PointNetKL: Deep Inference for GICP Covariance Estimation in Bathymetric SLAM

Ignacio Torroba, Christopher Iliffe Sprague, Nils Bore, and John Folkesson

***Abstract*—Registration methods for point clouds have become a key component of many SLAM systems on autonomous vehicles. However, an accurate estimate of the uncertainty of such registration is a key requirement to a consistent fusion of this kind of measurements in a SLAM filter. This estimate, which is normally given as a covariance in the transformation computed between point cloud reference frames, has been modelled following different approaches, among which the most accurate is considered to be the Monte Carlo method. However, a Monte Carlo approximation is cumbersome to use inside a time-critical application such as online SLAM. Efforts have been made to estimate this covariance via machine learning using carefully designed features to abstract the raw point clouds [1]. However, the performance of this approach is sensitive to the features chosen. We argue that it is possible to learn the features along with the covariance by working with the raw data and thus we propose a new approach based on PointNet [2]. In this work, we train this network using the KL divergence between the learned uncertainty distribution and one computed by the Monte Carlo method as the loss. We test the performance of the general model presented applying it to our target use-case of SLAM with an autonomous underwater vehicle (AUV) restricted to the 2-dimensional registration of 3D bathymetric point clouds.**

***Index Terms*—SLAM, novel deep learning methods, marine robotics, simultaneous localization and mapping, robot learning, unmanned underwater vehicles.**

I. Introduction

OVER the last few years, sensors capable of providing dense representations of 3D environments as raw data, such as RGB-D cameras, LiDAR or multibeam sonar, have become popular in the SLAM community. These sensors provide accurate models of the geometry of a scene in the form of sets of points, which allows for a dense representation of maps easy to visualise and render. The wide use of these sensors has given rise to the need for point cloud registration algorithms in the robotics and computer vision fields. For instance, the core of well-established SLAM frameworks for ground and underwater robots, such as [3] and [4], relies upon point cloud registration methods, such as the iterative closest point (ICP) [5], to provide measurement updates from dense 3D raw input.

Manuscript received December 9, 2019; accepted March 25, 2020. Date of publication April 20, 2020; date of current version May 6, 2020. This letter was recommended for publication by Associate Editor Prof. Giorgio Grisetti and Editor Prof. Sven Behnke upon evaluation of the reviewers' comments. *(I. Torroba and C. I. Sprague contributed equally to this work.) (Corresponding author: Christopher Iliffe Sprague.)*

The authors are with the Swedish Maritime Robotics Centre (SMaRC) and the Division of Robotics, Perception and Learning, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden (e-mail: torroba@kth.se; sprague@kth.se; nbore@kth.se; johnf@kth.se).

Digital Object Identifier 10.1109/LRA.2020.2988180

While well rooted in indoor and outdoor robotics, SLAM has not yet gained widespread use for AUVs. However, the need for SLAM is greater underwater, as navigation is extremely challenging. Over long distances, sonar is the only viable sensor to correct dead-reckoning estimates. Of the various types of sonar, multibeam echo sounders (MBES) provide the most suitable type of raw data for SLAM methods. This data is essentially a point cloud sampled from the bathymetric surface, and applying registration methods to these measurements is a well-studied problem [6]. However, when fusing the output of the registration into the Bayesian estimate of the AUV state, the uncertainty of the transform must be modelled, since it represents the weight of the measurement. Despite its importance in every SLAM domain, few works have addressed the problem of estimating this uncertainty accurately and efficiently, usually in the form of a covariance matrix. We argue that these two requirements are vital in our setting and need to be addressed simultaneously and onboard an AUV, with limited computational resources. While there have been recent attempts to derive the covariance of the registration process analytically, these approaches were limited either by the reliability of the estimation or its complexity. The latest and most successful techniques, however, have aimed at learning such a model. [1] is an example of this approach applied to constraints created from point clouds registrations with ICP. However, although successful, it is limited by the need to reduce the input point clouds to hand-crafted feature descriptors, whose design can be a non-trivial, task-dependent challenge. Hence, we motivate our work with the goal to circumvent the need to design such descriptors, and to instead learn the features directly from the underlying raw data. We accomplish this with the use of the relatively recent artificial neural network (ANN) architecture, PointNet [2]. We combine PointNet and a parameterisation of a Cholesky decomposition of the covariance of the objective function into a single model, named PointNetKL, which is invariant to permutations of its input. In this work, we use this architecture to estimate the Generalised-ICP (GICP) [7] uncertainty distributions directly from raw data and test the learned model in a real underwater SLAM scenario. Our contributions are listed as follows:

- We present PointNetKL, a new learning architecture built upon PointNet, for learning multivariate probability distributions from unordered sets of V-dimensional points.
- We apply this general architecture to the restricted case of learning $2D$ covariances from the constrained GICP registration of real $3D$ bathymetric point clouds from several underwater environments.

- We assess the performance and generalisation of our architecture in both the regression and SLAM tasks.

II. Formal Motivation

A point cloud registration algorithm can be defined as an optimisation problem aimed at minimising the distance between corresponding points from two partially overlapping sets of points, S_i and S_j, with rigidly attached reference frames T_i and T_j. In the general case, a point cloud $P_i = \{S_i, T_i\}$ consists of an unordered set of V-dimensional points S_i while a reference frame $T_i \in SE(3)$ represents a 6-DOF pose. According to this, an unbiased registration process can be modelled as a function h of the form

$$T_{ij} = T_i^{-1} T_j = h(P_i, P_j), \tag{1}$$

where $h(P_i, P_j)$ is the true relative rigid transformation between the two frames. This transformation is estimated to minimise the alignment error to S_j when applied to S_i. However, it is well known that due to the fact that point cloud registration locally optimises over a non-convex function, its solution is sensitive to convergence towards local minima and so its performance relies crucially on the initial relative transformation between the point clouds, computed as $\hat{T}_i^{-1}\hat{T}_j$. Where the hat denotes the current best estimate of the pose.

The uncertainty in the estimate of the true transformation can be represented within a SLAM framework as follows. Given an autonomous mobile robot whose state over time is given by $x_i \in \mathbb{R}^N$ and with a dead reckoning (DR) system that follows a transition equation 2, we can model a measurement update based on point cloud registration through Eq. 3, where $z_{ij} \in \mathbb{R}^M$ with $M \leq N$.

$$x_i = g(x_{i-1}, u_i) + \epsilon_i \tag{2}$$

$$z_{ij} = h(P_i, P_j) + \delta_{ij}. \tag{3}$$

ϵ_i and δ_{ij} model the error in the DR and in the registration respectively. Approximating the noise in the DR and measurement models as white Gaussian with covariances $R \in \mathbb{R}^{N\times N}$ and $Q \in \mathbb{R}^{M\times M}$, respectively, both x_i and z_{ij} will follow probability distributions given by

$$x_i \sim \mathcal{N}(g(x_{i-1}, u_i), R_i) \tag{4}$$

$$z_{ij} \sim \mathcal{N}(h(P_i, P_j), Q_{ij}) \tag{5}$$

With z_{ij} being measurements of the transform between point clouds from the registration algorithm and the true transform $h(P_i, P_j)$ being given by the T_i and T_j. The Bayesian estimate that results from this model is generally analytically intractable due to the nonlinear terms, but iterative maximum likelihood estimates (MLE) are possible. Such an iterative SLAM estimate requires a reliable approximation of the probability distribution of the registration error $\delta_{ij} \sim \mathcal{N}(0, Q_{ij})$. This is because Q_{ij}^{-1} represents the weight or "certainty" of the measurement z_{ij} when being added to the state estimate, which is a critical step in any state-of-the-art SLAM solution. However, this covariance is not available and research has focused on deriving both analytical and data-driven methods to estimate it. Over the next section, we revisit the most relevant of these methods and the previous work upon which our approach builds.

III. Related Work

As introduced above, the need of SLAM systems for a reliable approximation of the error distribution of point cloud registration processes has motivated a prolific body of work in this topic. Among the numerous existing registration techniques, the ICP algorithm and its variants, such as GICP, are the most widely used in the SLAM community. The methods to estimate the uncertainty of the solution of ICP-based techniques are traditionally divided into two categories: analytic and data-driven. Analytical solutions based on the Hessian of the objective function, such as [8], have yielded successful results on the $2D$ case thanks to its capacity to model the sensor noise. However, [9] shows that its extension to 3D contexts results in overly optimistic results, which do not reflect the original distribution. Another set of approaches, such as [10], consists in developing estimation models for specific sensors. Although more accurate, this kind of method suffers in its inability to generalise to different sensors.

There is a large body of work on non-parametric approaches to estimating probability distributions, e.g. [1], [11], [12]. In [11], Iversen *et al.* apply a Monte-Carlo (MC) approach to estimate the value of the covariance of ICP on synthetic depth images. Although accurate, this approach cannot be applied online due to its computation time. In [12], a general non-parametric noise model was proposed; however, its performance is adversely affected by scaling with higher-dimensional features. Landry *et al.* introduce, in [1], the use of the CELLO architecture for ICP covariance estimation. This method proves to be a reliable estimation framework but it is limited by the need to create hand-crafted features from the raw 3D point clouds. As argued in [13], designing these features for a given task is still an open issue and so the success of CELLO is highly dependant on the chosen features.

Going beyond hand-crafted features, a number of works have addressed the problem of learning feature representations from unordered sets such as point clouds; however, these approaches do not capture spatial structure. As a result, there have been a number of point cloud representations developed through multi-view [14] and volumetric approaches [15]. Unfortunately, multi-view representations are not amenable to open-scene understanding, and volumetric representations are limited by data resolution and the computational cost of convolution.

The work in [16] presents an inference framework based on a deep ANN, DICE, that can be trained on raw images. This work, to the best of our knowledge, represents the first instance of the use of a ANN to infer the uncertainty of a measurement model in a similar approach to ours. However, their network is limited to camera input and they require ground truth measurements to construct the training set, a commodity often hard to afford.

To counteract the need to preprocess the input point clouds as in [1] while being able to apply deep learning techniques as in [16], we have turned to the seminal work PointNet [2] for our method. When choosing between learning architectures, e.g. [17], [18], we choose PointNet for its simplicity. This being the

first use of ANNs for bathymetric GICP covariance estimation, our results can serve as a first baseline.

PointNet employs a relatively simple ANN architecture, that achieves striking performance in both classification and segmentation tasks upon raw point cloud data. It relies on the principle of composing an input-invariant function through the composition of symmetric functions, producing an input-invariant feature vector for a given point cloud.

While the PointNet architecture was originally intended for classification and segmentation, the internally generated feature vectors have been used for other purposes, such as point cloud registration [19] and computation of point cloud saliency maps [20]. Similarly to these works, we seek to employ PointNet for a different purpose, namely for the estimation of multivariate probability distributions.

IV. Approach

Our goal is to estimate, for each pair of overlapping bathymetric point clouds P_i, P_j, a covariance Q_{ij} that is as close as possible to modelling the actual uncertainty of their GICP registration in underwater SLAM, given by $\delta_{ij} \sim \mathcal{N}(0, Q_{ij})$. Learning these covariances entails the use of datasets with large amounts of overlapping point clouds with accurate associated positions, such as the one used in [1]. However, an equivalent dataset does not exist in the underwater robotics literature, due to the nature of bathymetric surveys with a MBES pointing downwards, where the consecutive MBES pings within swaths do not contain overlap.

In order to overcome this problem, we look into sources of uncertainty in the registration of bathymetric point clouds. In general Q_{ij} is a function of the amount of sensor noise, the statistics over the initial starting point for the iterative MLE, and the features in the overlapping sections of the two point clouds. If we assume that S_i has uniform terrain characteristics, i.e. there is not a small rocky corner on an otherwise flat point cloud, we can expect that Q_i does not vary much with the specific region of overlap as long as that region is above a certain percentage of the whole point cloud. This allows us to attribute an intrinsic covariance to each point cloud independent of the actual overlap, i.e. Q_{ij} becomes Q_i (which is also approximately Q_j). The validity of this is based on the fact that it is possible to aggregate raw sonar data on fairly uniform point clouds with the submaps approach used in [21], for example.

A. Learning Architecture

In this section we present a general approach to the problem of learning our target GICP covariance Q_i from a set of points S_i. Formally, we consider the problem of learning a function $\pi_\theta(S_i)$, parameterised by a set of learnable parameters θ, that maps a point cloud of U points $P_i = \{S_i \in \mathbb{R}^{U\times V}, T_i \in \mathbb{R}^M\}$ to a multivariate Gaussian probability distribution $\mathcal{N}(\mu_i, \Sigma_i)$: $\mu_i \in \mathbb{R}^M, \Sigma_i \in \mathbb{R}^{M\times M}$, where Σ_i is strictly positive-definite. We solve this problem by optimising π_θ's parameters θ to regress a dataset of the form

$$D = \{(S_0, \mu_0, \Sigma_0), \ldots, (S_K, \mu_K, \Sigma_K)\}, \tag{6}$$

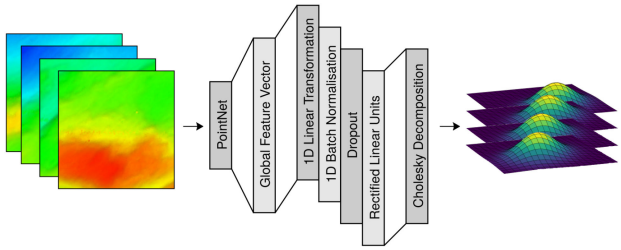


Fig. 1. Depiction of the architecture of PointNetKL. From left to right: a submap is fed into PointNetKL to produce a global feature vector that is then fed into a multilayer perceptron to produce the parameters of a Cholesky decomposition and construct a positive-definite covariance matrix.

consisting of K point clouds and their associated distributions.

To tackle this problem we consider the PointNet architecture [2] in order to work directly with the raw point clouds. We denote PointNet as function $\phi(S_i) : \mathbb{R}^{U\times V} \mapsto \mathbb{R}^Z$ that maps S_i to a Z-dimensional vector-descriptor $\zeta_i \in \mathbb{R}^Z$, describing the features of S_i. Using ζ_i we seek to learn a further mapping to the distribution $\mathcal{N}(\mu_i, \Sigma_i)$. Given the invariance of ζ_i, we postulate that the further abstraction to defining a probability distribution can be achieved by a simple MLP, denoted hereafter as $\psi(\zeta_i)$.

In order to define our target distribution δ_i, we task ψ to output a covariance matrix $\Sigma_i = Q_i$ and set μ_i equal to the null vector. We describe this model collectively as

$$\pi_\theta(S_i) = \psi\left(\phi\left(S_i\right)\right) : \mathbb{R}^{U\times V} \mapsto \mathbb{R}^{M\times M}, \tag{7}$$

where θ is the collective set of learnable parameters. Hereafter, we denote π_θ as π for brevity.

We leave the architecture of ϕ as originally described in [2], removing the segmentation and classification modules. For the hidden model of ψ we consider a fully connected feed-forward architecture of arbitrarily many layers and nodes per layer. For each layer we sequentially apply the following standard machine learning operations: linear transformation, $1D$ batch normalisation, dropout, and rectified linear units, as indicated in Fig. 1. This characterisation of ψ produces outputs in $\mathbb{R}_{\geq 0}$, which are transformed into desired ranges in the following section.

B. Covariance Matrix Composition

In order to map the outputs of ψ to a valid estimation of Σ_i we must enforce positive-definiteness. Following [16], we task ψ to produce the $(M^2 - M)/2 + M$ parameters of a Cholesky composition of the form

$$\Sigma_i = L(l_i)D(d_i)L(l_i)^\intercal : l_i \in \mathbb{R}^{(M^2-M)/2}, d_i \in \mathbb{R}^M_{>0}, \tag{8}$$

where $L(l_i)$ is a lower unitriangular matrix, $D(d_i)$ is a diagonal matrix with strictly positive values, and $[l_i, d_i]$ are the parameters to be produced by ψ. It is important to note that the strict positiveness of $D(d_i)$'s elements enforces the uniqueness of the decomposition and thus that of the probability distribution being estimated. Using a linear transformation layer, we map the penultimate outputs of ψ to $(M^2 - M)/2 + M$ values describing the elements of l_i and d_i. To enforce the positivity of d_i, we simply apply the exponential function such that the

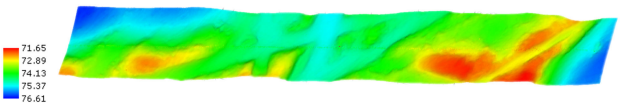


Fig. 2. 500 × 40 meters (approx.) sample of a swath of bathymetry from the Baltic dataset.

decomposition becomes $\Sigma_i = L(l_i)D(\exp(d_i))L(l_i)^\intercal$. We then use Σ_i to fully characterise δ_i.

C. Loss Function

To train π, we must compare its predicted distributions with the true ones in order to compute its loss. For this purpose we use the Kullback-Leibler (KL) divergence

$$D_{KL}\left(\mathcal{N}||\mathcal{N}_\pi\right) = \frac{1}{2}\Bigg(\mathrm{tr}\left(\Sigma_\pi^{-1}\Sigma\right) + (\mu_\pi - \mu)^\intercal \Sigma_\pi^{-1} \times (\mu_\pi - \mu) - M + \ln\left(\frac{\det(\Sigma_\pi)}{\det(\Sigma)}\right)\Bigg), \quad (9)$$

where $\mathrm{tr}(\cdot)$ and $\det(\cdot)$ are the trace and determinant matrix operations, respectively. This gives us the distance between the distribution $\mathcal{N}(\mu_\pi, \Sigma_\pi)$ predicted by π and the target distribution $\mathcal{N}(\mu, \Sigma)$. We optimise π to minimise D_{KL}, hence we coin its name, PointNetKL. As explained in IV-A, in our specific application $\mu = 0$ and so the variable μ will be omitted for brevity from here on.

D. Generation of Training Data

The datasets necessary to train, test and validate our ANN have been generated following an approach similar to [11]. A Monte Carlo approximation has been computed for every point cloud S_i as follows. Given a 3D point cloud $P_i = \{S_i, T_i\}$ and a distribution $O \sim \mathcal{N}(0, \Sigma_{sample})$, we generate a second point cloud P_j by perturbing P_i with a relative rigid transform T_j drawn from O. After the perturbation, Gaussian noise is applied to S_j and the resulting point clouds are registered using GICP. Given the fact that P_j is a perturbation of P_i, the error of the GICP registration can be computed as the distance between the obtained transformation $\hat{T}_j$ and the perturbation originally applied T_j, as in Eq. 10, [22]. This error is then used to calculate the covariance of the distribution δ_i for each point cloud following Eq. 11.

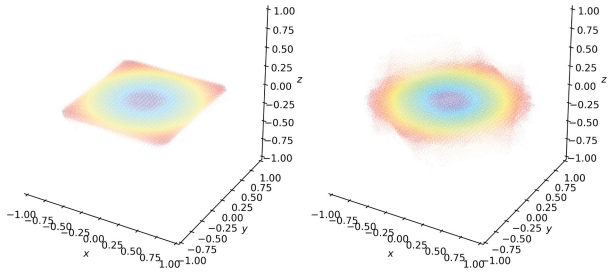
$$e_l = log(exp(\hat{T}_j)^{-1} T_j) \quad (10)$$

$$Q_i = \frac{1}{(L-1)} \sum_{l=1}^{L} e_l e_l^T \quad (11)$$

where L is the total number of MC iterations per point cloud.

V. Experiments

In the remainder of this paper we apply the presented general approach for learning Gaussian distributions from unordered sets of points to the specific problem of bathymetric graph SLAM with GICP registration as introduced in [21]. The reason to focus on GICP as opposed to ICP is that this method works better on the kind of bathymetric point clouds produced from surveys of unstructured seabed, as discussed in [6].



(a) Baltic (b) Bornö 1

Fig. 3. Depiction of the zero-meaned, normalised, and voxelised submaps of two datasets. The dispersion of the points around the unit sphere indicate diversity of the bathymetry.

The nomenclature used this far can be instantiated to this specific context as follows. A point cloud P_i consists now of a bathymetric submap in the form of a point cloud $S_i \in R^{U\times 3}$ and the estimates of the AUV pose while collecting the submap are $T_i = exp(x_i) \in \mathbb{R}^6$. In the case of the GICP registration, the vehicles used to collect the data provided a good direct measurement of the full orientation of the platform and the depth underwater. Due to this, the dimension of the measurement model z_i in Eq. 12 has been reduced to $m = 2$, since it is only the x and y coordinates that contain uncertainty. Consequently, the GICP registration is constrained during the training data generation to the dimensions x, y and therefore the covariances of δ_i and of the prior O become $\mathbb{R}^2$ matrices.

A. The Training Datasets

To the best of our knowledge, no dataset like the one in Eq. 6 with bathymetric submaps exists and therefore a new one has been created. When designing such a dataset several criteria must be fulfilled for the training to be successful and having generalisation of the results in mind: i) The success of GICP will be, to a certain degree, linked to the features in the submap being registered. In simple cases this can be easily interpreted by looking at the resulting covariance. Intuitively, on a perfectly flat submap, an elongated feature along the y direction will ease the registration perpendicular to that axis, yielding low values of the covariance for the x axis. Equivalently, the registration along y will result in a bigger uncertainty since the submaps can slide along the feature. Thus, given that π is learning a mapping from geometric features to covariance values, it is important that the dataset created contains enough variation in the bathymetry. ii) It is not possible to collect and train on enough data for π to be able to generalise successfully to the whole sea floor worldwide. However, [23] proved that pieces of seabed can be successfully modelled through Gaussian processes. Based on this it can be assumed that with a large and varied enough dataset, our model should be able to generalise to natural seabed environments never seen before. This would help to circumvent the fact that this kind of data is very scarce and difficult to obtain in comparison to image or point cloud datasets. iii) Different

TABLE I
DATASETS CHARACTERISTICS

Area	Vehicle	Size (km^2)	Submaps	$\sigma_z \in [0,1]$
Baltic	Surveyor	504.8	7543	0.067827
Shetland	Surveyor	21.3	301	0.066160
Bornö 1	Hugin	74.6	940	0.189505
Antarctica 7	Hugin	88.5	296	0.246802

vehicles are used to collect the data as to ease generalisation. iv) Ground truth (GT) is not available.

In order to ensure a complete distribution of geometric features within the dataset, we have analysed the spatial distribution of the points within the dataset used, with special attention to the dispersion in the z axis, given by the standard deviation σ_z. Fig. 3 shows a canonical representation of the zero-meaned, normalised, and voxelized submaps for two of the datasets used, whose characteristics are given in Table I.

With the view on the criteria exposed above, four bathymetric surveys have been included in the final dataset. They have been collected in four different environments, namely: the south-east Baltic sea, the Shetland Isles, Gullmarsfjorden near Bornö, and underneath the Thwaites glacier in Antarctica. A sample of bathymetry from the Baltic dataset can be seen in Fig. 2. For the data collection, two state-of-the-art vehicles have been used. A remotely operated vehicle (ROV) Surveyor Interceptor and a Kongsberg Hugin AUV with an acoustic beacon Kongsberg cNode Maxiboth deployed in the survey area. Both vehicles were equipped with a MBES EM2040. With these four bathymetric surveys, the training datasets of the form given in Eq. 6 have been generated as explained in IV-D, with $K = 9080$ submaps and $L = 3000$ MC iterations per submap. Following the reasoning in [1], we have set $\Sigma_{sample} = aI$ (where $I \in \mathbb{R}^2$) with $a = 9$ in order to model a reasonable underwater SLAM scenario.

B. Training Implementation

In this work, we leave the internal architecture of PointNet ϕ as it is in [2], outputting the vector-descriptor $\zeta \in \mathbb{R}^{1024}$. For the MLP, mapping ζ to the estimation of GICP covariance Q, we consider an architecture of 4 hidden layers, each having 1000 nodes, using the sequential operations described at the end of Section IV-A. A depiction of the final ANN can be seen in Fig. 1.

To learn the mapping of submaps S_i to covariances Q_i, we optimise the parameters of π to regress the dataset D under the cost function D_{KL} given in Eq. (9). We optimise these parameters with stochastic gradient descent (SGD), using the AMSGrad variant of the adaptive optimiser, Adam [24]. In order to improve the generalisation of π to different environments, we employ dropout [25] and weight decay [26].

For each training episode we sample a random subset (with replacement) of both the training and validation datasets, using the former for an optimisation iteration and the latter to evaluate the generalisation error in order to employ early-stopping [27]. We employ random subset sampling in conjunction with SGD in order to speed up training, as obtaining a gradient over the whole dataset described in Table I is intractable.

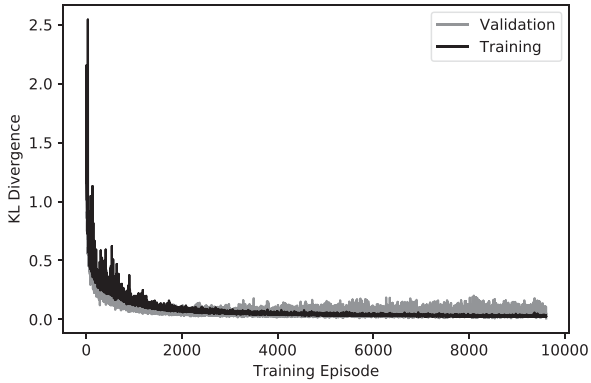


Fig. 4. Evolution of π's KL divergence loss on the training and validation sets. Note the unity of the training and validation curves over episodes, indicating generalisation. The jaggedness of the lines is a result of the stochasticity of the gradient descent, due to random subset sampling and dropout.

The implemented training hyperparameters are:
1) learning rate $= 1 \times 10^{-4}$,
2) $L2$ weight decay penalty $= 1 \times 10^{-4}$,
3) dropout probability $= 40\%$,
4) batch size $= 500$,
5) validation set proportion $= 20\%$,
6) early stopping patience[1] $= 20$.

Before the submaps are fed to network, they need to be pre-processed. Each set S_i is translated to be zero-mean, then normalised to a sphere by the largest magnitude point therein, and finally voxelised to obtain a uniform density grid sampling. This ensures that the raw density of each set S_i does not affect the underlying relation that π seeks to learn. Note, we use voxelisation here merely as a means to downsample the point clouds to lessen memory requirements. In principle, any downsampling method (not necessarily ordered) or none at all could be used.

C. Testing of the Covariances in Underwater SLAM

The validity of the covariances predicted by the network has been tested in the PoseSLAM framework in Eq. 12.

$$\{x_i^*\} = \arg\min_{\hat{x}} \sum_{i}^{N_{DR}} ||g(\hat{x}_{i-1}, u_i) - \hat{x}_i||^2_{R_i} + \sum_{\{i,j\}}^{N_{LC}} ||\hat{T}^{-1}(\hat{x}_i)T(\hat{x}_j) - z_{ij}||^2_{Q_i} \quad (12)$$

Where N_{DR} and N_{LC} are the number of dead reckoning and loop closure (LC) constraints, respectively. Q_i models the weight of each LC edge added to the graph as a result of a successful GICP registration of overlapping submaps P_i, P_j and as such it plays an important role in the optimisation.

For the tests, two underwater surveys outside the training dataset of the network have been used, named Bornö 8 and Thwaites 11. For the test on each scenario, a bathymetric pose graph is created and optimised similarly to [21]. The graph optimisations have been run with three different sets of covariances:

[1]Number of iterations to wait after last validation loss improved.

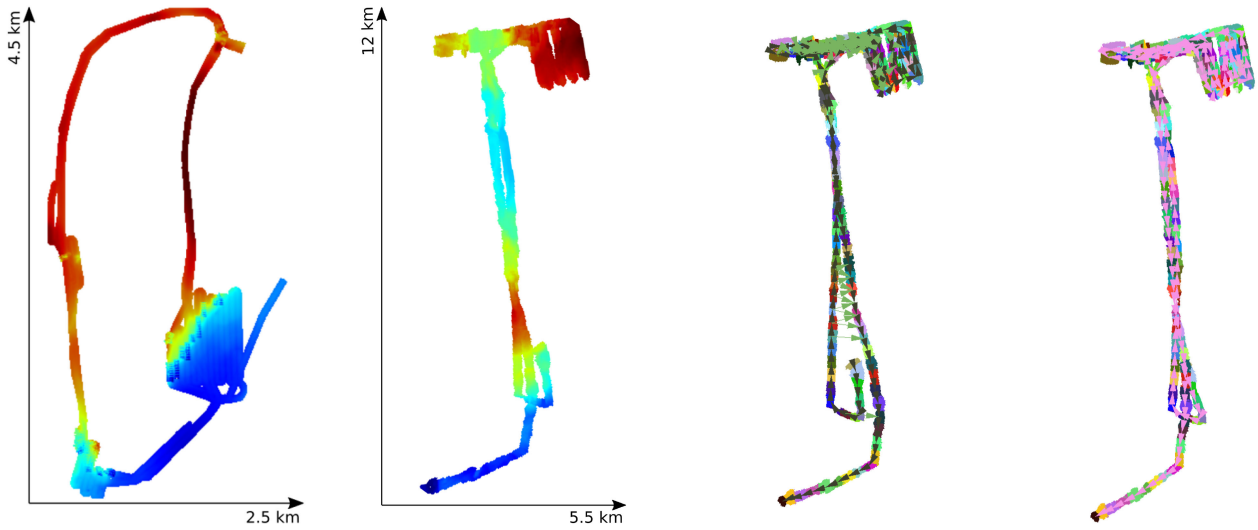


Fig. 5. Left to right: Bornö 8 and Thwaites 11 surveys. Example of corrupted graph for the later ($RMSE_{xyz}$ 301.1 m) and its MC solution ($RMSE_{xyz}$ 222.18 m).

TABLE II
KL DIVERGENCE LOSS OF POINTNETKL π ON THE TRAINING AND VALIDATION AND TESTING SETS

Training	Validation	Bornö 8	Thwaites 11
0.0184	0.0394	1.0578	1.9899

TABLE III
AVERAGE RUNTIME OF THE GENERATION METHODS

	MC	PointNetKL
Runtime (s) $mean \pm std\_dev$	14.71 $\pm$ 5.08	0.13 $\pm$ 0.02

Algorithm 1: Corrupted Pose Graph Construction.

1 **Function** `Create_graph` ($\boldsymbol{S_N}$):
2 $\quad \boldsymbol{G} \leftarrow \{\emptyset\}$
3 $\quad \boldsymbol{S_{graph}} \leftarrow \{\emptyset\}$
4 $\quad$ **for** S_i *in* $\boldsymbol{S_N}$ **do**
5 $\quad\quad \boldsymbol{S_{LC}} \leftarrow \{\emptyset\}$
6 $\quad\quad \boldsymbol{G} \leftarrow \text{AddDRedge}(S_i, S_{i-1})$
7 $\quad\quad \boldsymbol{S_{LC}} \leftarrow \text{DetectLC}(S_i, \boldsymbol{S_{graph}}, coverage)$
8 $\quad\quad$ **if** $\boldsymbol{S_{LC}} \neq \{\emptyset\}$ **then**
9 $\quad\quad\quad \boldsymbol{S_j} \leftarrow \text{Perturb}(S_i, O)$
10 $\quad\quad\quad \boldsymbol{z_j} \leftarrow \text{GICP}(\boldsymbol{S_{LC}}, S_j)$
11 $\quad\quad\quad \boldsymbol{S_i'} \leftarrow \text{Correct}(S_j, z_j)$
12 $\quad\quad\quad \boldsymbol{G} \leftarrow \text{AddLCedge}(S_i', \boldsymbol{S_{LC}})$
13 $\quad\quad\quad \boldsymbol{S_{graph}} \leftarrow S_i'$
14 $\quad\quad$ **else**
15 $\quad\quad\quad \boldsymbol{S_{graph}} \leftarrow S_i$
16 $\quad \boldsymbol{G_{corrupted}} \leftarrow \text{CorruptGraph}(\boldsymbol{G}, R_c)$
17 $\quad$ **return** $\boldsymbol{G_{corrupted}}$

those obtained with our method, the ones approximated with MC method and a constant covariance for each experiment. The results of the optimisation processes have then been compared based on two different error metrics, $RMSE_{xyz}$, [28] and the map-to-map metric proposed in [29]:

- $RMSE_{xyz}$: measures the error in reconstructing the AUV trajectory.
- The map-to-map error: measures the geometric consistency of the final map on overlapping regions.

The aim of these tests is to assess the influence of the GICP covariances on the quality of the PoseSLAM solution. To this end, two modifications have been introduced in the construction of the pose graph with respect to [21]:

1) The submaps created are all of roughly the same length.
2) The initial map and vehicle trajectory are optimised in our graph SLAM framework using an estimate of the real R from the vehicle and the MC estimated covariances for Q in Eq. (12) and used in lieu of actual ground truth, 'GT'.

This second point can be further motivated by: i) the relatively high quality of the survey together with the absence of actual GT; ii) we will be disrupting the vehicle trajectory by adding Gaussian noise much greater than the navigation errors. Thus comparing the different optimisation outputs with respect to the undisrupted optimised estimate gives a valid comparison of methods as long as estimates are sufficiently further from the navigation than we believe the navigation is from the actual ground truth. More to the point we do not intend to prove that the MC method leads to a consistent estimate, for that we refer to the [11]. Instead we wish to show that we can approximate well the solution that the MC method gives with our method.

C. Appendix B

Algorithm 1 outlines the process followed to create the graphs for the optimisation tests. As input it requires a 'GT' dataset, which we approximate optimising the navigation estimate from the vehicle DR as in [21] using the MC covariances computed. Given the estimated GT dataset divided into a set of N submaps $\boldsymbol{S_N}$, a graph $\boldsymbol{G}$ is constructed in lines 2 to 12. The initial bathymetry map from the undisrupted data can be seen in Figs. 5a and 5b. In line 14 the output graph is corrupted with additive Gaussian noise with covariance R_c. An instance of the resulting bathymetry and graphs are shown in Figs. 5c and 5d respectively. The arrows among consecutive submaps represent DR edges, while the non-consecutive ones depict the LC constraints. The noise has been added to the graph once built instead of to the navigation data to ensure that the loop closure detections are preserved disregarding of the noise factor used. However, an extra step must be taken to ensure a registration consistent with the assumptions on GICP initialisation as in Section IV-D. In the

TABLE IV
GRAPH-SLAM RESULTS FOR THE FOUR SETS OF COVARIANCES USED IN THE OPTIMISATION OF THE TWO 'CORRUPTED' DATASETS. OBSERVE HOW 'OUR' METHOD APPROXIMATES WELL THE 'MC' RESULT FOR THWAITES 11 AND EVEN OUTPERFORMS IT FOR BORNÖ 8

Dataset	Trajectory	Graph constraints	Error (m)	Navigation	Corrupted	Monte Carlo	Constant Q	Naïve GICP	Ours
Thwaites 11	51.9 km	N_{DR} 222 N_{LC} 176	$RMSE_{xyz}$ [28]	15.3	323.4	248.1	287.1	302.8	259.5
			Map-to-map [29]	32.43	34.17	33.75	34.04	33.75	33.65
Bornö 8	47.8 km	N_{DR} 395 N_{LC} 301	$RMSE_{xyz}$ [28]	9.4	210.8	98.8	114.1	76.7	67.3
			Map-to-map [29]	4.00	6.25	4.08	4.05	4.11	4.06

case of a loop detection, the target submap S_i is perturbed with a transformation drawn from O before being registered against the fused submaps in $\boldsymbol{S_{LC}}$. That is we assume that there is, along with a good loop closure detection, an estimated starting point for GICP as given by the distribution O. The $coverage$ variable determines how much overlap must exist between two submaps for it to be considered a loop closure. It has been set to 60% of S_i.

VI. RESULTS

A. Assessment of the NN Predictions

We assess the performance of π on the training, validation, and testing sets with the KL divergence loss described in Section IV-C. In Fig. 4, the training and validation loss maintain a strong coherence, indicating a good generalisation performance. As indicated in Table II, π not only converges to low training and validation values, it also generalises well to the unseen testing sets. Understandably, the network performs quite well in the Bornö 8 validation set, in comparison to Thwaites 11, because the former is rather homogeneous in terms of feature types, whereas the latter includes many severe features on a larger scale.

B. SLAM Results

The assessment of the covariances on an underwater SLAM context has been carried out on the two subsets from the Antarctica and Bornö datasets in Fig. 5. The results from the optimisation of the graphs generated from Algorithm 1 can be seen in Table IV. They have been averaged over 100 repetitions where noise was added to the 'GT', which was then optimised. The $RMSE_{xyz}$ error under 'Navigation' indicates the correction from the DR estimate of the navigation after optimising it. The resulting trajectory and bathymetry have been then corrupted with Gaussian noise parameterized by a covariance R_c^{6x6} whose only non-null component is $R_c(5,5) = 0,01$, modelling the noise added to the yaw of the vehicle. The average errors from the corrupted graphs are given on the column "Corrupted". The next four columns contain the final errors after the optimisations carried out with the covariances generated with the different methods tested. Our method has been compared against the covariances generated from the MC approximation in Section IV-D and against two baseline approaches. A vanilla method consisting on approximating the information matrix of GICP, so called here "Naïve GICP", using the average of the information matrix in Eq. 2 in the original paper [7], projected onto the xy plane for the current experiments. This approximation doesn't require any extra computation and can be run during the mission, but is sensitive to the noise and sparsity of the point clouds. For the second method, "Constant Q" as in [16], all LC edges have the same Q_i, which in our case is computed a posteriori by averaging the MC solutions of all the submaps.

When looking at the Thwaites 11 results, considering the 'MC' method as the gold standard, it can be seen that it yields the smallest $RMSE_{xyz}$ error, as expected. Our method outputs a slightly worse $RMSE_{xyz}$ and similar Map-to-map error to MC and is significantly better than the two baseline approaches. However, in the Bornö 8 dataset our method's $RMSE_{xyz}$ error is better than the 'MC' output and very similar to the 'Naïve GICP'. This might be explained by the bathymetric data itself. The submaps from Bornö contain far fewer features than those from Thwaites, and usually on the edges. This makes the submap's terrain less homogeneous, violating our assumption on Section IV. It is possible that the network had difficulty learning these non-homogeneous cases and tended to treat mostly flat submaps more like completely flat. Then in the actual estimation, where the feature parts may not overlap at all, this resulted in a better output of the optimisation. The covariances from the 'Naïve GICP' are generally flat for large, noisy bathymetric point clouds. This explains why they have performed so well in this dataset and so poorly in Thwaites 11, which contains more features.

C. Runtime Comparison

The execution times of 'MC' and 'Our' method have been compared on an Intel Core i7-7700HQ with 15,6 GiB RAM. The average covariance generation time over 107 submaps of size $mean = 6552.75, std\_dev = 766.00$ can be seen in Table III, supporting the claim that PointNetKL offers the best trade off between accuracy and processing time to run SLAM online on an AUV.

VII. CONCLUSIONS

We have presented PointNetKL, an ANN designed to learn the uncertainty distribution of the GICP registration process from unordered sets of multidimensional points. In order to train it and test it, we have created a dataset consisting of bathymetric point clouds and their associated registration uncertainties out of underwater surveys, and we have demonstrated how the architecture presented is capable of learning the target distributions. Furthermore, we have established the performance of our model within a SLAM framework in two large missions outside the training set.

The results presented indicate that PointNetKL is indeed able to learn the GICP covariances directly from raw point clouds and generalise to unknown environments. This motivates the

possibility of using models trained in accessible environments, such as the Baltic, to enhance SLAM in unexplored environments, e.g. Antarctica. Furthermore, the data generation process introduced has proved to work well and alleviate the need for a dataset with sequences of overlapping point clouds with ground truth positioning associated, which are scarce in the underwater domain. Further testing of the network on new graph-SLAM optimisations is required to fully characterise its performance and limitations, but the results presented support our thesis that PointNetKL can be successfully applied in online SLAM with AUVs.

The future extension of this work to the 3D domain will focus on estimating the heading of the AUV together with its $[x, y]$ state. Our intuition on this is that the same features in the submaps that anchor the registration in $[x, y]$ would lead the process if the GICP was unconstrained in the yaw as well. This means that the features extracted by PointNetKL should, in general, perform well in 3D.

ACKNOWLEDGEMENT

The authors thank MMT for the data used in this work and the Knut and Alice Wallenberg foundation for funding MUST, Mobile Underwater System Tools, project that provided the Hugin AUV for these tests.

This work was supported by Stiftelsen för Strategisk Forskning (SSF) through the Swedish Maritime Robotics Centre (SMaRC) (IRC15-0046).

REFERENCES

[1] D. Landry, F. Pomerleau, and P. Giguère, "Cello-3d: Estimating the covariance of ICP in the real world," in *Proc. Int. Conf. Robot. Autom.*, 2019, pp. 8190–8196.

[2] C. R Qi, H. Su, K. Mo, and L. J Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 652–660.

[3] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1271–1278.

[4] P. V Teixeira, M. Kaess, F. S Hover, and J. J Leonard, "Underwater inspection using sonar-based volumetric submaps," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 4288–4295.

[5] P. J Besl and N. D McKay, "Method for registration of 3-d shapes," in *Proc. Sensor fusion IV: Control Paradigms Data Structures*, 1992, vol. 1611, pp. 586–606.

[6] I. Torroba, N. Bore, and J. Folkesson, "A comparison of submap registration methods for multibeam bathymetric mapping," in *Proc. IEEE/OES Auton. Underwater Vehicle Workshop*, 2018, pp. 1–6.

[7] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proc. Robot.: Sci. Syst. V*, University of Washington, Seattle, USA: The MIT Press, vol. 2, Jun. 2009, doi: 10.15607/RSS.2009.V.021.

[8] A. Censi., "An accurate closed-form estimate of icp's covariance," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3167–3172.

[9] S. M. Prakhya, L. Bingbing, Y. Rui, and W. Lin, "A closed-form estimate of 3d icp covariance," in *Proc. 14th IAPR Int. Conf. Mach. Vision Appl.*, 2015, pp. 526–529.

[10] M. Barczyk and S. Bonnabel, "Towards realistic covariance estimation of icp-based kinect v1 scan matching: The 1d case," in *Proc. Amer. Control Conf.*, 2017, pp. 4833–4838.

[11] A. G. Buch, D. Kraft, et al, "Prediction of icp pose uncertainties using monte carlo simulation with synthetic depth images," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 4640–4647.

[12] A. Tallavajhula, B. Póczos, and A. Kelly, "Nonparametric distribution regression applied to sensor modeling," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 619–625.

[13] V. Peretroukhin, L. Clement, M. Giamou, and J. Kelly, "Probe: Predictive robust estimation for visual-inertial navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 3668–3675.

[14] M. Savva *et al.*, "Large-scale 3d shape retrieval from shapenet core55: Shrec'17 track," in *Proc. Workshop 3D Object Retrieval*, 2017, pp. 39–50.

[15] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 5648–5656.

[16] K. Liu, K. Ok, W. Vega-Brown, and N. Roy, "Deep inference for covariance estimation: Learning gaussian noise models for state estimation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 1436–1443.

[17] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 4490–4499.

[18] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Trans. Graph. (TOG)*, New York, NY, USA: ACM, vol. 38, no. 5, pp. 1–12, 2019.

[19] Y. Aoki, H. Goforth, R. Arun Srivatsan, and S. Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 7163–7172.

[20] T. Zheng, C. Changyou, Y. Junsong, L. Bo, and R. Kui *et al.*, "Pointcloud saliency maps," in *Proc. IEEE Int. Conf. Computer Vision*, 2019, pp. 1598–1606.

[21] I. Torroba, N. Bore, and J. Folkesson, "Towards autonomous industrial-scale bathymetric surveying," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6377–6382.

[22] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 679–693, Jun. 2014.

[23] L. Zhou, X. Cheng, and Y. Zhu, "Terrain aided navigation for autonomous underwater vehicles with coarse maps," *Meas. Sci. Technol.*, vol. 27, no. 9, 2016, Art. no. 095002.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, May 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[26] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7

[27] Y. Yao, L. Rosasco, and A. Caponnetto, "On early stopping in gradient descent learning," *Constructive Approximation*, vol. 26, no. 2, pp. 289–315, 2007.

[28] E. Olson and M. Kaess, "Evaluating the performance of map optimization algorithms," in *Proc. RSS Workshop Good Exp. Methodology Robot.*, vol. 15, 2009.

[29] C. Roman and H. Singh, "Consistency based error evaluation for deep sea bathymetric mapping with robotic vehicles," in *Proc. Int. Conf. Robot. Autom.*, 2006, pp. 3568–3574.