to G \boxplus \boldsymbol{\xi} = G e^{\widehat{\boldsymbol{\xi}}} \end{aligned}$$

3.2 Jacobian on a Riemannian Manifold

For any smooth function $f : \mathcal{M} \to \mathbb{R}^m$ with $\mathcal{M}$ being a Riemannian manifold, the differential of the function f at $p \in \mathcal{M}$ is defined by (Definition 13.3, [14])

$$\forall u \in T_p\mathcal{M}, Df_p(u) = \frac{d}{dt} f(\gamma(t))_{|t=0} \tag{5}$$

where γ is a geodesic of $\mathcal{M}$ going through p ($\gamma(0) = p$) and with velocity u ($\gamma'(0) = u$). If $\mathcal{M}$ is a Lie group and $\mathfrak{m}$ its Lie algebra, it can be shown ([14], Proposition 20.20) that the one-parameter subgroups $exp(tX_0),\ X_0 \in \mathfrak{m}$ are its geodesics.

In the case of a matrix Lie group $\mathcal{G}$, the geodesics are thus

$$\gamma(t) = T e^{t\widehat{g}}, T \in \mathcal{G}, \widehat{g} \in \mathfrak{g} \tag{6}$$

Put simply, $Df_p(u)$ is the directional derivative along the direction vector u. From this, in the case of $SO(3)$ and $SE(3)$ manifolds, we can compute the Jacobian matrix of f $J_{|p}^f$ by computing its derivatives along the generators G_i of $\mathfrak{so}(3)$ and $\mathfrak{se}(3)$. Formally

$$J_{|p}^f = [Df_p(PG_j)] \tag{7}$$

Note that we have the following relation

$$J_{|p}^f g = Df_p(P\widehat{g}),\ g \in \mathbb{R}^n \tag{8}$$

The advantage of using this jacobian definition is to only operate univariate derivatives which keeps calculus simple.

In the literature, jacobian computation on Lie groups are often done directly using matrix expression leading to complex tensorial calculus involving matrix vectorization and kronecker products [][].

While this approach gives exact matricial equalities, it is unnecessary complex for implementation purpose. The definition proposed here gives simple expression for each element of the Jacobian which are easier to implement and computationally efficient.

The differentials of $SE(3)$ group action and its inverse are given by

$$q \equiv T = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{9}$$

$$D \oplus_{|\mathbf{q},\mathbf{a}} (T\widehat{\boldsymbol{\xi}}) = T\widehat{\boldsymbol{\xi}}\mathbf{a} \tag{10}$$

$$D \ominus_{|\mathbf{q},\mathbf{a}} (T\widehat{\boldsymbol{\xi}}) = -\widehat{\boldsymbol{\xi}}T^{-1}\mathbf{a} \tag{11}$$

The jacobians of $SE(3)$ group action and its inverse are then given by

$$J^{\oplus}_{\mathbf{q}_{|\mathbf{q},\mathbf{a}}} \equiv \frac{\partial \mathbf{q} \oplus \mathbf{a}}{\partial \mathbf{q}} = R \begin{bmatrix} -[\mathbf{a}]_\times & I_3 \end{bmatrix} = RU_{\mathbf{a}} \tag{12}$$

$$J^{\oplus}_{\mathbf{a}_{|\mathbf{q},\mathbf{a}}} \equiv \frac{\partial \mathbf{q} \oplus \mathbf{a}}{\partial \mathbf{a}} = R \tag{13}$$

$$J^{\ominus}_{\mathbf{q}_{|\mathbf{q},\mathbf{a}}} \equiv \frac{\partial \mathbf{a} \ominus \mathbf{q}}{\partial \mathbf{q}} = \begin{bmatrix} [R^T\mathbf{a} - R^T t]_\times & -I_3 \end{bmatrix} = -R^T U_{\mathbf{a}-\mathbf{t}} (\mathbf{1}_2 \otimes R) \tag{14}$$

$$J^{\ominus}_{\mathbf{a}_{|\mathbf{q},\mathbf{a}}} \equiv \frac{\partial \mathbf{a} \ominus \mathbf{q}}{\partial \mathbf{a}} = R^T \tag{15}$$

3.3 Hessian on a Riemannian Manifold

For a smooth function $f : \mathbb{R}^n \mapsto \mathbb{R}$, the Hessian at p of a smooth function f is defined by

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n,\ Hess(f)_p(\mathbf{x}, \mathbf{y}) = \mathbf{x}^t H_p \mathbf{y} \tag{16}$$

where $H_p = \left[\frac{\partial^2 f_i}{\partial x_i \partial x_j}\right]$ is the usual hessian matrix. Similarly, for a smooth function f on a C^{k+1} Riemannian manifold $\mathcal{M}$, its hessian at $p \in \mathcal{M}$ is noted $Hess(f)_p(X, Y)$ where X, Y are two C^k vector fields on $\mathcal{M}$.[1] In particular, for $u \in T_p\mathcal{M}$, we have ([14], Proposition 15.22)

$$Hess(f)_p(u, u) = \frac{d^2}{dt^2} f(\gamma(t))_{|t=0} \tag{17}$$

where γ is a geodesic of $\mathcal{M}$ going through p ($\gamma(0) = p$) and with velocity u ($\gamma'(0) = u$).

Similarly to the differential, by noting that $T_T\mathcal{G} = T\mathfrak{g}$, the hessian matrix is deduced from the hessian operator and the vector fields $X : T \to TG_i \in T\mathfrak{g}$ such that

$$H^f_T = [Hess(f)_T (TG_i, TG_j)] \tag{18}$$

with G_i are the generators of $\mathfrak{g}$ as define respectively in eq (2) and (3) for $\mathfrak{so}(3)$ and $\mathfrak{se}(3)$.

The diagonal terms are computed using eq (17). Other terms are obtained by exploiting the fact that the hessian is a symmetric bilinear form :

$$\begin{aligned} &Hess(f)_T (TG_i, TG_j) = \\ &\frac{1}{2} (Hess(f)_T (TG_i + TG_j, TG_i + TG_j) \\ &- Hess(f)_T (TG_i, TG_i) - Hess(f)_T (TG_j, TG_j)) \end{aligned} \tag{19}$$

4 Probabilistic Iterative Closest Point (ICP)

The Iterative Closest Point (ICP) algorithm is an iterative non-linear 2-step process for registering unstructured 2D or 3D point clouds. In other words, it aims at estimating the displacement (rigid transformation) between two point clouds. Given an initial displacement, each iteration consists in associating points in the first cloud to points in the displaced second cloud.

[1] A C^k-vector field X is a mapping $p \in \mathcal{M} \to X(p) \in T_p\mathcal{M}$ such that X(p) is C^k with reference to p, see [14] Section 9.2

Once done, the algorithm computes with non-linear optimization the displacement minimizing the sum of errors between each pair of associated points. Originally, the error is measured as the L2 norm between the points but variants on the error function have been proposed.

A probabilistic variant taking in account point uncertainties is proposed by [18] and further exploited in the context similar to our work in 2D [15] and 3D [19].

In the previous works, the optimization is based on Euler angles representation and the error covariance is implicitly assumed independent of the estimated transformation. We improve upon previous iterations with two main contributions : we directly optimize on the SE(3) manifold using derivative on its Lie algebra and we also take into account the error covariance dependence on the estimated displacement.

The inputs of the algorithm are the reference point cloud $S_{ref} = \{\mathbf{r}_i\}_{i=1...n}$, the newly acquired point cloud $S_{new} = \{\mathbf{c}_i\}_{i=1...m}$ and an initial transformation $q^{(0)}$ such that

$$\begin{aligned} \mathbf{r}_i &\sim \mathcal{N}\left(\bar{\boldsymbol{r}}_i \in \mathbb{R}^3, \Sigma_{r_i}\right),\ \mathbf{c}_i \sim \mathcal{N}\left(\bar{\boldsymbol{c}}_i, \Sigma_{c_i}\right) && (20) \\ \mathbf{q}^{(0)} &\sim \mathcal{N}\left(\bar{\boldsymbol{q}}^{(0)} \in SE(3), \Sigma_{\mathbf{q}^{(0)}}\right) && (21) \end{aligned}$$

Note that we do not need to specify how we represent the transformation q. Generally, it is represented as 7D vector composed of a quaternion for the rotation part and a 3D translation. It can also be represented as a 6D vector using Euler angles for the rotation part.

We now consider the k-th iteration. The current displaced point cloud is $S_{new}^{(k)} = \left\{\mathbf{n}_i^{(k)}\right\}_{i=1...m}$ with $\mathbf{n}^{(k)} \sim \mathcal{N}\left(\bar{\boldsymbol{n}}_i^{(k)}, \Sigma_{\mathbf{n}_i^{(k)}}\right)$ and

$$\begin{aligned} \bar{\boldsymbol{n}}_i^{(k)} &= \bar{\boldsymbol{q}}^{(k-1)} \oplus \bar{\boldsymbol{c}}_i && (22) \\ \Sigma_{\mathbf{n}_i^{(k)}} &= J^{\oplus}_{\mathbf{q}_{|\bar{\boldsymbol{q}}^{(k-1)}, \bar{\boldsymbol{n}}_i}} \Sigma_{q^{(0)}} J^{\oplus^T}_{\mathbf{q}_{|\bar{\boldsymbol{q}}^{(k-1)}, \bar{\boldsymbol{n}}_i}} + J^{\oplus}_{\mathbf{a}_{|\bar{\boldsymbol{q}}^{(k-1)}, \bar{\boldsymbol{n}}_i}} \Sigma_{\mathbf{c}_i} J^{\oplus^T}_{\mathbf{a}_{|\bar{\boldsymbol{q}}^{(k-1)}, \bar{\boldsymbol{n}}_i}} && (23) \end{aligned}$$

where the involved jacobians $J^{\oplus}_{\mathbf{q}}$ and $J^{\oplus}_{\mathbf{a}}$ are defined in eq (12) and eq (13).

4.1 Associations

For the association step, we search for each point $\mathbf{n}_i^{(k+1)} \in S_{new}^{(k+1)}$ the set of candidates $A_i^{(k+1)} \subset S_{ref}$. In the following, the errors $\mathbf{e}_{ij} \sim \mathcal{N}(\bar{\boldsymbol{e}}_{ij}, \Sigma_{e_{ij}})$ between two points $\mathbf{n}_i^{(k+1)}$ and $\mathbf{r}_j$ are defined by

$$\begin{aligned} \bar{\boldsymbol{e}}_{ij} &= \bar{\boldsymbol{n}}_i - \bar{\boldsymbol{r}}_j = \left(\bar{\boldsymbol{q}}^{(k+1)} \oplus \bar{\boldsymbol{c}}_j\right) - \bar{\boldsymbol{r}}_j && (24) \\ \Sigma_{e_{ij}} &= \Sigma_{\mathbf{n}_i} + \Sigma_{\mathbf{r}_j} && (25) \end{aligned}$$

The Mahalanobis distance $D^2(\mathbf{n}_i, \mathbf{r}_j)$ of a pair of points is given by

$$D^2(\mathbf{n}_i, \mathbf{r}_j) = \bar{\boldsymbol{e}}_{ij}^T \Sigma_{e_{ij}}^{-1} \bar{\boldsymbol{e}}_{ij} \quad (26)$$

As the Mahalanobis distance follows a chi-squared distribution, we can use a confidence level $\alpha \in [0, 1]$ and consider a valid candidate for association if

$$D^2(\mathbf{n}_i, \mathbf{r}_j) < \chi^2_{3,\alpha} \quad (27)$$

In the case of *point-to-point* association, the final association $\mathbf{a}_i \in A_i^{(k+1)}$ for $\mathbf{n}_i$ is simply the element in $A_i^{(k+1)}$ which minimizes $D^2(\mathbf{n}_i, \cdot)$

$$\mathbf{a}_i = \arg \min_{\mathbf{r} \in A_i^{(k+1)}} D^2(\mathbf{n}_i, \mathbf{r}) \quad (28)$$

The drawback of the point-to-point association is to suppose that the exact same points belong to both point clouds. In practice, the two inputs point cloud results from two different sampling of the environment leading to misalignment.

Complementary to the point-to-point association, the *point-to-plane* association uses local information on the surface shape and performs better against misalignments. The general approach is then

Algorithm 1 PICP Association

Input: Association scheme T_{assoc}, Transformation $\mathbf{q}$, Gaussian point $\mathbf{n} \in \mathbf{q} \oplus S_{new}$, Candidate set $A \subset S_{ref}$

Output: Gaussian point $\mathbf{a} \in S_{ref}$ associated with $\mathbf{n}$

if (T_{assoc} = point-to-point) **then**
  $\mathbf{a} = \arg\min_{\mathbf{r} \in A} D^2(\mathbf{n}, r)$
else if (T_{assoc} = point-to-plane) **then**
  $\alpha = \sum_{\bar{\boldsymbol{r}} \in A} \mathrm{Tr}(\Sigma_{\mathbf{r}})^{-2}$
  $\mathbf{p}_\mu = \left(\sum_{\mathbf{r} \in A} \mathrm{Tr}(\Sigma_{\mathbf{r}})^{-2} \bar{\boldsymbol{r}}\right) \alpha^{-1}$
  $C = \sum_{\mathbf{r} \in A} \frac{(\bar{\boldsymbol{r}} - \mathbf{p}_\mu)(\bar{\boldsymbol{r}} - \mathbf{p}_\mu)^T}{Tr(\Sigma_{\mathbf{r}}^2)}$
  $K = \sum_{\mathbf{r} \in A} \frac{\bar{\boldsymbol{r}}(\bar{\boldsymbol{r}} - \mathbf{p}_\mu)^T}{Tr(\Sigma_{\mathbf{r}}^2)}$
  $H = \begin{bmatrix} K - \bar{\boldsymbol{v}}^T K^T \bar{\boldsymbol{v}} & \mathbf{p}_\mu \\ \mathbf{p}_\mu^T & \alpha \end{bmatrix}$
  $\bar{\boldsymbol{v}}$ = Eigen vector of C associated with minimal eigen value
  $\bar{d} = \bar{\boldsymbol{v}}^T \mathbf{p}_\mu$
  $\mathbf{p} \sim \mathcal{N}\left([\bar{\boldsymbol{v}}\ \bar{d}]^T, H^+\right)$
  $\bar{\boldsymbol{a}} = \bar{\boldsymbol{q}} \oplus \bar{\boldsymbol{n}} - \left((\bar{\boldsymbol{q}} \oplus \bar{\boldsymbol{n}})^T \bar{\boldsymbol{v}} - \bar{\boldsymbol{d}}\right) \bar{\boldsymbol{v}}$
  $\Sigma_a = J^{\mathbf{a}}_{\mathbf{q}} \Sigma_{\mathbf{q}} J^{\mathbf{a}^T}_{\mathbf{q}} + J^{\mathbf{a}}_{\mathbf{n}} \Sigma_{\mathbf{n}} J^{\mathbf{a}^T}_{\mathbf{n}} + J^{\mathbf{a}}_{\mathbf{p}} \Sigma_{\mathbf{p}} J^{\mathbf{a}^T}_{\mathbf{p}}$
  $\mathbf{a} \sim \mathcal{N}(\bar{\boldsymbol{a}}, \Sigma_a)$
end if

to first use point-to-point association for the first iterations and to switch later on to point-to-plane association.

First, we need to estimate the tangent planes $\mathbf{p}_i = \begin{bmatrix} \mathbf{v}_i^T & \mathbf{d}_i^T \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \bar{\boldsymbol{v}}_i^T & \bar{\boldsymbol{d}}_i^T \end{bmatrix}, \Sigma_{\mathbf{p}_i})$ defined by $\mathbf{v}_i^T \mathbf{x} = \mathbf{d}_i$ at the surface from which each points has been measured. If the point clouds are dense enough, the classic approach is to fit a plane on the candidate sets $A_i^{(k+1)}$. In [19] an efficient method to find the normals and their associated covariance is proposed. It is summarized in Algorithm 1. In our work, we consider rather sparse clouds so that this method is not practicable. However, in Section **??**, we propose a novel method to estimate the normals based on a Gaussian process regression of the environment surface.

Once the normals are computed, the final association $\mathbf{a}^\perp{}_i \sim \mathcal{N}(\bar{\boldsymbol{a}}_i^\perp, \Sigma_{\mathbf{a}_i^\perp})$ for $\mathbf{n}_i$ is the orthogonal projection of $\mathbf{n}_i$ on the plane $\mathbf{p}_i$ and is given by

$$\bar{\boldsymbol{a}}_i^\perp = \bar{\boldsymbol{n}}_i - \left(\bar{\boldsymbol{n}}_i^T \bar{\boldsymbol{v}}_i - \bar{\boldsymbol{d}}_i\right) \bar{\boldsymbol{v}}_i \tag{29}$$

$$\Sigma_{\mathbf{a}_i} = J^{\mathbf{a}_i^\perp}_{\mathbf{n}_i} \Sigma_{\mathbf{n}_i} J^{\mathbf{a}_i^{\perp T}}_{\mathbf{n}_i} + J^{\mathbf{a}_i^\perp}_{\mathbf{p}_i} \Sigma_{\mathbf{p}_i} J^{\mathbf{a}_i^{\perp T}}_{\mathbf{p}_i} \tag{30}$$

In both associations schemes, the final errors $\mathbf{e}_i$ are defined by

$$\bar{\boldsymbol{e}}_i = \bar{\boldsymbol{n}}_i - \bar{\boldsymbol{a}}_i \tag{31}$$

$$\Sigma_{\mathbf{e}_i} = \Sigma_{\mathbf{n}_i} + \Sigma_{\mathbf{a}_i} \tag{32}$$

Developping (32) with (23)(13)(12), we can write the covariance as follows

$$\Sigma_{\mathbf{e}_i} = \Sigma_{\mathbf{a}_i} + R \Omega_{\bar{\boldsymbol{c}}_i} R^T \tag{33}$$

where

$$\Omega_{\bar{\boldsymbol{c}}_i} = \Sigma_{\mathbf{c}_i} + U_{\bar{\boldsymbol{c}}_i} \Sigma_{\mathbf{q}} U_{\bar{\boldsymbol{c}}_i}^T \tag{34}$$

4.2 Optimization

After the association step, we have two sets of N associated points $\{a_i\}$ and $\{n_i\}$ with corresponding errors $\mathbf{e}_i = \mathbf{n}_i - \mathbf{a}_i$. The optimization step consists in searching for the transformation $\mathbf{q}^{(k+1)}$ minimizing the squared Mahalanobis distance of errors

$$\mathbf{q}^{(k+1)} = \arg \min_{\mathbf{q} \in SE(3)} F = f^2 = \bar{\boldsymbol{e}}^T \Sigma_{\mathbf{e}}^{-1} \bar{\boldsymbol{e}} \tag{35}$$

where the errors are concatenated into a single vector $\bar{\boldsymbol{e}} = \left[\bar{\boldsymbol{e}}_i^T\right]^T$. The covariance matrix $\Sigma_{\mathbf{e}}^{-1}$ is a block-diagonal matrix with its i-th block being $\Sigma_{\mathbf{e}_i}$. This is generally done iteratively using non-linear least square method such as Levenberg-Marquardt (LM) algorithm.

In this work, we optimize directly on $SE(3)$ manifold. If we omit the superscript k for the current ICP iteration, the l-th iteration of LM solves

$$\begin{aligned}
\boldsymbol{\xi}^{(l+1)} &= \arg\min_{\boldsymbol{\xi}\in\mathbb{R}^6} f^2\left(\bar{\boldsymbol{q}}^{(l)} \boxplus \boldsymbol{\xi}\right) \\
&= -\left(J_{\mathbf{q}}^{f^{(l)}\,T} J_{\mathbf{q}}^{f^{(l)}} + \lambda^{(l)} I\right)^{-1} J_{\mathbf{q}}^{f^{(l)}} f\left(\bar{\boldsymbol{q}}^{(l)}\right) \qquad (36)\\
\bar{\boldsymbol{q}}^{(l+1)} &= \bar{\boldsymbol{q}}^{(l)} \boxplus \boldsymbol{\xi}^{(l+1)} \qquad (37)
\end{aligned}$$

with λ the dumping parameter of LM. Note here the difference with equation (32) from [19]. In our case, we do not make the assumption that $\Sigma_{\mathbf{e}}$ is constant relative to the displacement $\mathbf{q}$.

To alleviate the notation, we omit the LM iteration superscript.

Proposition 1. *The differential $DF_{\mathbf{q}}$ of $F = f^2$ at $\mathbf{q} \in SE(3)$ can be expressed as*

$$DF_{\mathbf{q}}(T\widehat{\boldsymbol{\xi}}) = \sum_{i=1}^{N} \bar{\boldsymbol{e}}_i^T \Sigma_{\mathbf{e}_i}^{-1} R\left(2U_{\bar{\mathbf{c}}_i}\boldsymbol{\xi} + \left(\Omega_{\bar{\mathbf{c}}_i}\widehat{\boldsymbol{\omega}} - \widehat{\boldsymbol{\omega}}\Omega_{\bar{\mathbf{c}}_i}\right) R^T \Sigma_{\mathbf{e}_i}^{-1} \bar{\boldsymbol{e}}_i\right),\ \widehat{\boldsymbol{\xi}} \in \mathfrak{se}(3) \qquad (38)$$

with U, Ω defined by (??)(34). The jacobian $J_{|\mathbf{q}}^F$ is obtained using (7) and $J_{|\mathbf{q}}^f$ follows by

$$J_{|\mathbf{q}}^f = \frac{1}{2f(\mathbf{q})} J_{|\mathbf{q}}^F \qquad (39)$$

In particular, note that

$$\forall i \in \{4,5,6\},\ \widehat{\boldsymbol{\xi}} = G_i^{SE(3)} \implies \widehat{\boldsymbol{\omega}} = 0 \qquad (40)$$

This leads to a simple expression for the last 3 columns of $J_{|\mathbf{q}}^F$

$$J_{|\mathbf{q}}^F[4:6] = 2\sum_{i=1}^{N} \bar{\boldsymbol{e}}_i^T \Sigma_{\mathbf{e}_i} R = \bar{\boldsymbol{e}}^T \Sigma_{\mathbf{e}} \left(\mathbf{1}_N \otimes R\right) \qquad (41)$$

The proof is given in Appendix A.2.1.

4.3 Covariance estimation

In this section, we are interested on estimating the covariance $\Sigma_{\mathbf{q}}$ of the previously optimized transformation $\mathbf{q}$. A closed-form expression is given by [8] as follows

$$\begin{aligned}
\Sigma_{\mathbf{q}_{min}} &= \left(\frac{\partial^2 F}{\partial \mathbf{q}^2}\right)^{-1} \frac{\partial^2 F}{\partial \mathbf{q}\partial \mathbf{z}} \Sigma_z \frac{\partial^2 F}{\partial \mathbf{q}\partial \mathbf{z}}^T \left(\frac{\partial^2 F}{\partial \mathbf{q}^2}\right)^{-1} \\
&= \left(H_{\mathbf{q}}^F\right)^{-1} H_{\mathbf{q},\mathbf{z}}^F \Sigma_z H_{\mathbf{q},\mathbf{z}}^{F^T} \left(H_{\mathbf{q}}^F\right)^{-1} \qquad (42)
\end{aligned}$$

with $F = f^2$ and the hessian of F :

$$H^F = \begin{bmatrix} H_{\mathbf{q}}^F & H_{\mathbf{q},\mathbf{z}}^F \\ H_{\mathbf{q},\mathbf{z}}^F & H_{\mathbf{z}}^F \end{bmatrix} \qquad (43)$$

The Gaussian vector $\mathbf{z} = [\mathbf{z}_1^T \dots \mathbf{z}_p^T]$ with $\mathbf{z}_i^T = [\mathbf{a}_i^T \mathbf{c}_i^T]$ is a vector of dimension $6p$ concatenating the points from both clouds. Its covariance matrix is defined by the block-diagonal matrix of dimension $6p \times 6p$

$$\Sigma_{\mathbf{z}} = \begin{bmatrix} \Sigma_{a_1} & & \dots & \dots & & 0 \\ & \Sigma_{c_1} & & & & \\ \vdots & & \ddots & & & \vdots \\ \vdots & & & \ddots & & \vdots \\ & & & & \Sigma_{a_p} & \\ 0 & & \dots & \dots & & \Sigma_{c_p} \end{bmatrix} \qquad (44)$$

Algorithm 2 On-manifold Probabilistic Iterative Closest Point

Input: Initial gaussian transformation $\mathbf{q}$, point clouds $S_{ref} = \{\mathbf{r}_i\}$, $S_{new} = \{\mathbf{c}_i\}$, Maximum iteration k_{max}, Association scheme T_{assoc}

Output: Relative gaussian transformation $\mathbf{q}^*$ between S_{new} and S_{ref}

$k = 0,\ \mathbf{q}^* = \mathbf{q}$
do
  for $(i = 1; i \leq m; i++)$ **do**
    $\bar{\boldsymbol{n}}_i = \bar{\boldsymbol{q}}^{(k)} \oplus \bar{\boldsymbol{c}}_i$
    $\Sigma_{\mathbf{n}_i} = J^{\oplus}_{\mathbf{q}} \Sigma_{\mathbf{q}} J^{\oplus^T}_{\mathbf{q}} + J^{\oplus}_{\mathbf{a}} \Sigma_{\mathbf{c}_i} J^{\oplus^T}_{\mathbf{a}}$
    $A_i = \left\{\mathbf{r} \in S_{ref} | D^2(\mathbf{n}_i, \mathbf{r}) < \chi^2_{3,\alpha}\right\}$
    $\mathbf{a}_i = \textbf{PICP\_Association}(T_{assoc}, A_i, \mathbf{n}_i, S_{ref})$
    $\bar{\boldsymbol{e}}_i = \mathbf{n}_i - \mathbf{a}_i$
    $\Sigma_{\mathbf{e}_i} = \Sigma_{\mathbf{n}_i} + \Sigma_{\mathbf{a}_i}$
  end for
  $F = \frac{1}{2} \sum_{i=1}^{m} \bar{\boldsymbol{e}}_i^T \Sigma_{\mathbf{e}_i} \bar{\boldsymbol{e}}_i$
  for $(l = 0; l < l_{max}; l++)$ **do**
    $\boldsymbol{\xi}^* = \arg\min_{\boldsymbol{\xi}} F(\bar{\boldsymbol{q}}^{(k)} \boxplus \boldsymbol{\xi})$
    $\bar{\boldsymbol{q}}^{(k+1)} = \bar{\boldsymbol{q}}^{(k)} \boxplus \boldsymbol{\xi}^*$
  end for
  $k++$
while (!hasConverged() and $k < k_{max}$)
$\bar{\boldsymbol{q}}^* = \bar{\boldsymbol{q}}^{(k)}$
$\Sigma_{\mathbf{q}^*} = \left(H^F_{\mathbf{q}}\right)^{-1} H^F_{\mathbf{q},\mathbf{z}} H^{F\ T}_{\mathbf{q},\mathbf{z}} \left(H^F_{\mathbf{q}}\right)^{-1}$

Proposition 2. *The hessian $Hess(F)_{\mathbf{q}}$ is given for all $\widehat{\boldsymbol{\xi}} \in \mathfrak{se}(3)$*

$$\begin{aligned} Hess(F)_{\mathbf{q}}(T\widehat{\boldsymbol{\xi}}, T\widehat{\boldsymbol{\xi}}) \quad = \quad & 2\Big(\boldsymbol{\xi}^T U_{\bar{\boldsymbol{c}}}^T \Big(\Omega_{\bar{\boldsymbol{c}}}^{-1} U_{\bar{\boldsymbol{c}}} \boldsymbol{\xi} - (\Omega_{\bar{\boldsymbol{c}}}^{-1} \widehat{\boldsymbol{\omega}} - \widehat{\boldsymbol{\omega}} \Omega_{\bar{\boldsymbol{c}}}^{-1}) R^T \bar{\boldsymbol{e}}\Big) + \\ & \bar{\boldsymbol{e}}^T R \Big(\Omega_{\bar{\boldsymbol{c}}}^{-1} \widehat{\boldsymbol{\omega}} \left(\widehat{\boldsymbol{\omega}} \bar{\boldsymbol{c}} + \tau\right) + \widehat{\boldsymbol{\omega}} \Omega_{\bar{\boldsymbol{c}}}^{-1} \widehat{\boldsymbol{\omega}} R^T \bar{\boldsymbol{e}}\Big)\Big) \end{aligned} \tag{45}$$

The corresponding hessian matrix $H^F_{\mathbf{q}}$ is obtained using (18) and (19). Similarly to Proposition **??**, *the bottom-right 3×3 submatrix of $H^F_{\mathbf{q}}$ can be simply express by*

$$H^F_{\mathbf{q}}[3:6, 3:6] = 2\Omega_{\bar{\boldsymbol{c}}}^{-1} \tag{46}$$

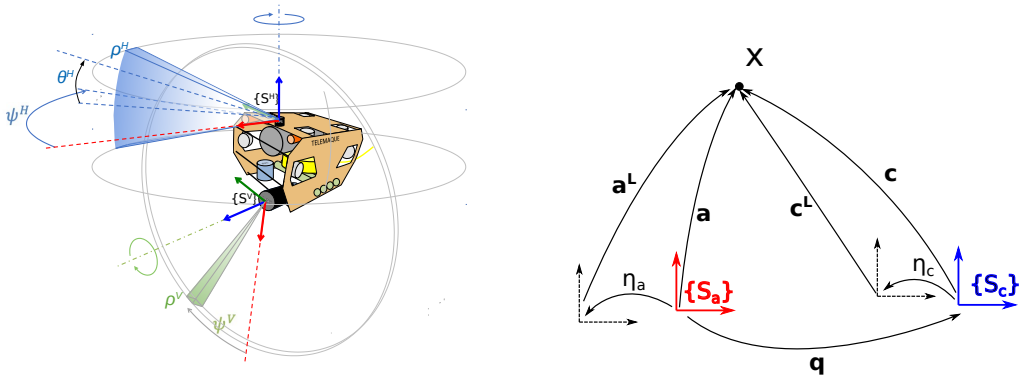
The derivation is given in Appendix A.2.2.
For $H^F_{\mathbf{q},\mathbf{z}}$, the expressions are more involved and are provided in Section A.2.4

5 Application to Sonar Data with unknown elevation angle

In this section, we propose a variant of probabilistic ICP adapted to sonar data which are not normally distributed. Indeed, sonar range measurements are obtained with beams which could have large overture. Thus, the elevation angle θ of the measurement is generally unknown and the 3D information is lost . Note that our approach is not limited to sonar and can apply to any other data involving similar data point distributions.

5.1 Notations

In the following, we reuse the notation introduced in Section 4 with some additions. We denote by b the sonar beam-width such that $\theta \in \left[-\frac{b}{2}, \frac{b}{2}\right]$. We also define $\boldsymbol{\eta}_{\mathbf{p}_i} \in SE(3)$ the poses relative to the current scan reference frame at which the data point $\mathbf{p}_i$ has been measured. The corresponding coordinates in the sonar local frame are denoted $\mathbf{p}_i^L$. The function g maps local spherical coordinates



(a) Illustration of the robot mounted with two perpendicular sonar

(b) Illustration of both scans (TODO : basique en attendant)

Figure 2: Notations related to the Robot sensing and local scans (TODO : refaire)

to local Cartesian coordinates and is given by

$$g : \begin{cases} [0,\ +\infty] \times [-\frac{b}{2}, \frac{b}{2}] \times [0,\ 2\pi] \to \mathbb{R}^3 \\ \rho, \theta, \phi \mapsto \mathbf{p}^L = \rho \begin{bmatrix} \cos(\theta)\cos(\phi) \\ \cos(\theta)\sin(\phi) \\ \sin(\theta) \end{bmatrix} \end{cases} \tag{47}$$

Similarly, we define the function h which maps the local spherical coordinates to the global reference frame in Cartesian coordinates

$$h : \begin{cases} [0,\ +\infty] \times [-\frac{b}{2}, \frac{b}{2}] \times [0,\ 2\pi] \times SE(3) \to \mathbb{R}^3 \\ \rho, \theta, \phi, \eta \mapsto \eta \oplus g(\rho, \theta, \phi) \end{cases} \tag{48}$$

Hence for each data point $\mathbf{p}$ from any of the two point clouds

$$\exists \rho, \theta, \phi,\ \text{s.t.}\ \mathbf{p} = h\left(\rho, \theta, \phi, \boldsymbol{\eta}_{\mathbf{p}}\right) \tag{49}$$

Figure **??** illustrates the notation and the problem configuration.

In the sonar local frame, the measurements in spherical coordinates $[\rho\ \theta\ \phi]^T$ can be modelized as follows

$$\rho_i \sim Gamma\left(\alpha_{\rho_i}, \beta_{\rho_i}\right) \tag{50}$$

$$\theta_i \sim SBeta\left(\alpha_i, \beta_i, -\frac{b}{2}, \frac{b}{2}\right) \tag{51}$$

$$\phi_i \sim \mathcal{N}\left(\bar{\phi}_i, \sigma^2_{\phi_i}\right) \tag{52}$$

where $SBeta(\alpha, \beta, min, max)$ is a shifted and scaled Beta distribution.

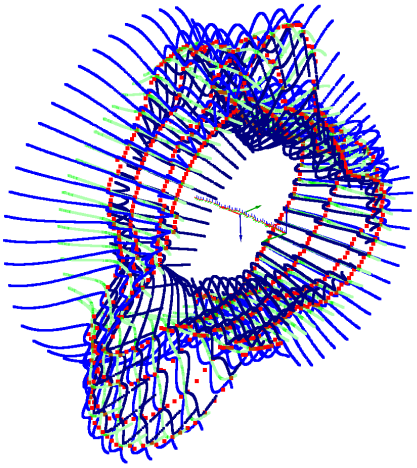
In practice, the mean and variance of ρ are such that it can be approximated by a normal distribution. In the following we use this approximation by noting

$$\rho_i \sim \mathcal{N}\left(\bar{\rho}_i, \sigma^2_{\rho_i}\right) \tag{53}$$

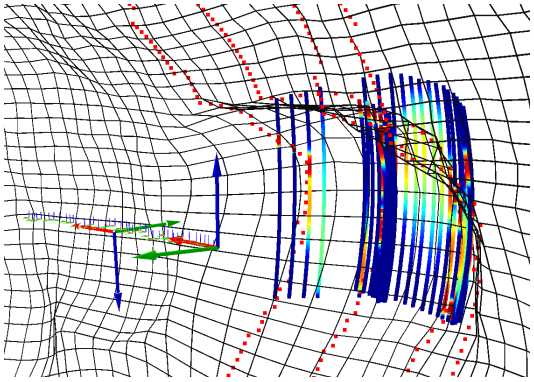
Note that θ modelization is valid even when no prior information is available for the elevation angle θ. Indeed, in this case, θ follows a uniform distribution $\theta \in \mathcal{U}\left(\frac{b}{2},\ \frac{b}{2}\right)$ which is also a Beta distribution as we have

$$\forall a, b \in \mathbb{R},\ \mathcal{U}(a, b) = SBeta(1, 1, a, b) \tag{54}$$

In order to estimate the displacement between two points cloud following the presented distribution, we first need to use a Gaussian approximation for the points in global Cartesian coordinates. We can then apply the on-manifold pIC presented in Section 4.



(a) Surface estimation



(b) Scaled-beta distribution of estimated elevation angles for horizontal sonar measurements from the first scan

Figure 3: Illustration of the elevation angle estimation method in [5] on two successives scans.

5.2 Elevation angles estimation

In this section, we briefly cover our previous work [5] which focuses on estimating the elevation angles from a wide-beam sonar. We also cover some improvements made in the current implementation used in this paper.

We consider a robot mounted with two perpendicular sonars : one placed vertically with narrow beams and the other placed horizontally with wide beams. This is illustrated in Figure **??**. This work aims at leveraging the information on the surface provided by the vertical sonar to estimated the elevation angles from the horizontal sonar measurements. This is done by estimating a stochastic model of the environment surface with a Gaussian process trained on the vertical sonar measurements. Formally, the Gaussian process $\mathcal{GP}$ is defined on the vertical sonar output (range) ρ^v as a function of the sonar rotation angle ψ_v and the pose $\boldsymbol{\eta}$ (in practice, the curvilinear abscissa s) along the robot trajectory such that

$$\rho^v = f(s, \psi^v) + \epsilon, \;\; f \sim \mathcal{GP} \tag{55}$$

with a centered noise $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$.

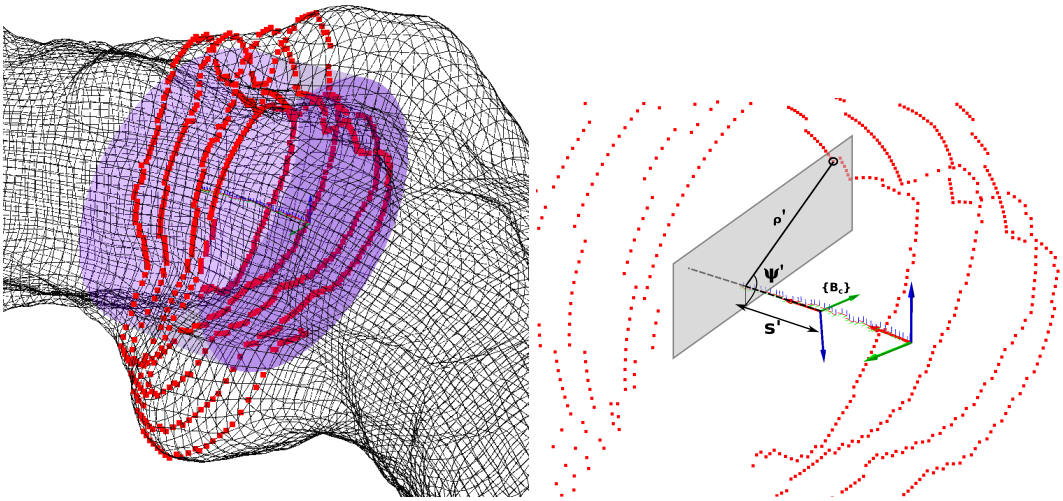
Once the Gaussian Process trained on the vertical sonar measurements, we sample each horizontal measurement beam into N angles. We estimate the likelihood of each angle as the corresponding 3D point likelihood to belong to the estimated surface. We also compute the associated uncertainty based on Fisher information. Thus, we obtain a scaled-beta distribution for the elevation angles for every horizontal measurement beams.

Two main modifications to [5] were made in this work and are detailed in the following sections.

5.2.1 Reference Curve

The Gaussian process in the original paper is trained on data given by the vertical sonar. The horizontal sonar data are thus expressed as if they were measured by the vertical sonar to be able to use the trained Gaussian process. Doing this, we have to consider the robot poses along its trajectory. This can lead to non-smooth surface estimation as the angular position of the robot are not explicitly taken in account in the Gaussian process kernels (TODO : figure ?). While this does not impact much the elevation angle estimations, it leads to wrong normal estimation required for the point-to-plane ICP.

Instead of using the vertical sonar as a reference for the Gaussian process and for horizontal sonar estimation, we can use an independent curve contained inside the environment for the scan length. Previously, sonar data were expressed by a robot pose $\boldsymbol{\eta}$, the sonar angle ψ and the range ρ. This is replaced by the point p on the curve such that the orthogonal plane to the curve at p contained the



(a) Prior Cylinder (b) Sonar data expressed with the reference curve

Figure 4: Modification of original method in [5] using a reference curve instead of the robot trajectory

sonar measurement. ρ, ψ are then the polar coordinates of the sonar measurement in this orthogonal plane. This is illustrated in Figure 4. Doing this, both the vertical and horizontal measurements are expressed relative to the reference curve.

In this work, we use the prior cylinder (see [5] section 4.2.1) axis as our reference curve. Let note respectively $B_c = [\mathbf{x_c}\ \mathbf{y_c}\ \mathbf{z}_c] \in \mathcal{M}_{3,3}$, $\mathbf{o}_c \in \mathbb{R}^3$ the cylinder orthornormal basis and center. Note that the first basis vector $\mathbf{x_c}$ corresponds to the axis cylinder.

If $\mathbf{p} \in \mathbb{R}^3$ is a point measured by the vertical or horizontal sonar, we then have the new abscissa $s' = \mathbf{x_c}^T(\mathbf{p} - \mathbf{o}_c)$ with range in the cylinder orthogonal plane $\rho' = ||\mathbf{y_c}^T(\mathbf{p} - \mathbf{o}_c)\mathbf{y}_c + \mathbf{z_c}^T(\mathbf{p} - \mathbf{o}_c)\mathbf{z}_c||$ and angle $\psi' = \arctan\left(\frac{\mathbf{z_c}^T(\mathbf{p}-\mathbf{o}_c)}{\mathbf{y_c}^T(\mathbf{p}-\mathbf{o}_c)}\right)$. This is illustrated in Figure 4.

In the case where $\mathbf{p}$ was measured by the vertical sonar, $s',\ \rho', \psi'$ replace respectively s, ρ^v, ψ^v in Eq (55). Previously, the surface was estimated based on the vertical sonar measurements. The horizontal sonar data were then expressed as if they were seen by the vertical sonar. We now replace the derivation in [5] section 4.2.3 computing $\hat{s}, \hat{\rho}^v, \hat{\psi}^v$ with s', ρ', ψ' which as a side effect simplify the computation.

5.2.2 Trajectory interpolation

While in [5] we assumed that robot odometry and sonar measurements were synchronous, this is generally not the case in practice. This requires to interpolate the robot poses at each sonar measurement. We use a simple on-manifold linear interpolation scheme with constant velocity between two robot poses measurement. Let consider $\boldsymbol{\eta}_1 \sim \mathcal{N}(\bar{\boldsymbol{\eta}}_1, \Sigma_{\eta_1}), \boldsymbol{\eta}_2 \sim \mathcal{N}(\bar{\boldsymbol{\eta}}_2, \Sigma_{\eta_2})$ two successives robot poses and a geodesic γ such that $\gamma(0) = \boldsymbol{\eta}_1$ and $\gamma'(0) = \log(\boldsymbol{\eta}_2 \ominus \boldsymbol{\eta}_1)$. Thus $\gamma(t) = e^{t\log(\boldsymbol{\eta}_2 \ominus \boldsymbol{\eta}_1)}$ and, for $t \in [0, 1]$, the interpolated mean pose $\bar{\boldsymbol{\eta}}(t)$ between $\bar{\boldsymbol{\eta}}_1$ and $\bar{\boldsymbol{\eta}}_2$ is given by

$$\bar{\boldsymbol{\eta}}(t) = \bar{\boldsymbol{\eta}}_1 \oplus e^{t\log(\boldsymbol{\eta}_2 \ominus \boldsymbol{\eta}_1)} \tag{56}$$

We also need to interpolate the uncertainty between the poses. The space of definite positive matrices is a smooth Riemannian manifold. In particular, we are interested here in the geodesics based on the Fisher-Rao metric [22]. As $\Sigma_{\eta_1}, \Sigma_{\eta_1}$ are two definite positives matrices, the geodesic $\Sigma_\eta(t)$ connecting Σ_{η_1} and Σ_{η_2} is given by

$$\Sigma_\eta(t) = \Sigma_{\eta_1}^{\frac{1}{2}} \left(\Sigma_{\eta_1}^{-\frac{1}{2}} \Sigma_{\eta_2} \Sigma_{\eta_1}^{-\frac{1}{2}}\right)^t \Sigma_{\eta_1}^{\frac{1}{2}},\ t \in [0, 1] \tag{57}$$

Note that the non-integer power of a matrix can be computed base on the Schur-Padé algorithm [13].

5.3 Normal estimation

In this section, we derive a method for computing normals distribution associated to each data point based on the Gaussian process regression of the environment surface. The normals are required for the point-to-plane association as described in Section 4.1. We first propose the derivation for the approach in [5]. The derivation is similar when using a reference curve and in particular the prior cylinder axis as explained in Section 5.2.1

As we do not have a closed-form expression of our surface in Cartesian coordinates, we approximate the normal at a point by sampling points on the surface around it and taking their cross product. Let $\mathbf{X}_p$ a point on the surface defined by

$$\begin{aligned} \mathbf{X}_p &= h\left(\rho_p,\ 0,\ \psi_p,\ \boldsymbol{\eta}^S(s_p)\right) && (58)\\ \rho_p &= f_{GP}(s_p, \psi_p) && (59)\end{aligned}$$

with $\boldsymbol{\eta}^S_{s_p} = \boldsymbol{\eta}^S(s_p)$ is the vertical sonar pose at the curvilinear abscissa s_p. We define the following R.V.

$$\begin{aligned} \rho_{s_p^\pm} &= f_{GP}\left(s_p \pm \delta s,\ \psi_p\right) + \epsilon,\\ \rho_{\psi_p^\pm} &= f_{GP}\left(s_p,\ \psi_p \pm \delta\psi\right) + \epsilon\\ \boldsymbol{\eta}^S_{s_p^\pm} &= \boldsymbol{\eta}^S\left(s_p \pm \delta s\right)\\ \mathbf{u}(\psi) &= \begin{bmatrix}\cos(\psi_p) & \sin(\psi_p) & 0\end{bmatrix}^T\\ \mathbf{u}_\pm &= \mathbf{u}(\psi_p \pm \delta\psi) && (60)\end{aligned}$$

where $\delta s, \delta\psi$ are small increments. We note $R^S_\pm$, $\mathbf{t}^S_\pm$ the corresponding rotation matrices and translation vectors. Thus we have the sampled points around $\mathbf{X}_p$

$$\begin{aligned} \mathbf{X}_{s_p^\pm} &= h\left(\rho_{s^\pm},\ 0,\ \psi_p,\ \boldsymbol{\eta}^S_{s_p^\pm}\right) && (61)\\ \mathbf{X}_{\psi_p^\pm} &= h\left(\rho_{\psi^\pm},\ 0,\ \psi_p^\pm, \boldsymbol{\eta}^S_{s_p}\right) && (62)\end{aligned}$$

The cross-product between the vectors $\mathbf{X}_{s_p^+} - \mathbf{X}_{s_p^-}$ and $\mathbf{X}_{\psi_p^+} - \mathbf{X}_{\psi_p^-}$ gives the unnormalized normal $\mathbf{n}(\mathbf{X}_p)$ at $\mathbf{X}_p$

$$\begin{aligned} \mathbf{n}(\mathbf{X}_p) &= \left(\mathbf{X}_{s_p^+} - \mathbf{X}_{s_p^-}\right) \times \left(\mathbf{X}_{\psi_p^+} - \mathbf{X}_{\psi_p^-}\right)\\ &= \rho_{s_p^+}\rho_{\psi_p^+}\left(R^S_+\mathbf{u} \times R^S\mathbf{u}_+\right) + \rho_{s_p^-}\rho_{\psi_p^-}\left(R^S_-\mathbf{u} \times R^S\mathbf{u}_-\right)\\ &\quad -\rho_{s_p^-}\rho_{\psi_p^+}\left(R^S_-\mathbf{u} \times R^S\mathbf{u}_+\right) - \rho_{s_p^+}\rho_{\psi_p^-}\left(R^S_+\mathbf{u} \times R^S\mathbf{u}_-\right)\\ &\quad +\rho_{\psi_p^+}\left(\left(\mathbf{t}^\mathbf{S}_+ - \mathbf{t}^\mathbf{S}_-\right) \times R\mathbf{u}_+\right) - \rho_{\psi_p^-}\left(\left(\mathbf{t}^\mathbf{S}_+ - \mathbf{t}^\mathbf{S}_-\right) \times R\mathbf{u}_-\right)\\ &= \rho_{s_p^+}\rho_{\psi_p^+}a_0 + \rho_{s_p^-}\rho_{\psi_p^-}a_1 + \rho_{s_p^-}\rho_{\psi_p^+}a_2 + \rho_{s_p^+}\rho_{\psi_p^-}a_3\\ &\quad +\rho_{\psi_p^+}b_0 - \rho_{\psi_p^-}b_1\\ &= \mathbf{n}\left(\rho_{s_p^+}, \rho_{\psi_p^+}, \rho_{s_p^-}, \rho_{\psi_p^-}\right) && (63)\end{aligned}$$

In order to compute the mean and covariance of $\mathbf{n}$, we need the covariance matrix Σ of the joint gaussian vector $\begin{bmatrix}\rho_{s_p^+} & \rho_{\psi_p^+} & \rho_{s_p^-} & \rho_{\psi_p^-}\end{bmatrix}^T$. It can be computed using the formula (2.24) provided in [26], p16.

For any order 2 term in (63), we have $E\left(\rho_{s_p^\pm}\rho_{\psi_p^\pm}\right) = \sigma_{\rho_{s_p^\pm}\rho_{\psi_p^\pm}} + \bar{\rho}_{s_p^\pm}\bar{\rho}_{\psi_p^\pm}$ and thus

$$\begin{aligned} \bar{\mathbf{n}}\left(\mathbf{X_p}\right) &= \mathbf{n}\left(\bar{\rho}_{s_p^+}, \bar{\rho}_{\psi_p^+}\bar{\rho}_{s_p^-}, \bar{\rho}_{\psi_p^-}\right) + a_0\sigma_{\rho_{s_p^+}\rho_{\psi_p^+}} + a_1\sigma_{\rho_{s_p^-}\rho_{\psi_p^-}}\\ &\quad +a_2\sigma_{\rho_{s_p^-}\rho_{\psi_p^+}} + a_3\sigma_{\rho_{s_p^+}\rho_{\psi_p^-}} && (64)\end{aligned}$$

The covariance $Var(\mathbf{n})$ of $\mathbf{n}$ is obtained with classic relation $Var(\mathbf{n}) = E\left(\mathbf{n}\mathbf{n}^T\right) - \bar{\mathbf{n}}\bar{\mathbf{n}}^T$. We therefore need to compute the 4-order multivariate polynomial $\mathbf{n}\mathbf{n}^T$. It has the general expression

$$\mathbf{n}\mathbf{n}^T = \sum_{\substack{i,j,k,l=0\\ i+j+k+l\leq 4}}^{4} \rho^i_{s_p^+}\rho^j_{\psi_p^+}\rho^k_{s_p^+}\rho^l_{\psi_p^-}A_{i,j,k,l} \qquad (65)$$

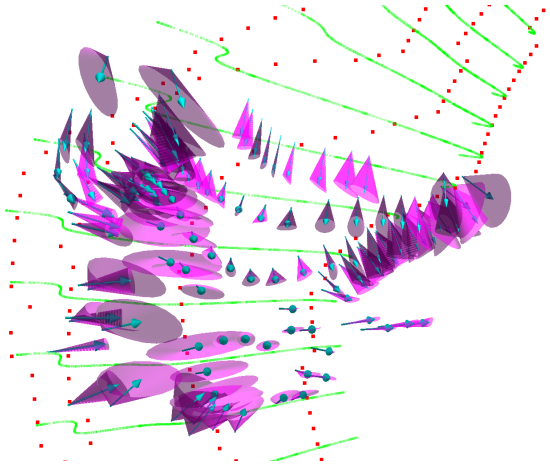


Figure 5: Estimated mean normals (turquoise arrow) and their 1-σ uncertainty cone (purple)

where $A_{i,j,k,l}$ are 3×3 symmetric matrices defined by outer product between the a_i and b_j defined in (64). Their expressions are given in Appendix A.3. It follows

$$Var(\mathbf{n}) = \sum_{\substack{i,j,k,l=0 \\ i+j+k+l \leq 4}}^{4} E\left(\rho^i_{s_p^+} \rho^j_{\psi_p^+} \rho^k_{s_p^+} \rho^l_{\psi_p^-}\right) A_{i,j,k,l} - \bar{\mathbf{n}}\bar{\mathbf{n}}^T \tag{66}$$

The covariance term of order 3 and 4 are computed thanks to the Isserlis theorem which, in case of centered joint Gaussian RV Z_i, gives $E(Z_1 Z_2 Z_3) = 0$ and $E(Z_1 Z_2 Z_3 Z_4) = \sigma_{12}\sigma_{34} + \sigma_{13}\sigma_{24} + \sigma_{14}\sigma_{23}$. We deduce then, for non-centered variables Y_i

$$\begin{aligned} E(Y_1 Y_2 Y_3) &= \sigma_{23} Y_1 + \sigma_{13} Y_2 + \sigma_{12} Y_3 + \bar{Y}_1 \bar{Y}_2 \bar{Y}_3 & (67) \\ E(Y_1 Y_2 Y_3 Y_4) &= \sigma_{12}\sigma_{34} + \sigma_{13}\sigma_{24} + \sigma_{14}\sigma_{23} + \sigma_{12}\bar{Y}_3\bar{Y}_4 + \sigma_{13}\bar{2}\bar{4} + \sigma_{14}\bar{Y}_2\bar{Y}_3 \\ &\quad + \sigma_{23}\bar{Y}_1\bar{Y}_4 + \sigma_{24}\bar{Y}_1\bar{Y}_3 + \sigma_{34}\bar{Y}_1\bar{Y}_2 + \bar{Y}_1\bar{Y}_2\bar{Y}_3\bar{Y}_4 & (68) \end{aligned}$$

Finally, we obtain the distribution on the normalized normal by noting f_n the normalization operator such that

$$\begin{aligned} f_n(\mathbf{n}) &= \frac{\mathbf{n}}{||\mathbf{n}||} \sim \mathcal{N}\left(f_n(\bar{\mathbf{n}}), J^{f_n}_{|\bar{\mathbf{n}}} \Sigma_{\mathbf{n}} J^{f_n^T}_{|\bar{\mathbf{n}}}\right) & (69) \\ J_{f_n}(\mathbf{n}) &= \frac{1}{||\mathbf{n}||^3} \begin{bmatrix} ||\mathbf{n}||^2 - n_x^2 & -n_x n_y & -n_x n_z \\ -n_x n_y & ||\mathbf{n}||^2 - n_y^2 & -n_y n_z \\ -n_x n_z & -n_y n_z & ||\mathbf{n}||^2 - n_z^2 \end{bmatrix} & (70) \end{aligned}$$

Note that here we have $d_i = \mathbf{v}_i^T \mathbf{a}_i$ in the general expression of the associated point $\mathbf{a}^\perp$ defined in (29). It follows

$$\bar{\boldsymbol{a}}_i^\perp = \bar{\boldsymbol{n}}_i - \bar{\boldsymbol{v}}_i^T \left(\bar{\boldsymbol{n}}_i - \bar{\boldsymbol{a}}_i\right) \bar{\boldsymbol{v}}_i \tag{71}$$

with the expression of the jacobians

$$\begin{aligned} J^{\mathbf{a}_i^\perp}_{\mathbf{n}_i} &= I_3 - \bar{\boldsymbol{v}}_i \bar{\boldsymbol{v}}_i^T & (72) \\ J^{\mathbf{a}_i^\perp}_{\mathbf{v}_i} &= \left(\bar{\boldsymbol{v}}_i \left(\bar{\boldsymbol{n}}_i - \bar{\boldsymbol{a}}_i\right)^T + \bar{\boldsymbol{v}}_i^T \left(\bar{\boldsymbol{n}}_i - \bar{\boldsymbol{a}}_i\right) I_3\right) J_{f_n}(\bar{\boldsymbol{v}}_i) & (73) \end{aligned}$$

Figure **??** shows an example of normals and their associated incertitude cone estimated as explained in this section.

5.4 Gaussian approximation

Let $\mathbf{p}^L \in \mathbb{R}^3$ a measured point in the sonar local frame. To compute its gaussian approximation, we have to estimate its mean $\bar{\boldsymbol{p}}^L$ and covariance matrix Σ_{p^L}. From (47) we have directly

$$\bar{\boldsymbol{p}}^L = \bar{\rho} \begin{bmatrix} E(\cos\phi)E(\cos\theta) \\ E(\sin\phi)E(\cos\theta) \\ E(\sin\theta) \end{bmatrix} \tag{74}$$

The covariance matrix is given by the classic relation

$$\begin{aligned} \Sigma_{p^L} &= E(\mathbf{X}\mathbf{X}^T) - \bar{\mathbf{X}}\bar{\mathbf{X}}^T \\ &= \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_z^2 \end{bmatrix} \end{aligned} \tag{75}$$

with

$$\sigma_x^2 = E(\rho^2)E(\cos^2\phi)E(\cos^2\theta) - \bar{\rho}^2 E(\cos\phi)^2 E(\cos\theta)^2 \tag{76}$$

$$\sigma_y^2 = E(\rho^2)E(\sin^2\phi)E(\cos^2\theta) - \bar{\rho}^2 E(\sin\phi)^2 E(\cos\theta)^2 \tag{77}$$

$$\sigma_z^2 = E(\rho^2)E(\sin^2\theta) - \bar{\rho}^2 E(\sin\theta)^2 \tag{78}$$

$$\sigma_{xy} = E(\rho^2)E(\cos\phi\sin\phi)E(\cos^2\theta) - \bar{\rho}^2 E(\cos\phi)E(\sin\phi)E(\cos\theta)^2 \tag{79}$$

$$\sigma_{xz} = E(\rho^2)E(\cos\phi)E(\cos\theta\sin\theta) - \bar{\rho}^2 E(\cos\phi)E(\cos\theta)E(\sin\theta) \tag{80}$$

$$\sigma_{yz} = E(\rho^2)E(\sin\phi)E(\cos\theta\sin\theta) - \bar{\rho}^2 E(\sin\phi)E(\cos\theta)E(\sin\theta) \tag{81}$$

$$E(\rho^2) = \bar{\rho}^2 + \sigma_\rho^2 \tag{82}$$

The following propositions give the expressions of the different involved expectations over trigonometric functions

Proposition 3. *Let θ a random value following a shifted and scale Beta distribution $SBeta(\alpha, \beta, -\frac{b}{2}, \frac{b}{2})$. Then*

$$E(\cos\theta) = \sum_{n=0}^{+\infty} \frac{(-1)^n}{2n!} \left(\frac{b}{2}\right)^{2n} {}_2F_1(-2n, \alpha, \alpha+\beta; 2) \tag{83}$$

$$E(\cos^2\theta) = 1 + \frac{1}{2}\sum_{n=1}^{+\infty} \frac{(-1)^n}{2n!} b^{2n} {}_2F_1(-2n, \alpha, \alpha+\beta; 2) \tag{84}$$

$$E(\sin\theta) = \frac{-b}{2}\sum_{n=0}^{+\infty} \frac{(-1)^n}{2n+1!} \left(\frac{b}{2}\right)^{2n} {}_2F_1(-2n-1, \alpha, \alpha+\beta; 2) \tag{85}$$

$$E(\cos\theta\sin\theta) = \frac{-b}{2}\sum_{n=0}^{+\infty} \frac{(-1)^n}{2n+1!} b^{2n} {}_2F_1(-2n-1, \alpha, \alpha+\beta; 2) \tag{86}$$

where ${}_2F_1$ is the hypergeometric function.

The proof is given in Appendix A.2.5. Note that $E(\cos\theta\sin\theta), E(\sin^2\theta)$ are deduced from Proposition 3 using the relations $\cos\theta\sin\theta = \frac{\sin 2\theta}{2}$ and $\sin^2\theta = 1 - \cos^2\theta$.

We also have the following error bounds on the previous expectations

Proposition 4. *Let $N > 0$ and $\theta \in [-\frac{b}{2}, \frac{b}{2}]$. The reminders of order N for the series in Proposition 3 are given by*

$$|R_N\left(E(\cos\theta)\right)| \leq \frac{1}{(2N)!}\left(\frac{3b}{2}\right)^{2N} \cosh\left(\frac{3b}{2}\right) \tag{87}$$

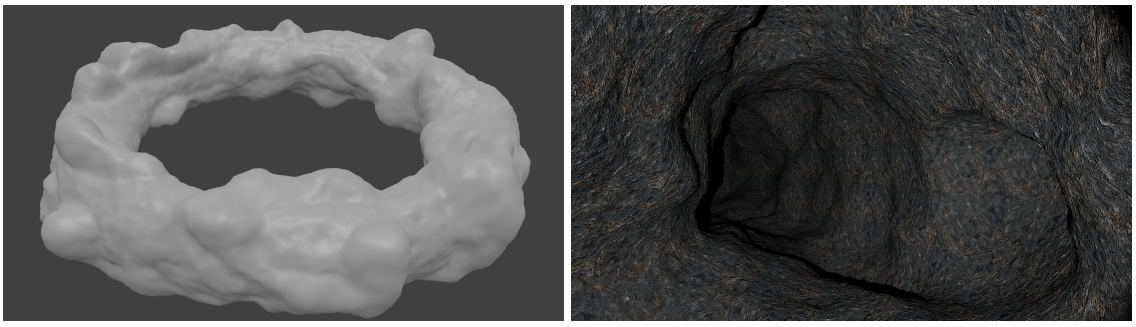
$$|R_N\left(E(\sin\theta)\right)| \leq \frac{1}{(2N+1)!}\left(\frac{3b}{2}\right)^{2N+1} \sinh\left(\frac{3b}{2}\right) \tag{88}$$

$$|R_N\left(E(\cos^2\theta)\right)| \leq \frac{1}{2}\frac{1}{(2N)!}(3b)^{2N}\cosh(3b) \tag{89}$$

$$|R_N\left(E(\sin\theta)\right)| \leq \frac{1}{2}\frac{1}{(2N+1)!}(3b)^{2N+1}\sinh(3b) \tag{90}$$

N	$E(\cos\theta)$	$E(\sin\theta)$	$E(\cos^2\theta)$	$E(\sin\theta)$
2	$4.23 * 10^{-2}$	$5.60 * 10^{-3}$	$7.47 * 10^{-1}$	$2.59 * 10^{-1}$
3	$1.18 * 10^{-3}$	$1.11 * 10^{-4}$	$8.33 * 10^{-2}$	$2.07 * 10^{-2}$
4	$1.76 * 10^{-5}$	$1.29 * 10^{-6}$	$4.98 * 10^{-3}$	$9.63 * 10^{-4}$
5	$1.64 * 10^{-7}$	$9.88 * 10^{-9}$	$1.85 * 10^{-4}$	$2.39 * 10^{-5}$

Table 1: Error bounds on expectation in Proposition 3



(a) Global view (b) Inside view

Figure 6: Karst model used for our experiments

The proof is given in Appendix A.2.6. The corresponding errors for small values of N in our application where $b = 0.61$ are given in Table 1. In our following experiments, we use $N = 5$.

Proposition 5. *Let* $\phi \sim \mathcal{N}\left(\bar{\phi}, \sigma_\phi^2\right)$. *Then*

$$E(\cos\phi) = \cos\bar{\phi}e^{-\frac{\sigma_\phi^2}{2}} \tag{91}$$

$$E(\sin\phi) = \sin\bar{\phi}e^{-\frac{\sigma_\phi^2}{2}} \tag{92}$$

$$E(\cos^2\phi) = \frac{1}{2}\left(1 + e^{-2\sigma_\phi^2}\cos 2\bar{\phi}\right) \tag{93}$$

$$E(\sin^2\phi) = \frac{1}{2}\left(1 - e^{-2\sigma_\phi^2}\cos 2\bar{\phi}\right) \tag{94}$$

$$E(\cos\phi\sin\phi) = \frac{1}{2}e^{-2\sigma_\phi^2}\sin 2\phi \tag{95}$$

The proof is given in Section A.2.7.

Finally, the pdf of the point $\mathbf{p} = \boldsymbol{\eta}_\mathbf{p} \oplus \mathbf{p}^L$ expressed in the current scan reference frame is given by

$$\mathbf{p} \sim \mathcal{N}\left(\bar{\boldsymbol{\eta}}_\mathbf{p} \oplus \bar{\boldsymbol{p}}^L, J^{\oplus}_{\mathbf{q}_{|\bar{\boldsymbol{\eta}}_\mathbf{p}, \bar{\boldsymbol{p}}^L}} \Sigma_{p^L} J^{\oplus^T}_{\mathbf{q}_{|\bar{\boldsymbol{\eta}}_\mathbf{p}, \bar{\boldsymbol{p}}^L}}\right) \tag{96}$$

where $\boldsymbol{\eta}_\mathbf{p}$ is the pose at which the point has been observed.

6 Experiments

In this section, we assess the performance of proposed algorithms with several experiments. First, in order to validate our computation, we simulate point clouds following probability distributions (50)(51)(52) with known association and apply our pICP algorithm.

In Section **??**, we generate data points from a simulated karst environment shown in Figure 6. We briefly introduce our method exposed in [5] used to estimate the distribution of elevation angles θ_i. We also detail some modification from our original work and propose a method for normal computation.

6.1 Evaluation of the optimization step in pIC

In this section, we assess the improvement of our on-manifold pICP compared to previous methods.

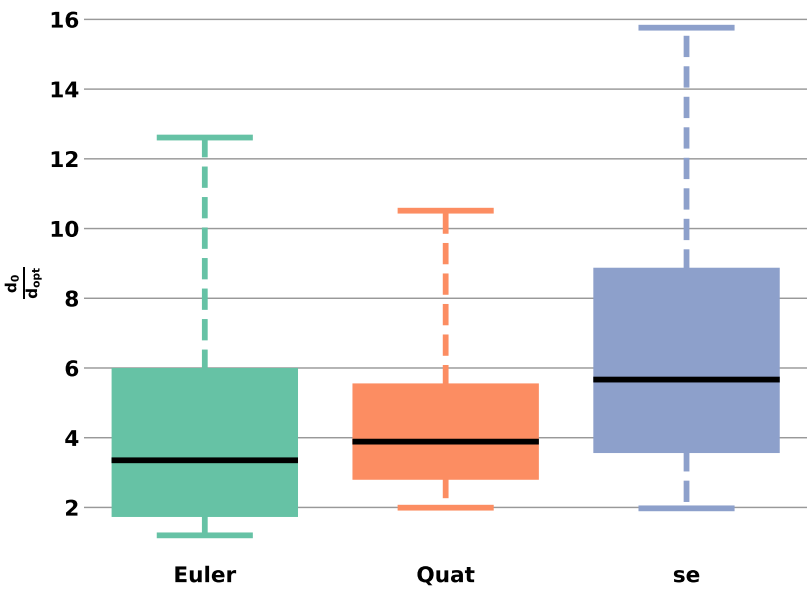


Figure 7: Comparison of distances ratio for 500 trials. Medians are indicated by black lines. The bottom/top whiskers correspond to 5%/95% of the distributions.

At each trial, we randomly generate 100 3D points distribution $\mathbf{c}_i$ and a random normal transform $\mathbf{q}$. From each distribution $\mathbf{c}_i$ we sample one point $\hat{\mathbf{c}}_i$ considered as a ground truth point. Similarly, we sample a ground truth transformation $\hat{\mathbf{q}}$ from $\mathbf{q}$. The ground truth points of the second point cloud are computed as $\hat{\mathbf{a}}_\mathbf{i} = \hat{q} \oplus \hat{\mathbf{c}}_i$. Finally, the distributions $\mathbf{a}_i$ are generated with random covariance Σ_{a_i} and the means $\bar{\boldsymbol{a}}_i$ as samples from the distributions $\mathcal{N}(\hat{\mathbf{a}}_i, \Sigma_{a_i})$. The initial transformation is given by $\mathbf{q}^{(0)} \sim \mathcal{N}(\bar{\boldsymbol{q}}^{(0)} = \bar{\boldsymbol{q}}, \Sigma_q)$.

First, we consider only the optimization step of PICP with perfectly associated points. We compare the optimized transformation q_{opt} obtained using Euler angles, quaternion and the proposed on-manifold optimization. The distance of q_{opt} to the ground truth $\hat{\mathbf{q}}$ is computed using (1) and the metric on $SE(3)$ defined by (4) such that

$$d_{opt} = d(q_{opt}, \hat{q}) = \sqrt{log\left(q_{opt} \ominus \hat{q}\right)^T G \; log\left(q_{opt} \ominus \hat{q}\right)} \tag{97}$$

In order to compare the different trials, we consider the ratio d_0/d_{opt} with the initial distance $d_0 = d(\bar{\boldsymbol{q}}^{(0)}, \hat{q})$.

In Figure 7, we represent the distributions of ratio for 500 trials and maximum of iterations set to 100. As expected, the Euler representation is more prone to failure depending on the trial configuration. The quaternion representation is more limited by its convergence rate due to the renormalization at each iteration step. The on-manifold approach provides the best results expectation-wise.

6.2 Quantitative evaluation on simulated karst environment

It is difficult to obtain ground truth data in field experiments for karst exploration. At best, real field experiments can be qualitatively analyzed to assess the pertinence of an approach. In majority of literature in underwater robotics, systems are globally evaluated but there are few "test-unit" of the algorithms composing it. For instance, in [15][17][19], the proposed adaption of the pIC algorithm is indirectly evaluated through its integration in a SLAM framework.

[6] first evaluates its pIC approach by matching real scans with them-selves and adding a Gaussian noise to the initial transformation estimate. While based on data from real experiments, it do not

Parameter	Value
Robot sensing	
Depth sensor std	0.016 m
IMU orientation std	0.16 deg
Dead-reckoning noise std on x,y	0.022 m
Dead-reckoning noise std on yaw	0.13 deg
MSIS	
Vertical sonar range resolution	0.2 m
Horizontal sonar range resolution	0.05 m
Horizontal sonar beam width	35 deg
Simulation	
Steps for full sonar scan	200
Vertical sonar period step	1
Horizontal sonar period step	2
Dead-reckoning odometry period step	10
Gaussian process	
Lengthscale l_s	Fixed (1)
Lengthscale l_ψ	Fixed (0.25)
Signal std σ_f	Fixed (0.64)
Noise std σ_n	Learned

Table 2: Parameters used in the simulation experiments

take in account for the surface sampling error e.g. associated points from each scan do not correspond exactly to the same point on the environment surface.

We propose here to evaluate quantitatively our on-manifold pIC independently from any SLAM framework. We also compare our approach to reimplementation of the 2D approach proposed [15] which has been successfully applied in real underwater caves mapping [17].

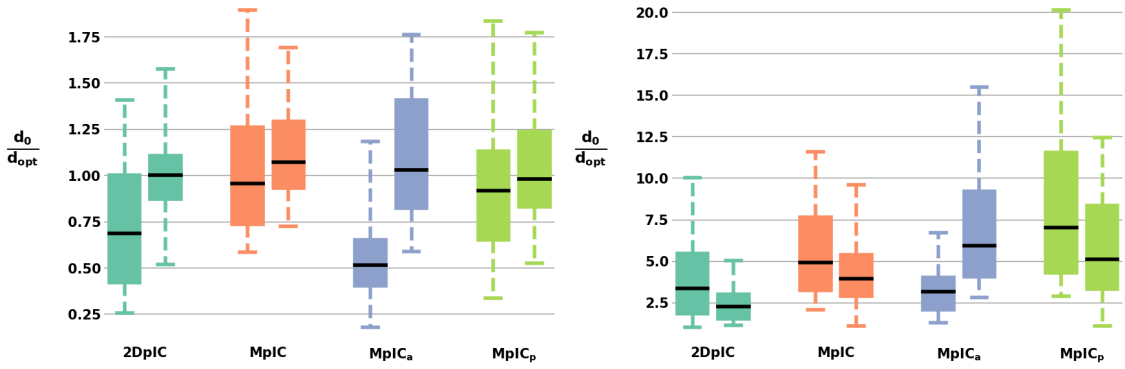
We consider for this a simulated karst environment as shown in Figure **??**. The ground-truth trajectory is sampled from a circle contained inside the karst model. At each time step, we generate a noisy odometry by adding Gaussian noise to the odometry obtained from the ground truth trajectory. The different sensors are simulated based on values provided in [16][17]. The different parameters are resumed in Table 2.

In a real experiments, dead-reckoning data (Kalman filter fusion of IMU and DVL sensors) and sonars are received asynchronously with an attached timeStamp. Here, timestamps are replaced by the simulation step. We consider the vertical sonar measurement as our reference so that each step corresponds to a vertical sonar measurement. The relative frequency of the other data are based on the real dataset provided by [17]. We can observe that we have between 2 and 3 full vertical sonar scan for one horizontal scan. Furthermore, one odometry data is received every 10 vertical sonar measurements. Thus, we consider an horizontal sonar measurement every 2 steps and one odometry data every 10 steps. Note that we only consider point clouds from the horizontal sonar for the pIC. Thus scan matching is executed every 400 steps excepted for the first time (800 steps to generate the two first scans).

The vertical and horizontal MSIS measurements are simulated by ray-tracing. The obtained ranges are then quantified to reproduce the original sonar resolution based on the max range and the number of bin intensity.

To obtain the robot pose at each sonar measurement, we interpolate the poses obtained by dead-reckoning as explained in Section 5.2.2.

The parameters used for the Gaussian process are shown in Table 2. Following [5], we use a product covariance kernel $K = K_s * K_\psi$ where K_s is a Matern52 kernel on the abscissa s and K_ψ a



(a) Successive scans (small initial transformation error) (b) Loop closure (large initial transformation error)

Figure 8: Diagrams of relative errors for successive scans and loop closure. **2DpIC** : pIC proposed by [15], **MpIC**: MpIC (point-To-point), **MpIC$_a$** : MpIC(point-to-point) including uniform arcs, **MpIC$_p$** : MpIC (point-to-plane). Left (resp. right) boxes in each group have been computed with association confidence level $\alpha = 0.95$ (resp. $\alpha = 0.5$)

Matern52 kernel based on angle chordal distance such that

$$K_s(s,s') = \sigma_f \left(1 + \frac{\sqrt{5}}{|s-s'|} l_s + \frac{5|s-s'|^2}{3l_s^2}\right) exp\left(-\frac{\sqrt{5}|s-s'|}{l_s}\right) \tag{98}$$

$$K_\psi(\psi,\psi') = \sigma_f \left(1 + \frac{\sqrt{5}}{2\sin\frac{|\psi-\psi'|}{2}} l_\psi + \frac{20\sin\frac{|\psi-\psi'|}{2}^2}{3l_\psi^2}\right) exp\left(-\frac{2\sqrt{5}\sin\frac{|\psi-\psi'|}{2}}{l_\psi}\right) \tag{99}$$

Recall that σ_n^2 is the variance of the Gaussian additive noise defined in Eq (55).

As in Section 6.1, we consider the ratio $\frac{d_0}{d_{opt}}$ and compare the distributions obtained with different approaches :

- **2DpIC** : implementation proposed by [15][16]
- **MpIC** : the proposed approach with point-to-point association
- **MpIC$_a$** : MpIC including measurements with a uniform distribution for the elevation angle
- **MpIC$_p$** : MpIC with point-to-plane association as explained in Section 5.3

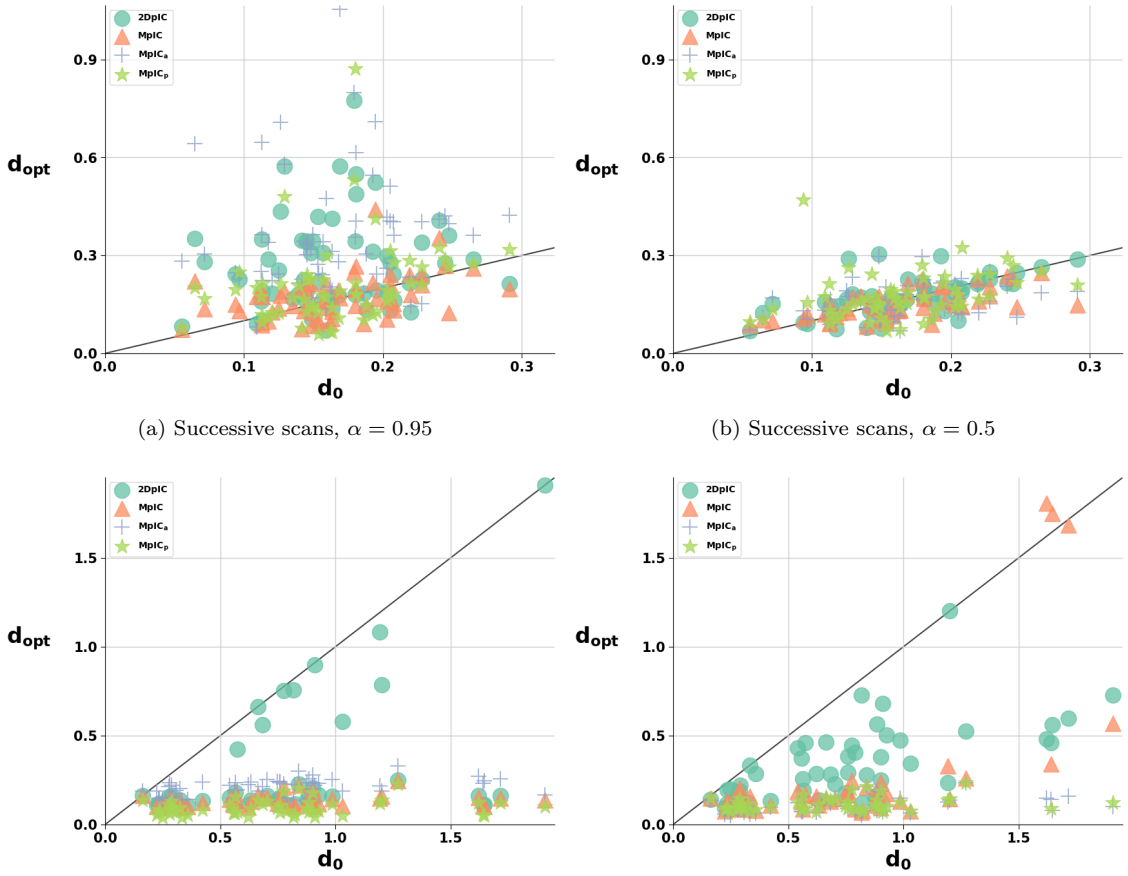
While this normalized ratio allows a global evaluation of the results, it has some drawbacks. For instance, a ratio of 1 has not the same meaning when d_0 is small or large. A more fine-grained representation in scatter plot is also proposed.

In order to assess the effect of the association threshold defined in Eq (27), we also compare the results obtained with two confidence level $\alpha = 0.95$ and $\alpha = 0.5$. The higher the confidence level, the higher the threshold $\chi^2_{2,\alpha}$ or $\chi^2_{3,\alpha}$ is. In Figures 8a and 8b, left (resp. right) boxes corresponds to $\alpha = 0.95$ (resp. $\alpha = 0.5$).

We consider two cases : scan matching of successive scans and loop closure. For the first one, we estimate the distance d_{opt} for each successive scan matching for one loop inside the modelized karst (61 scans matching). In order to simulate the same order of magnitude encountered in loop closure scan matching, we consider a pair of scans and set higher diagonal values to the initial transformation covariance σ_{q_0}. We sample the initial transformation $q_0 \sim \mathcal{N}(q_0, \sigma_{q_0})$ and apply the different pIC algorithms. We repeat this $n = 50$ times. In the following, we analyze and discuss the results obtained.

6.2.1 Successive scans

Figures 8a, 9c and 9d summed up the results obtained for the scan matching of successive scans.



(a) Successive scans, $\alpha = 0.95$

(b) Successive scans, $\alpha = 0.5$

(c) Loop closure, $\alpha = 0.95$

(d) Loop closure, $\alpha = 0.5$

Figure 9: Scatter plots for successive scans and loop closure. The black line corresponds to $x = y$ (unitary ratio)

The resulting distributions show that the point-to-point association for MpIC improved over the previous 2D approach. While the median is slightly higher, the important feature is a lower variance and higher minimum ratios. Note that here MpIC only use points inside the surface estimated with the vertical sonar. Points seen behind and in front of the robot are thus ignored. This means that the matched point cloud are smaller and less discriminative. Nevertheless, the recovery of the 3D information overcomes this drawback.

In order to deal with this drawback, we try to include the horizontal sonar measurements outside of the estimated surface by considering a uniform elevation angle distribution along the beam width. The results are somewhat a mixed of both the 2D and MpIC cases. It inherits an accuracy improvement from the 3D information but at the expense of higher variance as in the 2D case.

While point-to-plane ICP provides generally better performance than its point-to-point version, this is not the case here for small transformation.

Note that for successive scans, the 3D transformation uncertainty is relatively low. Indeed, the transformation uncertainty is the accumulation of low odometry uncertainties acquired along the scan trajectory. Informally, this means that the transformation uncertainty is to a large extend included in the interval of confidence obtainable with the pIC algorithm (mal-dit ... figure ?). In other words, pIC would provide only little improvements over the initial transformation hence the low ratio in Figure 8a. This is not the case when scan matching is done in a Loop closure. Here, the transformation uncertainty has been accumulated along several scans and is relatively larger than the local scan odometry uncertainties.

In regards to association confidence level, it is clear that $\alpha = 0.95$ gives relatively bad results regardless of the method used. It is intuitive as this confidence level corresponds to a higher association threshold and thus can potentially match points with lower likelihood. In the case of successive scans,

we can expect the initial transformation to be near the local optimum so that matching points are initially near from each other. Low likelihood points should then be discarded to avoid spurious associations.

6.2.2 Loop Closure

Similarly to the previous section, Figures 8b, **??** and **??** summed up the results with large initial transformation error.

Regardless of the method, improvements over dead-reckoning odometries are clearer than in the case of small transformation. This confirms our previous hypothesis on the fact that for successive scans, the initial error uncertainty is almost contained in the algorithm output uncertainty. In case of larger initial error, the point-to-plane MpIC (MpIC_p) gives more accurate results over point-to-point MpIC.

Unlike the previous case, for large initial transformation lower confidence level for association provides better associations. This is intuitive as in the first iterations, points are rather far between leading to higher Mahalanobis distances (lower likelihood). (Quoi rajouter ?)

7 Conclusion

In this paper we propose MpIC, an on-manifold derivation for the 3D probabilistic Iterative Closest Point (pIC) algorithm based on basic differential geometry related to the SE(3) Lie group. We then propose an application to underwater sonar scan matching in the context of karst exploration. We extended our previous work on elevation angle estimation from wide-beam sonar [5] to generate 3D point Gaussian distribution from the raw sonar measurements. The approach is evaluated on a simulated karst environment to allow quantitative analysis. It is also compared to the state-of-the-art approach proposed in [17].

In the robotic literature, ICP algorithms are generally evaluate indirectly in a SLAM framework and/or qualitatively on real data experiments [15][16][19]. Quantitative results, when available, are obtained by adding self-matching a scan with added noise [18][6]. We propose here a compromise by proposing a quantitative evaluation of ICP with a more complex setting similar to real experiment.

We shown that, as one could expect, MpIC outperforms pIC based on Euler angles and Quaternion representations. We then analyzed and compared different pIC approaches when applied to acoustic sonar data. Compared to previous 2D method, our MpIC approach converge to slightly more accurate transformation. While point-to-plane association scheme is generally more accurate than point-to-point, it is not the case in our experimental configuration. Here, point-to-plane is prone to converge to wrong local minima. Our experiments also shown more general results on pIC. For successive scan matching with normally distributed dead-reckoning noise, the different pIC methods provides only slight improvement on the estimated transformation. Depending on the environment configuration and sensors noise, it may be better to rely solely on dead-reckoning estimate. However, in real settings one can expect dead-reckoning noise outliers which do not follow the modelized Gaussian distribution.

For large initial transformations such as encountered in loop closure scan matching, MpIC also provides better results over the 2D approach. In particular, unlike to the previous case of successive scan, the point-to-plane association scheme outperforms point-to-point associations.

The association confidence level is an important parameter of the pIC algorithms. We show through our experiments that it should be adapted to the ratio between the initial transformation error and the scan local odometries error. This is rather intuitive : for small displacements, corresponding points are near from each other so that the association threshold should be low to avoid spurious associations. For larger displacements, correspondings points are far from each other requiring thus a larger association threshold.

While the results are promising, there is still room for improvements. We approximated our points distribution by Gaussian distributions. For points coming from sonar measurements with a high elevation angle, a large part of the Gaussian distribution covers an area outside of the original beam measurement. We consider relaxing the Gaussian approximation and use a more generic distance function to replace the Mahalanobis distance. This approach would certainly be costly computationnaly-wise but can be interesting to further increase localization accuracy offline. On the application side, this work was a necessary step towards an integration in a graphSLAM framework for karst aquifer exploration

References

[1] Gabriel Agamennoni et al. "Point clouds registration with probabilistic data association". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 4092–4098.

[2] Calin Belta and Vijay Kumar. "Euclidean metrics for motion generation on SE (3)". In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 216.1 (2002), pp. 47–60.

[3] Paul J Besl and Neil D McKay. "Method for registration of 3-D shapes". In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–606.

[4] Jose-Luis Blanco. "A tutorial on se (3) transformation parameterizations and on-manifold optimization". In: *University of Malaga, Tech. Rep* 3 (2010).

[5] Yohan Breux and Lionel Lapierre. "Elevation Angle Estimations of Wide-Beam Acoustic Sonar Measurements for Autonomous Underwater Karst Exploration". In: *Sensors* 20.14 (2020), p. 4028.

[6] Antoni Burguera. "A novel approach to register sonar data for underwater robot localization". In: *2017 Intelligent Systems Conference (IntelliSys)*. IEEE. 2017, pp. 1034–1043.

[7] Antoni Burguera, Yolanda González, and Gabriel Oliver. "The UspIC: Performing scan matching localization using an imaging sonar". In: *Sensors* 12.6 (2012), pp. 7855–7885.

[8] Andrea Censi. "An accurate closed-form estimate of ICP's covariance". In: *Proceedings 2007 IEEE international conference on robotics and automation*. IEEE. 2007, pp. 3167–3172.

[9] Andrea Censi. "An ICP variant using a point-to-line metric". In: *2008 IEEE International Conference on Robotics and Automation*. Ieee. 2008, pp. 19–25.

[10] Yang Chen and Gérard Medioni. "Object modelling by registration of multiple range images". In: *Image and vision computing* 10.3 (1992), pp. 145–155.

[11] Sébastien Druon, Marie-José Aldon, and André Crosnier. "Color constrained icp for registration of large unstructured 3d color data sets". In: *2006 IEEE International Conference on Information Acquisition*. IEEE. 2006, pp. 249–255.

[12] Shaoyi Du et al. "Robust rigid registration algorithm based on pointwise correspondence and correntropy". In: *Pattern Recognition Letters* 132 (2020), pp. 91–98.

[13] Nicholas J Higham and Lijing Lin. "A Schur–Padé algorithm for fractional powers of a matrix". In: *SIAM Journal on Matrix Analysis and Applications* 32.3 (2011), pp. 1056–1078.

[14] Jocelyn Quaintance Jean Gallier. *Differential Geometry and Lie Groups A computational Perspective*. 2019.

[15] Angelos Mallios et al. "Scan matching SLAM in underwater environments". In: *Autonomous Robots* 36.3 (2014), pp. 181–198.

[16] Angelos Mallios et al. "Toward autonomous exploration in confined underwater environments". In: *Journal of Field Robotics* 33.7 (2016), pp. 994–1012.

[17] Angelos Mallios et al. "Underwater caves sonar data set". In: *The International Journal of Robotics Research* 36.12 (2017), pp. 1247–1251.

[18] Luis Montesano, Javier Minguez, and Luis Montano. "Probabilistic scan matching for motion estimation in unstructured environments". In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2005, pp. 3499–3504.

[19] Albert Palomer, Pere Ridao, and David Ribas. "Multibeam 3D underwater SLAM with probabilistic registration". In: *Sensors* 16.4 (2016), p. 560.

[20] François Pomerleau, Francis Colas, and Roland Siegwart. "A review of point cloud registration algorithms for mobile robotics". In: *Foundations and Trends in Robotics* 4.1 (2015), pp. 1–104.

[21] Szymon Rusinkiewicz. "A symmetric objective function for ICP". In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), pp. 1–7.

[22] Salem Said et al. “Riemannian Gaussian distributions on the space of symmetric positive definite matrices”. In: *IEEE Transactions on Information Theory* 63.4 (2017), pp. 2153–2170.

[23] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. “Generalized-icp.” In: *Robotics: science and systems*. Vol. 2. 4. Seattle, WA. 2009, p. 435.

[24] James Servos and Steven L Waslander. “Multi-Channel Generalized-ICP: A robust framework for multi-channel scan registration”. In: *Robotics and Autonomous systems* 87 (2017), pp. 247–257.

[25] Joan Sola, Jeremie Deray, and Dinesh Atchuthan. “A micro Lie theory for state estimation in robotics”. In: *arXiv preprint arXiv:1812.01537* (2018).

[26] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.

A Appendices

A.1 Definitions and properties of matrix product

We note $M_{m,n}$ the vector space of real $m \times n$ matrices. To simplify notations, we note $[a_{ij}] \in M_{m,n}$ a matrix with generic term a_{ij}. Similarly, we define block matrices as $[B_{ij}]$. The determinant of a matrix M is noted $|M|$ when there is no ambiguity with the absolute value operator.

Definition 1. *Let* $\boldsymbol{\omega} = \begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^T \in \mathbb{R}^3$. *The operator* $[\ ]_\times$ *maps 3D vector to the space of skew-symmetric matrix and is defined as*

$$[\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

Using this operator, the cross product operation can be expressed as a matrix vector operation :

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^3,\ \mathbf{x} \times \mathbf{y} = [\mathbf{x}]_\times \mathbf{y}$$

Definition 2. *The Kronecker product, noted* $\otimes$, *between two matrices* $A = [a_{ij}] \in M_{m,n},\ B \in M_{p,q}$ *is defined by the partitioned matrix*

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix} = [a_{ij}B] \in M_{mp,nq} \tag{100}$$

The Kronecker power notation is defined by

$$A^{\otimes n} = A \otimes A \dots \otimes A$$

Proposition 6. *Let* $A \in M_{m,n},\ X \in M_{n,p}$ *and* $B \in M_{p,q}$. *Then*

$$vec\,(AXB) = \left(B^T \otimes A\right) vec\,(X)$$

Proposition 7. *Let* $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, $A \in M_{p,m}$ *such that each element* a_{ij} *of* A *are a function from* $\mathbb{R}^n$ *to* $\mathbb{R}$. *Suppose that both* f *and* A *are function of the same variable* $\mathbf{x} \in \mathbb{R}^n$. *Then*

$$\frac{dAf}{d\mathbf{x}} = A\frac{df}{d\mathbf{x}} + \left(I_p \otimes \mathbf{f}^T\right) \frac{dvec\left(A^T\right)}{d\mathbf{x}}$$

Proof. Let $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{a}_i^T$ the i-th row of A. We then have

$$\begin{aligned} \frac{dAf}{dx} &= \left[\frac{\partial \mathbf{a}_i^T \mathbf{f}}{\partial x_j}\right] \\ &= \left[\mathbf{a}_i^T \frac{\partial \mathbf{f}}{\partial x_j}\right] + \left[\mathbf{f}^T \frac{\partial \mathbf{a}_i}{\partial x_j}\right] \end{aligned}$$

The first term is simply

$$\left[\mathbf{a}_i^T \frac{\partial \mathbf{f}}{\partial x_j}\right] = A\frac{d\mathbf{f}}{d\mathbf{x}}$$

The second term is given by

$$\begin{aligned}\left[\mathbf{f}^T \frac{\partial \mathbf{a}_i}{\partial x_j}\right] &= \begin{bmatrix}\mathbf{f}^T & & \\ & \ddots & \\ & & \mathbf{f}^T\end{bmatrix}\begin{bmatrix}\frac{\partial \mathbf{a}_1}{\partial x_1} & \cdots & \frac{\partial \mathbf{a}_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial \mathbf{a}_p}{\partial x_1} & \cdots & \frac{\partial \mathbf{a}_p}{\partial x_n}\end{bmatrix} \\ &= \left(I_p \otimes \mathbf{f}^T\right)\left[\frac{\partial \mathbf{a}_i}{\partial x_j}\right]\end{aligned}$$

where $\left[\frac{\partial \mathbf{a}_i}{\partial x_j}\right]$ is $p \times 1$ block-row matrix. By considering the transpose of A, we have the corresponding expression

$$\left[\frac{\partial \mathbf{a}_i}{\partial x_j}\right] = \frac{dVec\left(A^T\right)}{d\mathbf{x}}$$

□

A.2 Derivations and proofs

A.2.1 Proof of Proposition 1

Proof. Based on (5) and (6) with $\mathcal{G} = SE(3)$, we compute from (26)

$$\frac{d}{dt}F_i(\gamma(t)) = \bar{\boldsymbol{e}}_i^T\left(2\Sigma_{\mathbf{e}_i}^{-1}\frac{d\bar{\boldsymbol{e}}_i}{dt} + \frac{d\Sigma_{\mathbf{e}_i}^{-1}}{dt}\bar{\boldsymbol{e}}_i\right) \tag{101}$$

We have from (12)

$$\frac{d\bar{\boldsymbol{e}}_i}{dt}_{|0} = RU_{\bar{\boldsymbol{c}}_i}\boldsymbol{\xi} \tag{102}$$

Similarly,

$$\frac{dR}{dt}_{|0} = R\widehat{\boldsymbol{\omega}} \tag{103}$$

From (25) and (103), we have

$$\frac{d\Sigma_{\mathbf{e}_i}^{-1}}{dt}_{|0} = -\Sigma_{\mathbf{e}_i}^{-1}\frac{d\Sigma_{\mathbf{e}_i}}{dt}_{|0}\Sigma_{\mathbf{e}_i}^{-1} \tag{104}$$

$$\begin{aligned}\frac{d\Sigma_{\mathbf{e}_i}}{dt}_{|0} &= \frac{dR\Omega_{\bar{\boldsymbol{c}}_i}R^T}{dt}_{|0} \\ &= \frac{dR}{dt}_{|0}\Omega_{\bar{\boldsymbol{c}}_i}R^T + R\Omega_{\bar{\boldsymbol{c}}_i}\frac{dR}{dt}^T_{|0} \\ &= R\left(\widehat{\boldsymbol{\omega}}\Omega_{\bar{\boldsymbol{c}}_i} - \Omega_{\bar{\boldsymbol{c}}_i}\widehat{\boldsymbol{\omega}}\right)R^T\end{aligned} \tag{105}$$

Replacing (102)(104)(105) in (101) gives

$$DF_{\mathbf{q}}(T\widehat{\boldsymbol{\xi}}) = \sum_{i=1}^{N}\bar{\boldsymbol{e}}_i^T\Sigma_{\mathbf{e}_i}^{-1}R\left(2U_{\bar{\boldsymbol{c}}_i}\boldsymbol{\xi} + \left(\Omega_{\bar{\boldsymbol{c}}_i}\widehat{\boldsymbol{\omega}} - \widehat{\boldsymbol{\omega}}\Omega_{\bar{\boldsymbol{c}}_i}\right)R^T\Sigma_{\mathbf{e}_i}^{-1}\bar{\boldsymbol{e}}_i\right) \tag{106}$$

□

A.2.2 Proof of Proposition 2

A.2.3 Derivation of H_q^F

By derivating (101) we have

$$\begin{aligned}\frac{d^2}{dt^2}F(\gamma(t)) &= \frac{d\bar{\mathbf{e}}}{dt}^T\left(2\Sigma_{\mathbf{e}}^{-1}\frac{d\bar{\mathbf{e}}}{dt}+\frac{d\Sigma_{\mathbf{e}}^{-1}}{dt}\bar{\mathbf{e}}\right)\\ &\quad+\bar{\mathbf{e}}^T\left(3\frac{d\Sigma_{\mathbf{e}}^{-1}}{dt}\frac{d\bar{\mathbf{e}}}{dt}+2\Sigma_{\mathbf{e}}^{-1}\frac{d^2\bar{\mathbf{e}}}{dt^2}+\frac{d^2\Sigma_{\mathbf{e}}^{-1}}{dt^2}\bar{\mathbf{e}}\right)\end{aligned} \tag{107}$$

In homogeneous coordinates we have

$$\frac{d^2\bar{\mathbf{e}}}{dt^2}=T\widehat{\boldsymbol{\xi}}^2e^{t\widehat{\boldsymbol{\xi}}}\bar{\mathbf{c}} \tag{108}$$

from which we deduce in heterogeneous coordinates

$$\frac{d^2\bar{\mathbf{e}}}{dt^2}_{|0}=R\widehat{\boldsymbol{\omega}}\left(\widehat{\boldsymbol{\omega}}\bar{\mathbf{c}}+\tau\right) \tag{109}$$

We recall (104)

$$\frac{d\Sigma_{\mathbf{e}_i}^{-1}}{dt}=\Sigma_{\mathbf{e}_i}^{-1}R\left(e^{t\widehat{\boldsymbol{\omega}}}\Omega_{\bar{\mathbf{c}}_i}e^{-t\widehat{\boldsymbol{\omega}}}\widehat{\boldsymbol{\omega}}-\widehat{\boldsymbol{\omega}}e^{t\widehat{\boldsymbol{\omega}}}\Omega_{\bar{\mathbf{c}}_i}e^{-t\widehat{\boldsymbol{\omega}}}\right)R^T\Sigma_{\mathbf{e}_i}^{-1} \tag{110}$$

Derivating it once we obtain after simplification,

$$\frac{d^2\Sigma_{\mathbf{e}_i}^{-1}}{dt^2}_{|0}=\Sigma_{\mathbf{e}_i}^{-1}R\left(2\Lambda_{\omega_i}R^T\Sigma_{\mathbf{e}_i}^{-1}R\Lambda_{\omega_i}+2\widehat{\boldsymbol{\omega}}\Omega_{\bar{\mathbf{c}}_i}\widehat{\boldsymbol{\omega}}-\Omega_{\bar{\mathbf{c}}_i}\widehat{\boldsymbol{\omega}}^2-\widehat{\boldsymbol{\omega}}^2\Omega_{\bar{\mathbf{c}}_i}\right)R^T\Sigma_{\mathbf{e}_i}^{-1} \tag{111}$$

The final result is obtained by replacing the intermediate computation in (107)

$$\begin{aligned}\frac{d^2}{dt^2}F(\gamma(0)) &= \bar{\mathbf{e}}_i^T\Sigma_{\mathbf{e}_i}^{-1}R\Big(2\widehat{\boldsymbol{\omega}}\left(\widehat{\boldsymbol{\omega}}\bar{\mathbf{c}}_i+\tau\right)+D_{\omega_i}R^T\Sigma_{\mathbf{e}_i}^{-1}\bar{\mathbf{e}}_i+K_{\omega_i}R\left(2K_{\omega_i}\bar{\mathbf{e}}_i+3U_{\bar{\mathbf{c}}_i}\boldsymbol{\xi}\right)\Big)+\\ &\quad\boldsymbol{\xi}^TU_{\bar{\mathbf{c}}_i}^TR^T\Sigma_{\mathbf{e}_i}^{-1}R\Big(2U_{\bar{\mathbf{c}}_i}\boldsymbol{\xi}+K_{\omega_i}\bar{\mathbf{e}}_i\Big)\\ \Lambda_{\omega_i} &= \Omega_{\bar{\mathbf{c}}_i}\widehat{\boldsymbol{\omega}}\\ K_{\omega_i} &= (\Lambda_{\omega_i}+\Lambda_{\omega_i}^T)R^T\Sigma_{\mathbf{e}_i}^{-1}\\ D_{\omega_i} &= 2\widehat{\boldsymbol{\omega}}\Omega_{\bar{\mathbf{c}}_i}\widehat{\boldsymbol{\omega}}-\Omega_{\bar{\mathbf{c}}_i}\widehat{\boldsymbol{\omega}}^2-\widehat{\boldsymbol{\omega}}^2\Omega_{\bar{\mathbf{c}}_i}\end{aligned} \tag{112}$$

A.2.4 Derivation of $H_{\mathbf{q},\mathbf{z}}^F$

Proof. The hessian matrix $H_{\mathbf{q},\mathbf{z}_i}^{F_i}$ is defined by

$$H_{\mathbf{q},\mathbf{z}_i}^{F_i} = \frac{\partial J_{\mathbf{q}}^{F_i^T}}{\partial\mathbf{z}_i} \tag{113}$$

$$= \left[\frac{\partial DF_{i_{\mathbf{q}}}(TG_k^{SE(3)})}{\partial\mathbf{z}_i}\right] \tag{114}$$

$$= \left[\frac{\partial DF_{i_{\mathbf{q}}}(TG_k^{SE(3)})}{\partial\mathbf{a}_i}\ \frac{\partial DF_{i_{\mathbf{q}}}(TG_k^{SE(3)})}{\partial\mathbf{c}_i}\right] \tag{115}$$

First we have

$$\frac{\partial\bar{\mathbf{e}}}{\partial\mathbf{z}}=\begin{bmatrix}\frac{\partial\bar{\mathbf{e}}}{\partial\mathbf{a}} & \frac{\partial\bar{\mathbf{e}}}{\partial\mathbf{c}}\end{bmatrix}=\begin{bmatrix}-I_3 & R\end{bmatrix}$$

It follows

$$\frac{\partial DF_{i_{\mathbf{q}}}(T\widehat{\boldsymbol{\xi}})}{\partial\mathbf{a}_i} = -2\left(\boldsymbol{\xi}^TU_{\bar{\mathbf{c}}_i}^T+\bar{\mathbf{e}}_i^TK_{\omega_i}^T\right)R^T\Sigma_{\mathbf{e}_i}^{-1} \tag{116}$$

from which we deduce

$$\begin{aligned} H^{F_i}_{\mathbf{q},\mathbf{a}_i} &= \left[\frac{\partial DF_{i_\mathbf{q}}(T\widehat{\boldsymbol{\xi}}_k)}{\partial \mathbf{a}_i}\right] \\ &= -2\left(U^T_{\bar{\mathbf{c}}_i}R^T\Sigma^{-1}_{\mathbf{e}_i} + \left[\bar{\boldsymbol{e}}_i^T K^T_{\omega_i}(\widehat{\boldsymbol{\xi}}_k)R^T\Sigma^{-1}_{\mathbf{e}_i}\right]\right) \\ &= -2\left(U^T_{\bar{\mathbf{c}}_i}R^T\Sigma^{-1}_{\mathbf{e}_i} + \left(I_3\otimes\bar{\boldsymbol{e}}_i^T R^T\Sigma^{-1}_{\mathbf{e}_i}\right)\left[(\Lambda_{\omega_i}+\Lambda^T_{\omega_i})(\widehat{\boldsymbol{\xi}}_k)\right]R^T\Sigma^{-1}_{\mathbf{e}_i}\right) \\ &= -2\left(U^T_{\bar{\mathbf{c}}_i} + \left(I_3\otimes\bar{\boldsymbol{e}}_i^T R^T\Sigma^{-1}_{\mathbf{e}_i}\right)\left(K+K^T\right)\right)R^T\Sigma^{-1}_{\mathbf{e}_i} \end{aligned} \tag{117}$$

with $K = (I_6\otimes\Omega_{\bar{\mathbf{c}}_i})\begin{bmatrix} G_1^{SO(3)} \\ G_2^{SO(3)} \\ G_3^{SO(3)} \\ 0_{9,3}\end{bmatrix}$. For the derivation w.r.t $\bar{\boldsymbol{c}}_i$, we have

$$\frac{\partial DF_{i_\mathbf{q}}(T\widehat{\boldsymbol{\xi}})}{\partial \mathbf{c}_i} = 2\left(A_1+A_2\right)+A_3+A_4 \tag{118}$$

$$A_1 = \bar{\boldsymbol{e}}_i^T\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}\boldsymbol{\xi}}{\partial\bar{\boldsymbol{c}}_i} \tag{119}$$

$$A_2 = \boldsymbol{\xi}^T U^T_{\bar{\boldsymbol{c}}_i}R^T\Sigma^{-1}_{\mathbf{e}_i}\frac{\partial\bar{\boldsymbol{e}}_i}{\partial\bar{\boldsymbol{c}}_i} \tag{120}$$

$$= \boldsymbol{\xi}^T U^T_{\bar{\boldsymbol{c}}_i}R^T\Sigma^{-1}_{\mathbf{e}_i}R \tag{121}$$

$$A_3 = \bar{\boldsymbol{e}}_i^T\left(\Sigma^{-1}_{\mathbf{e}_i}RK_{\omega_i} + \left(\Sigma^{-1}_{\mathbf{e}_i}RK_{\omega_i}\right)^T\right)\frac{\partial\bar{\boldsymbol{e}}_i}{\partial\bar{\boldsymbol{c}}_i} \tag{122}$$

$$= 2\bar{\boldsymbol{e}}_i^T\Sigma^{-1}_{\mathbf{e}_i}R\left(\Lambda_{\omega_i}+\Lambda^T_{\omega_i}\right)R^T\Sigma^{-1}_{\mathbf{e}_i}R \tag{123}$$

$$A_4 = \bar{\boldsymbol{e}}_i^T\left[\frac{\partial(\Sigma^{-1}_{\mathbf{e}_i}RK_{\omega_i})^T}{\partial\bar{\boldsymbol{c}}_{i,j}}\right](I_3\otimes\bar{\boldsymbol{e}}_i) \tag{124}$$

We then have $H^{F_i}_{\mathbf{q},\mathbf{c}_i} = 2\left(\left[A_1(\widehat{\boldsymbol{\xi}}_k)\right]+\left[A_2(\widehat{\boldsymbol{\xi}}_k)\right]\right)+\left[A_3(\widehat{\boldsymbol{\xi}}_k)\right]+\left[A_4(\widehat{\boldsymbol{\xi}}_k)\right]$. The first term is given by

$$\left[A_1(\widehat{\boldsymbol{\xi}}_k)\right] = \left[\bar{\boldsymbol{e}}_i^T\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}\boldsymbol{\xi}_k}{\partial c_{i,1}}\ \bar{\boldsymbol{e}}_i^T\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}\boldsymbol{\xi}_k}{\partial c_{i,2}}\ \bar{\boldsymbol{e}}_i^T\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}\boldsymbol{\xi}_k}{\partial c_{i,3}}\right] \tag{125}$$

$$= \left(I_6\otimes\bar{\boldsymbol{e}}_i^T\right)\left[vec\left(\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,1}}\right)\ vec\left(\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}}{\partial c_{1,2}}\right)\ vec\left(\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,3}}\right)\right] \tag{126}$$

Using the following results

$$\frac{\partial\Sigma_{\mathbf{e}_i}}{\partial c_{i,j}} = R\frac{\partial\Omega_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}}R^T \tag{127}$$

$$\frac{\partial\Omega_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}} = \left(\frac{\partial U_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}}\Sigma_q U^T_{\bar{\boldsymbol{c}}_i} + U_{\bar{\boldsymbol{c}}_i}\Sigma_q\frac{\partial U^T_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}}\right) \tag{128}$$

$$\frac{\partial U^T_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}} = \left[-G_j^{SO(3)}\quad 0_{3,3}\right] \tag{129}$$

We then have

$$\frac{\partial\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}} = \Sigma^{-1}_{\mathbf{e}_i}R\left(\frac{\partial U_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}} - \left(\frac{\partial U_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}}\Sigma_q U^T_{\bar{\boldsymbol{c}}_i} + U_{\bar{\boldsymbol{c}}_i}\Sigma_q\frac{\partial U^T_{\bar{\boldsymbol{c}}_i}}{\partial c_{i,j}}\right)R^T\Sigma^{-1}_{\mathbf{e}_i}RU_{\bar{\boldsymbol{c}}_i}\right) \tag{130}$$

The other terms are given by

$$\left[A_2(\widehat{\boldsymbol{\xi}}_k)\right] = U^T_{\bar{\boldsymbol{c}}_i}R^T\Sigma^{-1}_{\mathbf{e}_i}R \tag{131}$$

$$\left[A_3(\widehat{\boldsymbol{\xi}}_k)\right] = 2\left(I_6\otimes\bar{\boldsymbol{e}}_i^T\Sigma^{-1}_{\mathbf{e}_i}R\right)\left(K+K^T\right)R^T\Sigma^{-1}_{\mathbf{e}_i}R \tag{132}$$

$$\left[A_4(\widehat{\boldsymbol{\xi}}_k)\right] = (I_6\otimes\bar{\boldsymbol{e}}_i)\begin{bmatrix}\frac{\partial(\Sigma^{-1}_{\mathbf{e}_i}RK_{\omega_i}(G_k^{SO(3)}))^T}{\partial\bar{\boldsymbol{c}}_{i,j}} \\ 0_{9,3}\end{bmatrix}(I_3\otimes\bar{\boldsymbol{e}}_i) \tag{133}$$

with

$$\frac{\partial(\Sigma_{\mathbf{e}_i}^{-1} R K_{\omega_i}(G_k^{SO(3)}))^T}{\partial c_{i,j}} = \Sigma_{\mathbf{e}_i}^{-1}\left(\left(R(K_j' + K_j'^T) - \frac{\partial \Sigma_{\mathbf{e}_i}}{\partial c_{i,j}} K_{\omega_i}^T\right) R^T - R K_{\omega_i} \frac{\partial \Sigma_{\mathbf{e}_i}}{\partial c_{i,j}}\right) \Sigma_{\mathbf{e}_i}^{-1} \quad (134)$$

$$K_j' = \left(I_6 \otimes \frac{\partial \Omega_{\bar{\mathbf{c}}_i}}{\partial c_{i,j}}\right) \begin{bmatrix} G_1^{SO(3)} \\ G_2^{SO(3)} \\ G_3^{SO(3)} \\ 0_{9,3} \end{bmatrix} \quad (135)$$

□

A.2.5 Proof of Proposition 3

Proof. First, recall the SBeta pdf f followed by θ :

$$\begin{aligned} f(\theta) &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\left(\theta + \frac{b}{2}\right)^{\alpha-1} \left(\frac{b}{2} - \theta\right)^{\beta-1}}{b^{\alpha+\beta-1}} \\ &= K(\alpha,\beta,b)\left(\theta + \frac{b}{2}\right)^{\alpha-1}\left(\frac{b}{2} - \theta\right)^{\beta-1} \end{aligned} \quad (136)$$

where Γ is the Gamma function. Using the power series of cosinus, we have

$$\begin{aligned} E(\cos\theta) &= \int_{-\frac{b}{2}}^{\frac{b}{2}} cos(\theta) f(\theta) d\theta \\ &= K(\alpha,\beta,b) \int_{-\frac{b}{2}}^{\frac{b}{2}} \sum_{n=0}^{+\infty} \frac{(-1)^n}{2n!} \theta^{2n} \left(\theta + \frac{b}{2}\right)^{\alpha-1} \left(\frac{b}{2} - \theta\right)^{\beta-1} d\theta \end{aligned} \quad (137)$$

One can easily show that the serie-integral inversion holds and gives

$$E(\cos\theta) = K(\alpha,\beta,b) \sum_{n=0}^{+\infty} \frac{(-1)^n}{2n!} \int_{-\frac{b}{2}}^{\frac{b}{2}} \theta^{2n} \left(\theta + \frac{b}{2}\right)^{\alpha-1} \left(\frac{b}{2} - \theta\right)^{\beta-1} d\theta \quad (138)$$

With a simple change of variable, the integral can be express as

$$\begin{aligned} \int_{-\frac{b}{2}}^{\frac{b}{2}} \theta^{2n} \left(\theta + \frac{b}{2}\right)^{\alpha-1} \left(\frac{b}{2} - \theta\right)^{\beta-1} d\theta &= \frac{b^{\alpha+\beta+2n-1}}{4^n} \int_0^1 \theta^{\alpha-1} (1-2\theta)^{2n} (1-\theta)^{\beta-1} d\theta \\ &= \frac{b^{\alpha+\beta+2n-1}}{4^n} \frac{{}_2F_1(-2n, \alpha, \alpha+\beta; 2)}{K(\alpha,\beta,b) b^{\alpha+\beta-1}} \\ &= \left(\frac{b}{2}\right)^{2n} {}_2F_1(-2n, \alpha, \alpha+\beta; 2) \end{aligned} \quad (139)$$

where we used the integral form of the Hypergeometric function ${}_2F_1$.

The result for $E(\sin\theta)$ and $E(\cos^2\theta)$ are obtained similarly using the power series

$$\begin{aligned} \sin\theta &= \sum_{n=0}^{+\infty} \frac{(-1)^{2n+1}}{2n+1!} \theta^{2n+1} \\ \cos^2\theta &= 1 + \sum_{n=1}^{+\infty} \frac{(-1)^n}{2n!} 2^{2n-1} \theta^{2n} \end{aligned}$$

□

A.2.6 Proof of Proposition 4

Proof. First define the Pochhammer symbol as a generalization of the usual factorial. Let $a \in \mathbb{R}$ and $n \in \mathbb{N}^*$:

$$(a)_n = a(a+1)\dots(a+n-1) \tag{140}$$

and set $(a)_0$ otherwise. The hypergeometric function ${}_2F_1$ depends on 3 parameters and is defined for $m \in \mathbb{N}$ and $b, c \in \mathbb{R}^+$ as :

$${}_2F_1(-m, b, c; z) = \sum_{n=0}^{m} (-1)^n \binom{m}{n} \frac{(b)_n}{(c)_n} z^n \tag{141}$$

We aim at controlling the reminder for the series :

$$R_N\left(E(\cos\theta)\right) = \sum_{n=N}^{+\infty} \frac{(-1)^n}{2n!} \left(\frac{b}{2}\right)^{2n} {}_2F_1(-2n, \alpha, \alpha+\beta; 2) \tag{142}$$

Notice that

$${}_2F_1(-2n, \alpha, \alpha+\beta; 2) = \sum_{k=0}^{2n} (-1)^k \binom{2n}{k} \frac{(\alpha)_k}{(\alpha+\beta)_k} 2^k \tag{143}$$

Since $\frac{(\alpha)_k}{(\alpha+\beta)_k} \leq 1$, we get $|{}_2F_1(-2n, \alpha, \alpha+\beta; 2)| \leq \sum_{k=0}^{2n} \binom{2n}{k} 2^k = 3^{2n}$ so that

$$\begin{aligned}
\left|R_N\left(E(\cos\theta)\right)\right| &\leq \sum_{n=N}^{+\infty} \frac{1}{(2n)!} \left(\frac{3b}{2}\right)^{2n} = \sum_{k=0}^{+\infty} \frac{1}{(2(k+N))!} \left(\frac{3b}{2}\right)^{2(k+N)} && (144)\\
&\leq \frac{1}{(2N)!} \left(\frac{3b}{2}\right)^{2N} \sum_{k=0}^{+\infty} \frac{1}{(2n)!} \left(\frac{3b}{2}\right)^{2n} && (145)\\
&\leq \frac{1}{(2N)!} \left(\frac{3b}{2}\right)^{2N} \cosh\left(\frac{3b}{2}\right) && (146)
\end{aligned}$$

The proof for the other bounds is obtained following similar computation. □

A.2.7 Proof of Proposition 5

Proof. First, let consider a centered standard normal random value $Z \sim \mathcal{N}(0,1)$. We then have $\phi = \sigma_\phi Z + \bar{\phi}$ and

$$\begin{aligned}
E(e^{i\phi}) &= E(e^{i(\sigma_\phi Z + \bar{\phi})}) && (147)\\
&= E(e^{i\bar{\phi}}) E(e^{i\sigma_\phi Z})\\
&= \frac{E(e^{i\bar{\phi}})}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{i\sigma_\phi z} e^{-\frac{z^2}{2}} dz\\
&= \frac{E(e^{i\bar{\phi}})}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-\frac{(z-i\sigma_\phi)^2}{2}} e^{-\frac{\sigma_\phi^2}{2}} dz\\
&= \frac{E(e^{i\bar{\phi}})}{\sqrt{2\pi}} e^{-\frac{\sigma_\phi^2}{2}} \int_{-\infty-i\sigma_\phi}^{+\infty-i\sigma_\phi} e^{-\frac{z^2}{2}} dz\\
&= E(e^{i\bar{\phi}}) e^{-\frac{\sigma_\phi^2}{2}} && (148)
\end{aligned}$$

Taking the real and imaginary part of (148) gives $E(\cos\theta)$ and $E(\sin\theta)$. The other results are deduced from classic trigonometric relations. □

A.3 Expressions of $A_{i,j,k,l}$ in Section 5.3

The non-null matrices A_{ijkl} defined in (65) are given by

$$
\begin{aligned}
A_{2,2,0,0} &= a_0 a_0^T & A_{0,0,2,2} &= a_1 a_1^T \\
A_{0,2,2,0} &= a_2 a_2^T & A_{2,0,0,2} &= a_3 a_3^T \\
A_{2,1,0,1} &= a_0 a_3^T + a_3 a_0^T & A_{1,2,1,0} &= a_0 a_2^T + a_2 a_0^T \\
A_{0,1,2,1} &= a_1 a_2^T + a_2 a_1^T & A_{1,0,1,2} &= a_1 a_3^T + a_3 a_1^T \\
A_{1,1,1,1} &= a_0 a_1^T + a_1 a_0^T + a_2 a_3^T + a_3 a_2^T & A_{1,2,0,0} &= a_0 b_0^T + b_0 a_0^T \\
A_{0,2,1,0} &= a_2 b_0^T + b_0 a_2^T & A_{0,0,1,2} &= a_1 b_1^T + b_1 a_1^T \\
A_{1,0,0,2} &= a_3 b_1^T + b_1 a_3^T & A_{0,1,1,1} &= a_1 b_0^T + b_0 a_1^T + a_2 b_1^T + b_1 a_2^T \\
A_{1,1,0,1} &= a_0 b_1^T + b_1 a_0^T + a_3 b_0^T + b_0 a_3^T & A_{0,2,0,0} &= b_0 b_0^T \\
A_{0,0,0,2} &= b_1 b_1^T & A_{0,1,0,1} &= b_0 b_1^T + b_1 b_0^T
\end{aligned}
$$

with $a_0, a_1, a_2, a_3, b_0, b_1$ defined in (63)