3DEG: Data-Driven Descriptor Extraction for Global re-localization in subterranean environments

Nikolaos Stathoulopoulos
niksta@ltu.se

Anton Koval
antkov@ltu.se

George Nikolakopoulos
geonik@ltu.se

Luleå University of Technology,
Department of Computer, Electrical and Space Engineering,
Robotics & AI Group,
Luleå, 97187, Sweden

Abstract

Current global re-localization algorithms are built on top of localization and mapping methods and heavily rely on scan matching and direct point cloud feature extraction and therefore are vulnerable in featureless demanding environments like caves and tunnels. In this article, we propose a novel global re-localization framework that: a) does not require an initial guess, like most methods do, while b) it has the capability to offer the top-*k* candidates to choose from and last but not least provides an event-based re-localization trigger module for enabling, and c) supporting completely autonomous robotic missions. With the focus on subterranean environments with low features, we opt to use descriptors based on range images from 3D LiDAR scans in order to maintain the depth information of the environment. In our novel approach, we make use of a state-of-the-art data-driven descriptor extraction framework for place recognition and orientation regression and enhance it with the addition of a junction detection module that also utilizes the descriptors for classification purposes.

Keywords global re-localization, sparse 3D LiDAR scans, 3D point cloud map, deep learning

1 Introduction

As autonomous robot exploration in GPS-denied environments is gaining more and more attention, it is constantly creating the demand for robust algorithms, capable of handling harsh, poorly illuminated, unstructured and unexplored environments, essentially making them safer for humans to explore Nikolakopoulos and Agha [2021], Lindqvist et al. [2022]. An important part of an exploration or navigation mission is the ability to operate under a global map that provides useful information for path planning, coordination of multiple robots, localization of objects and survivors in Search And Rescue (SAR) missions. In these scenarios, it is common for the localization algorithms, to momentarily fail due to sensor faults, dust particles or drifting, resulting in misalignment of the robot's current pose in the global map. Thus, the task of global re-localization focuses on estimating the current robot pose in a given map, enabling to restart missions in previously mapped environments or correct the aforementioned misalignment.

Even though, camera images have been of great interest for place recognition, mainly due to their rich and descriptive information, they struggle with environment changes and fail under low-light applications Kominiak et al. [2020], Agha et al. [2021]. On the other hand, LiDAR sensors are immune to appearance changes and illumination Shan et al. [2020a] and along with the advances of efficient data representations and feature descriptors of LiDAR point clouds through deep learning, they have demonstrated promising results Schaupp et al. [2019], Du et al. [2020] in the field of computer vision.

In this article, we will revisit the problem of global re-localization, by proposing an overall novel scheme with a state-of-the-art data-driven descriptor extraction process for 3D LiDAR point clouds. With the focus on subterranean environments with low features and multi-robot exploration, we present an event-based, global re-localization framework,

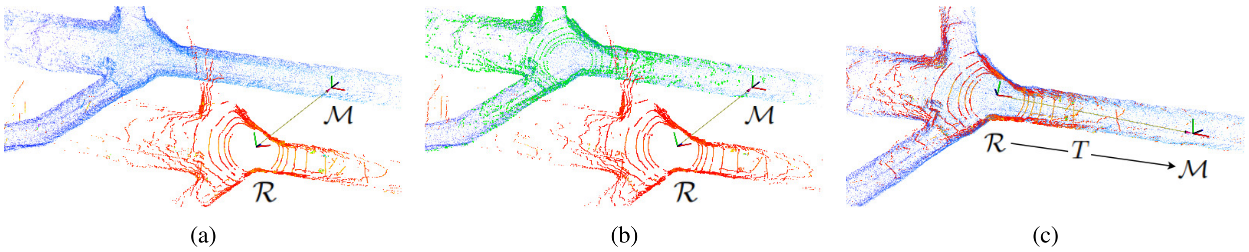

Figure 1: (a) Initially, the LiDAR scan of the robot frame $\mathcal{R}$ is not aligned with the point cloud map of the map frame $\mathcal{M}$. (b) Then, during the re-localization process, the nearest submap based on the indexes of the vectors in the k-d tree, denoted with green color, is chosen and used to find the transform T. (c) Finally, the LiDAR scan of the robot is aligned with the map.

that extends the autonomy of the robots. Our contributions can be summarized as follows: (a) we introduce a complete framework for re-localization in a given global 3D point cloud map, as a single file and not sets of point clouds. The novel approach utilizes data-driven descriptors for place recognition, as depicted in Figure 2, with the ability to go through the top-k candidates, which further extends the resiliency of the overall system. (b) Through real-world field experiments we demonstrate a robust performance in terms of successful re-localization in challenging subterranean environments and without the need of a manual initial guess, compared to other ROS Stanford Artificial Intelligence Laboratory et al. available packages with a similar input format, such as LIO-SAM_based_relocalization Shan et al. [2020b], FAST-LIO_localization Xu and Zhang [2021] and hdl_global_localization Koide et al. [2019]. (c) We propose a novel modular architecture for driving descriptors towards specific events or tasks through the addition of a classification module that can extend the autonomy of robot exploration, since the robot can decide when to trigger the re-localization process based on an event, like the detection of a junction, that will maximize the likelihood of success. (d) With joint training, negative mining, fine-tuning and label smoothing, we showcase the process of data-handling for transitioning from big public datasets and place recognition solutions to learning from a few field mission data and a direct semantic global re-localization approach that to the best of the authors' knowledge is absent from the current literature.

2 Related work

Predominantly, the global re-localization problem is composed of two stages: (a) place recognition, which derives the frame in the map that is topologically close to the current frame, and (b) pose estimation, which yields the relative pose from the map frame to the current robot's frame. Our proposed framework can be seen as a bridge between descriptor extraction for place recognition and global pose estimation in a prebuilt point cloud map. Therefore, we divide this Section into related work for learned descriptors for place recognition and/or pose estimation and related work for available global re-localization solutions in a point cloud map.

2.1 Learned Descriptors:

Descriptors can be categorized as handcrafted He et al. [2016], Kim and Kim [2018], hybrid Vidanapathirana et al. [2020] or learned based Schaupp et al. [2019], Du et al. [2020], Uy and Lee [2018], Vidanapathirana et al. [2021], Dubé et al. [2018]. Handcrafted methods have the advantage of not requiring re-training to adapt to different environments or platforms. Even though methods, such as the well known ScanContext Kim and Kim [2018] have demonstrated reliable performance in varying scenarios, the discriminatory ability of such methods remains limited. As the name suggests, hybrid methods, aim to unite mathematical models with data-driven models to benefit from both Shlezinger et al. [2021]. First demonstrated by Locus Vidanapathirana et al. [2020], an approach for LiDAR based place recognition by mathematically modeling the topological relationships and temporal consistencies of point segments, while structural visual aspects of the segments were encoded using a data-driven 3D-CNN. Although it achieved state-of-the-art performance on the KITTI dataset Geiger et al. [2013], it struggles to adapt to environments where the extracted segments are structurally different to its training data. With LocNet Gidaris and Komodakis [2015], Yin et al. Yin et al. [2017] used semi-handcrafted range histogram features, as an input to a 2D Convolutional Neural Network (CNN), demonstrating the power of Deep Neural Networks (DNNs) to learn suitable data representations and exploiting the most relevant cues in the input data. Learned based methods have shown promising results with the universal approximation function properties of neural networks Hornik et al. [1989] and in the past years, CNNs have become the state-of-the-art method for generating learning based descriptors, mainly due to their ability to find complex patterns in data Krizhevsky et al. [2012]. PointNetVLAD Uy and Lee [2018] pioneered by using an end-to-end trainable global descriptor for 3D point cloud recognition. Extracted local features from PointNet Qi et al. [2016] are deployed to the NetVLAD aggregator Arandjelović et al. [2015] to form a global descriptor of the scene. Recently developed SegMap Dubé et al. [2018] makes use of CNNs for encoding small dimensional representations and decoding to decompress the representations back to the original input, as part of the five core modules: segment extraction, description, localization, map reconstruction and1 Introductionare then used for both fetching the nearest neighbor place and for estimating yaw discrepancy.

2.2 Global re-localization

Currently, there are only a few ROS packages available, that support global re-localization in a 3D point cloud map. Koide et al. Koide et al. [2019] provided a series of packages that also included global re-localization as part of the localization and mapping process. The localization process is done by an Unscented Kalman Filter-based pose estimation, fusing IMU and 3D LiDAR data. In the sequel, the scheme performs a Normal Distribution Transform (NDT) scan matching between the global map and the input scan in order to correct the estimated pose. For the global re-localization it provides three engines, Branch and Bounce Search (BBS) Hess et al. [2016], FPFH+RANSAC Rusu et al. [2009], Buch et al. [2013] or FPFH+Teaser++ Yang et al. [2021]. Based on LIO-SAM Shan et al. [2020b], LIO-SAM_based_re-localization makes use of multiple factors, from the IMU, the 3D LiDAR and the loop closure process, to jointly optimize the factor graph. The introduction of the key-frames and the sliding window scan-matching approach, selectively registers new key-frames to a fixed size set of prior sub-key-frames in order to improve real-time performance. The latest package developed is based on FAST-LIO Xu and Zhang [2021] and it provides a global pose estimation in a pre-built point cloud map, by fusing low-frequency global localization and high-frequency odometry. With a feature extraction process, edge and planar features are extracted from the input 3D LiDAR scan and along with the IMU measurements are fed into the state estimation module.

Our proposed re-localization framework serves as a bridge between the place recognition methods and the available integrated re-localization solutions. We address the limitation of most place recognition methods, that rely on data sequences instead of a global point cloud map, as well as the fact that we introduce a modular architecture to improve performance in certain environments depending on the features. In addition to this, we address the limitation of most re-localization methods, that use a single global point cloud map file as an input, that they either need a manual initial guess or they have a long computational time.

3 Methodology

Problem formulation

We will begin by defining the problem addressed in this article and outlining the novel proposed solution pipeline. Afterwards, we will go in further details on the Neural Network architecture, the training process and the experimental results. Our goal is to introduce a global re-localization algorithm able to yield a rigid transform $T(\mathbf{p}) = R(\theta)\mathbf{p} + \mathbf{t}$, so that, from a single 3D LiDAR point cloud scan $P \subset \mathbb{R}^3$, where $\{P\}$ is a set of points $\mathbf{p} \in \{P\}$, we can transform the current robot frame $\mathcal{R}$ to the global map frame $\mathcal{M}$, through a series of rotations $R \in SO(3)$ and translations $\mathbf{t} \in \mathbb{R}^3$. For the rotation matrix we only consider the yaw angle θ, since roll and pitch angles do not change significantly when exploring an underground environment with aerial or ground mobile robots, and for the translation vector we consider all the three axes, x, y, z and therefore we are looking for a 4 DoF transform, keeping the complexity of the overall system lower, yet delivering a robust result in most case scenarios.

$$T(\mathbf{p}) \coloneqq \mathcal{R} \to \mathcal{M} \tag{1}$$

To tackle this problem and acquire the transform $T(\mathbf{p})$, we follow the steps depicted in Figure 2, which can be summarized as: a) Map Partitioning, b) Point Cloud Projection, c) Descriptor Extraction, d) Initial Pose Estimation, and e) Pose Refinement.

3.1 Map Partitioning

Since we are working directly with a point cloud map and not with data sequences, we need to partition the map into individual scans in order to move on to the next stages of the pipeline. Essentially, we are creating a $k - d$ tree database of the visited places, which we can later search efficiently with the descriptors. By providing the point cloud map along with the discrete trajectory, we partition the map into n scans, where n is the number of points in the trajectory. Then we transform the point clouds according to the corresponding trajectory points, so we always have the partitioned map points in respect to the robot frame and not the map frame.

3.2 Point Cloud Projection

The Point Cloud Projection module is responsible for converting the incoming LiDAR point cloud scan $\mathbf{v}$ or the i-th submap scan, into a 2D depth image, using a spherical projection model. A list of point coordinates p_x, p_y and p_z are mapped onto a 2D spherical grid, with their pixel value defined by the range of each point from the sensor's frame. The benefit of converting LiDAR point cloud scans into range images, over using a depth sensor, is the 360° view that provides, giving us the capability of producing orientation invariant descriptors. On the downside, the converted range images capture a less dense view of the surroundings compared to the one from a depth sensor, making it harder to extract detailed features. One extra step in our pipeline before the point cloud projection, as seen on Figure 2, is that of taking the registered point cloud from the LiDAR Inertial Odometry (LIO) module. By not using directly the incoming point cloud scan but getting only the registered points, we can better match the resolution of the range images produced from the map partitioning process.

3.3 Descriptor Extraction

The Descriptor Extraction module derives a compact representation for place recognition, orientation regression and classification information from the surrounding topological characteristics. In this article, we make use of a Convolutional Neural Network (CNN) that takes as an input 2D range images and generates two vectors q and w respectively, which will be further discussed in subsection 4. The vector q encodes orientation invariant, place dependent information, while the vector w is a compact representation for rotation variant information that is used for regressing the yaw discrepancy in a later stage of the pipeline Schaupp et al. [2019]. We use the place specific vectors q_n, generated from the partitioned map scans, to build a $k-d$ tree, and then with the current robot's scan, we predict the vector q_t, and query the $k-d$ tree for nearby place candidates.

3.4 Initial Pose Estimation

After querying the $k-d$ tree with the current vector q_t, we obtain the nearest neighbors along with the corresponding vectors q_k, w_k, and therefore we can get the trajectory points associated, yielding the initial translation vector $\mathbf{t_0}$. The next step is to feed the orientation estimation module with the rotation variant vectors w_t and w_k, which will estimate the yaw discrepancy $\delta\theta$ between the query point cloud P and the nearest retrieved candidate from the partitioned map. With $\delta\theta$, we can form the rotation matrix $R_0(\delta\theta)$ and have a complete initial estimation, denoted as: $T_0(\mathbf{p}) = R_0(\delta\theta)\mathbf{p} + \mathbf{t_0}$.

3.5 Pose Refinement

With the initial pose estimation, we can further refine the pose by using it as an initial condition in a registration algorithm like Iterative Closest Point (ICP) Zhang [2014]. Furthermore, the distance between the two vectors q_t and q_k can be utilized to further define the distance threshold of the registration method and make the process faster.

3.6 Event-based triggering

To further extend the autonomy of our robots, during field exploration missions, we make use of the classification module to identify when the robot has reached a junction. This event will trigger the global re-localization process, for example in multi-robot exploration, where more than one robot needs to share the same map. By providing the current vector q_t, the *classifier* can distinguish between a straight tunnel, a junction, or a turn. The event is triggered when a junction is detected, as it is usually a more featured space and has a higher chance of success.

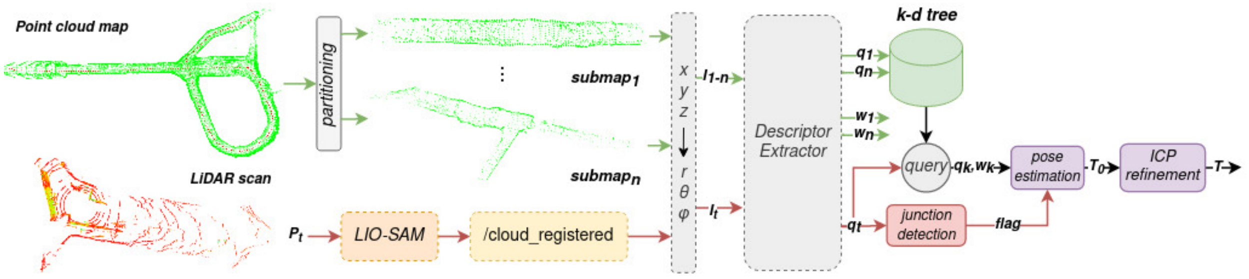

Figure 2: The overall pipeline of the proposed architecture. The red arrows follow the workflow of the current scan P_t, while the green arrows follow the workflow of the given point cloud map.

4 Neural Network

According to the aforementioned related work, we have decided to work with a network architecture based on 2D range images generated from 3D LiDAR scans, and not with the point clouds directly, since the deep learning advancements in feature extraction from images have always demonstrated a robust result. Therefore, our network architecture and descriptor extraction process is based on Schaupp et al. [2019] and is adjusted following the principles described in Simonyan and Zisserman [2014], Appalaraju and Chaoji [2017], as well as our own proposed addition for driving learned descriptors to certain features.

4.1 Network Architecture

The architecture of the proposed CNN is composed of 2D convolutional layers followed by Max Pooling layers. Then, the Fully Connected layers compress the features and map them into a compact descriptor representation, as depicted in Figure 3. We start with 64 filters for the first convolutional layer and a 5×5 filter size, since we want a larger area to compensate for the sparse nature of our depth images. For the other two layers, we move down to 32 filters with a 3×3 filter size. The output of the *descriptor extractor* are 2×64×1 vectors, q and w respectively. As mentioned before, the vector q encodes place dependent information and is invariant to orientation changes, while w is an orientation specific vector and is used to decrease the angle discrepancy Schaupp et al. [2019]. This process is handled by an extra *orientation estimation* module, which takes as an input two vectors w and outputs a 2×1 vector y_{yaw}, after two fully connected layers and a tanh activation function. The third module of the presented network is a classifier and is consisted of two fully connected layers and a *softmax* activation function. In this case, we use the *classifier* to detect the topological characteristics of the surrounding environment and more specific, to classify among: (a) a straight tunnel, (b) a junction, or (c) a turn. The classification process is performed based on a derived vector q and depending on the mission, the *classifier* can be trained on detecting other characteristics, for example pipelines or shafts.

4.2 Loss functions

The overall network is composed of three different modules, where each one of them pursues a different goal. The *descriptor extraction* module needs to find two orthogonal vectors q and w. The *orientation estimation* module estimates the yaw difference from two compact vectors, w and the *classifier* has the goal of predicting the correct class, based on a descriptor vector q. For each of these three goals, a loss function is defined, denoted as L_{pr} for the place-recognition loss, L_θ for the orientation loss and L_c for the classifier's loss. Starting with L_{pr}, in order to train our network for the task of place recognition, we use the triplet loss method Hoffer and Ailon [2014]. As demonstrated in Figure 3, we feed the neural network with three types of images: anchor images, similar to the anchor images and dissimilar images, denoted as I_A, I_S and I_D respectively. We also define d_S as the Euclidean distances between the descriptors q_A from I_A and the descriptors q_S from I_S. The same applies for d_D and the descriptors q_D from I_D. The loss function is designed so that similar and dissimilar point cloud pairs are pushed close together and far apart in the derived vector space. The parameter m is a margin distance for distinguishing between similar and dissimilar pairs. The triplet loss is defined as follows:

$$L_{pr}(d_S, d_D) = \underbrace{||f(I_A) - f(I_S)||^2}_{d_S} - \underbrace{||f(I_A) - f(I_D)||^2}_{d_D} + m \tag{2}$$

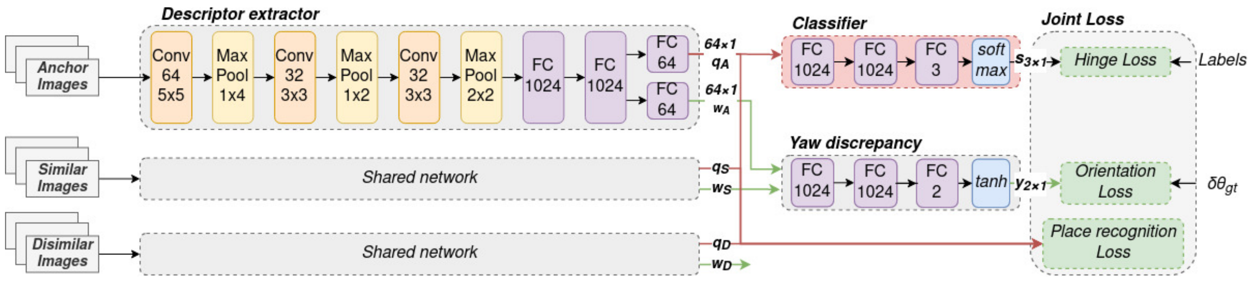

Figure 3: The Siamese network consists of three convolutional layers followed by max pooling and then two fully connected layers, as proposed in Schaupp et al. [2019]. The number of filters and the kernel size have been adjusted to the considered testing environments and sensor utilized, and in addition we have an extra classifying module that boosts the descriptor extraction process.

The orientation estimation loss L_θ is a regression loss function that is computed based on the *orientation estimation* module's output y_{yaw} and the ground truth $\delta\theta_{gt}$. For predicting the orientation discrepancy, we only make use of the orientation dependent vectors w_A and w_S. The orientation loss is defined as:

$$L_\theta(y_{yaw}, \delta\theta_{gt}) = \frac{1}{2}\left((y_{yaw,0} - \cos(\delta\theta_{gt}))^2 + (y_{yaw,1} - \sin(\delta\theta_{gt}))^2\right) \tag{3}$$

As mentioned in Schaupp et al. [2019], by transforming the ground truth yaw angle $\delta\theta_{gt}$ into the Euclidean space, we avoid the ambiguity between $0 - 360°$ which could lead to false corrections during training. The last loss function we need to define is the classification loss L_c. For this, we choose the Hinge Loss function, defined as follows.

$$L_c(s_j, s_{l_i}) = \sum_{j \neq s_i} \max(0, s_j - s_{l_i} + 1) \tag{4}$$

Essentially, the Hinge Loss function is summing across all the incorrect classes ($i \neq j$) and comparing the output of the predicted vector s returned for the j-th class label (the incorrect class) and the l_i-th class (the correct class).

4.3 Training the descriptors

For training all the modules of our network, we use a joint training process to achieve a good localization recall, an accurate yaw angle estimation and a robust classification, by learning the weights of all three Neural Networks together. For this, we combine all three loss functions, defined as L:

$$L = L_{pr} + L_\theta + L_c \tag{5}$$

We sample the training point cloud data and then based on the margin m and their ground truth poses, we characterize a similar and a dissimilar to the anchor point cloud, in order to prepare the triplets for the three-tuple shared network. As a data augmentation step, we randomly rotate the point clouds around the yaw axis, making sure that the orientation between anchor and the similar point clouds is different while still being from a similar place. The three point clouds are converted to the range images and then are fed to the *descriptor extractor* network that outputs the three corresponding pair-vectors, (q_A, w_A), (q_S, w_S) and (q_D, w_D). The three place dependent vectors are used to compute the L_{pr} loss, while w_A and w_S are passed to the *orientation estimation* network and the corresponding output y_{yaw} along with $\delta\theta_{gt}$ are used to compute the L_θ loss. The vector q_A is also passed to the *classifier*, where the output s with the labels are used to compute the L_c loss. The combined loss L is then evaluated and with the ADAM Kingma and Ba [2014] learning optimizer the weights are updated.

5 Experiments and Results

In this section, we will go through the experimental results, starting from evaluating the performance of the Neural Network architecture, as well as comparing it to its base version. Then we will further evaluate the complete proposed framework and compare it to the existing ROS available solutions. All experiments are carried out in real-world settings with a focus on subterranean environments.

5.1 Neural Network evaluation

5.2 Datasets

For the training and evaluation process, we use three dataset collections. The first dataset collection Koval et al. [2022a] contains recordings from an underground tunnel located in Luleå, Sweden, as seen on Figure 4. For this area, we have recordings from two different robotic platforms. The first is with Spot from Boston Dynamics Dynamics and Robotics, equipped with an autonomy package Koval et al. [2022b], that includes the Velodyne VLP16 PuckLite 3D LiDAR. The second robotic platform is a custom-built quadrotor Lindqvist et al. [2021] and is equipped with the same 3D LiDAR and on-board computer as Spot. It is important to mention that even though both platforms have similar sensors, the acquired data may differ due to movement noise, dust and accuracy of the IMU, as one is a ground quadruped robot and the other is a flying robot. The main difference can be seen on the registered point clouds, as well as the generated range images, since they are operating in different heights and with different form of vibration due to walking or flying. The second dataset collection, is from a real underground mining facility. Unlike the first dataset collection, this one features larger tunnels, up to 10 meters wide, with multiple junctions and a featureless environment. The third and last dataset collection is from the same underground tunnel, as the first one, but from a different passage. This environment offers a more narrow and corridor-like environment. From all datasets, we make use of the 3D LiDAR scans and the odometry data in order to train our models. The labels for training the junction detection module were hand-crafted on all datasets.

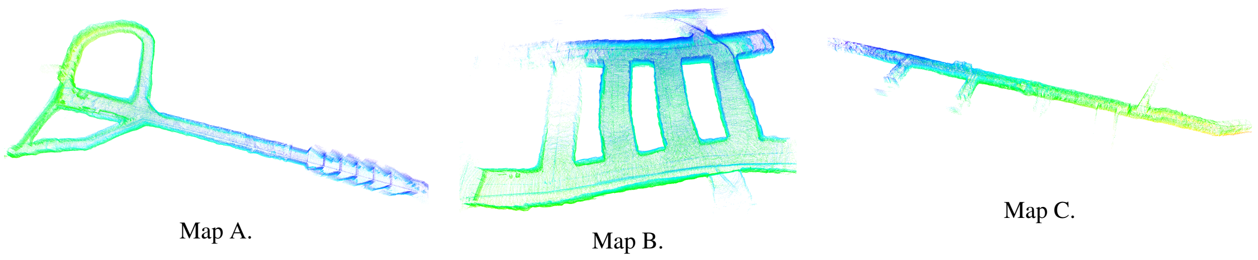

Figure 4: From map A, only the recordings from the upper branch are used for training, while the rest are used for the evaluation. Map B and C are only utilized for evaluation.

5.3 Data sampling and training process

As mentioned in Section 4, the neural network is based on the triplet network architecture and therefore requires sampling three tuples of anchor, positive and negative pairs. We consider two point clouds as similar, if their ground-truth poses, defined as p, are within 3 meters, $|p_A - p_S| \leq 3$. For the negative pair, we first randomly sample point clouds out of the 3 meters radius, $|p_A - p_D| \geq 3$, and then on a later stage we sample them within a $3 - 6$ meters radius, $3 \leq |p_A - p_D| \leq 6$, around the anchor. This is known as the hard-negative mining strategy Bucher et al. [2016], which provides a boost in the network performance as it utilizes harder to distinguish triplets in the advanced phase of convergence. As we are working with limited datasets from the subterranean environments, we choose to train the neural network with data collected from only the first dataset collection. For the training process of the *classifier*, it is important to have a balanced dataset. Natively, the dataset will be biased towards straight corridors due to the environment and therefore, we first train our neural network with the complete dataset without the classifying module. Then we sample the initial dataset in a way that it is balanced among the three classes; a) straight corridor, b) junction, and c) turn, and perform additional training with the classifying module enabled. Furthermore, because it is hard to determine when a junction starts and ends, we make use of the regularization technique of label smoothing Goodfellow et al. [2016], Müller et al. [2019].

$$labels_{new} = labels_{old} \cdot (1 - a) + \frac{a}{N} \tag{6}$$

Label smoothing helps us tackle the problem of overfitting and overconfidence of the *classifier*. By applying soft labels, as calculated in Equation 6, instead of the hard 0 and 1, we ultimately get a lower loss when there is an incorrect prediction and subsequently, our model will penalize and learn incorrectly by a slightly less degree. The hyperparameter a determines the amount of smoothing and N is the number of classes.

5.4 Place recognition results

An advantage of frameworks like Schaupp et al. [2019] over the other discussed re-localization frameworks, is that they offer the top-k candidates for the place recognition problem. As seen on Figure 5 and in Table 1, the localization recall results show that 3DEG outperforms the OREOS in all scenarios, while in some cases the base model performs better than the one with the extra classifying module. The recall percentage is higher on the first map due to being the map that we used part of to train the neural networks. In addition, we notice that for the second map, that contains the most

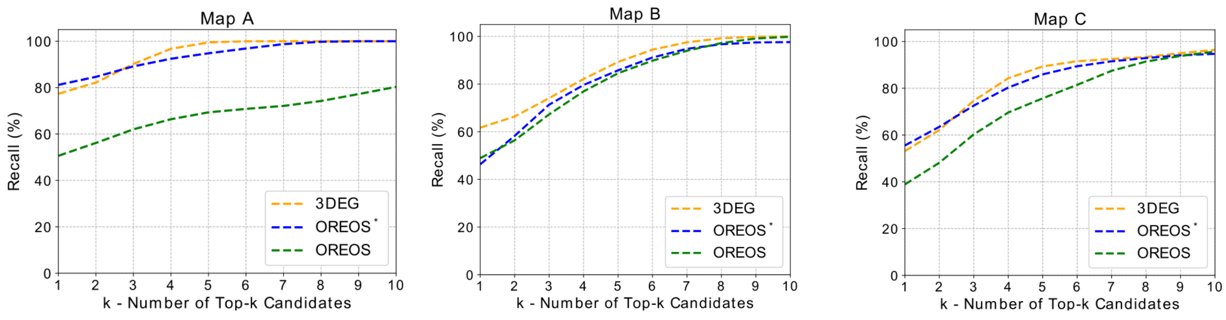

Figure 5: Comparison of the performance of the proposed framework 3DEG, our implementation of OREOS, as described by Schaupp et al., and our implementation of OREOS but with adjusted network parameters as seen on Figure 3, denoted as OREOS∗.

junctions, the junction detection module provides a significant boost on the top-1, with more than 10%. The results of Table 1, for the first two rows, do not include the ICP refinement. It demonstrates the mean error of the yaw discrepancy estimation before the refinement, between the robot's frame and the chosen submap frame. From our experience, if two point clouds have a high rotational discrepancy (more than 15^o-25^o), ICP fails to align them properly. On the other hand, after the yaw estimation and the initial pose estimation performed by our framework, the yaw discrepancy will be less than 20^o and therefore the ICP can align them successfully. As expected, there is no major difference in the performance of the yaw estimation, with the mean and standard deviation matching that of OREOS.

	Map A			Map B			Map C		
METHOD	3DEG	OREOS*	OREOS	3DEG	OREOS*	OREOS	3DEG	OREOS*	OREOS
MEAN (deg)	14.98	**13.53**	14.47	**14.72**	15.22	15.38	17.44	**16.45**	16.59
STD (deg)	22.34	23.14	**19.41**	19.87	20.63	**19.60**	**21.49**	21.90	22.18
MEAN (m)	**0.23**	-	-	0.25	-	-	0.28	-	-
STD (m)	**0.12**	-	-	0.17	-	-	0.19	-	-
RECALL (%)	**92.4**	-	-	89.1	-	-	91.5	-	-

Table 1: Comparison of the mean absolute error and the standard deviation in degrees for the yaw estimation, the recall percentage of the *classifier* and the mean and standard deviation in meters for the final estimated pose.

Moreover, we present the mean error of the final estimated pose from the ground truth and the standard deviation. The results for each map arose from running the relocalization process as the robot explores the map, for approximately every meter traveled. In Table 1, we only present them for 3DEG since the final estimation is performed by the ICP registration.

5.5 Global re-localization results

A part of our contribution is that the proposed framework is a complete global re-localization package that works with a given 3D point cloud map and a trajectory, by utilizing a place recognition framework, and thus we evaluate its performance against the available re-localization ROS packages, mentioned in Section 2. In Table 2, we present the time that each package needs to re-localize, as well as the CPU load and the VRAM usage. The BBS engine from the hdl_global_localization was not able to correctly re-localize in any of the tested places, and consequently was not included in the table. For the FPFH+RANSAC engine, both methods of DIRECT1 and DIRECT7 were tested, and we have included only the fastest one. Even though we noticed higher re-localization times and memory usage than LIO-SAM_based_re-localization and FAST-LIO_localization, it is worth noting that the biggest delay in our pipeline is the final ICP registration for refining the pose, which can be replaced with other faster registration methods like Fast-ICP Zhang et al. [2021] or TEASER++ Yang et al. [2021]. Throughout our experiments, only FAST-LIO_localization was able to keep a robust re-localization performance and that only after a very precise initial guess, something that is not required by the proposed 3DEG framework.

	Starting long corridor				Lower corridor			
METHOD	FPFH+RANSAC	LIO-SAM	FAST-LIO	3DEG	FPFH+RANSAC	LIO-SAM	FAST-LIO	3DEG
TIME (sec)	2.61+22.99	0.501	**0.229**	1.331	-	-	**0.205**	1.212
CPU (%)	83.5	**13.4**	14.5	19.7	-	-	**9.3**	15.3
VRAM (GiB)	**1.87**	1.96	4.01	5.00	-	-	**4.01**	5.00
MEAN (m)	0.64	0.44	0.35	**0.30**	-	-	0.37	**0.28**
	1st junction				2nd junction			
METHOD	FPFH+RANSAC	LIO-SAM	FAST-LIO	3DEG	FPFH+RANSAC	LIO-SAM	FAST-LIO	3DEG
TIME (sec)	1.26+24.61	1.018	**0.164**	1.294	12.62+23.22	0.552	**0.132**	1.113
CPU (%)	92.1	14.8	15.9	**13.6**	86.9	**14.2**	15.5	14.5
VRAM (GiB)	**1.87**	1.96	4.01	5.00	**1.87**	1.96	4.01	5.00
MEAN (m)	0.60	0.35	0.32	**0.23**	0.62	0.39	0.36	**0.24**

Table 2: Experimental evaluation of the available re-localization packages, from Map A of Figure 4, with comparison of the average computational time, CPU load, memory usage and mean error.

6 Limitations

Nevertheless, our approach still has limitations. Working in subterranean environments, where the presence of dirt and dust is directly translated into noise, significantly affects the low-resolution VLP16 scans. This results in the degradation of the resolution of the generated range images, making it hard to train the descriptors. Another limitation is the currently used registration method, which can either fail to refine the pose or have a high time and computational cost, especially if the distance threshold is not chosen properly. The angle regression is only present in the yaw angle, providing a 4 DoF initial estimation instead of 6 DoF. To accommodate a different type of environment, re-training is needed with a new classifier, better capturing the features of that environment. Last but not least, as a future step we are planning to better optimize the code, which for the moment is not optimal and highly affects the runtime and memory usage of the algorithm.

7 Conclusions

In this article we have established the 3DEG, a novel complete global re-localization in a 3D point cloud map framework, based on data-driven descriptors, that is able to autonomously start the process of re-localization upon the detection of a junction in an environment. In addition to this, our proposed framework offers resiliency by providing multiple candidates in a semi-autonomous operation, increasing the success rate for important missions like search and rescue. All in all, this paper's main goal is the proposal of a full relocalization pipeline suitable for challenging tunnel environments where existing conventional place recognition only or pose estimation only methods seem to fail.

Acknowledgments

The authors would like to give special thanks to Epiroc AB for providing access to the mining areas. This work has been partially funded in part by the European Unions Horizon 2020 Research and Innovation Programme under the Grant Agreements No. 869379 illuMINEation, No.101003591 NEXGEN-SIMS.

References

George Nikolakopoulos and Ali Agha. Pushing the limits of autonomy for enabling the next generation of space robotics exploration missions. *Computer*, 54(11):100–103, 2021.

Björn Lindqvist, Samuel Karlsson, Anton Koval, Ilias Tevetzidis, Jakub Haluška, Christoforos Kanellakis, Ali akbar Agha-mohammadi, and George Nikolakopoulos. Multimodality robotic systems: Integrated combined legged-aerial mobility for subterranean search-and-rescue. *Robotics and Autonomous Systems*, 154:104134, 2022. ISSN 0921-8890. doi:https://doi.org/10.1016/j.robot.2022.104134.

Dariusz Kominiak, Sina Sharif Mansouri, Christoforos Kanellakis, and George Nikolakopoulos. Mav development towards navigation in unknown and dark mining tunnels. In *2020 28th Mediterranean Conference on Control and Automation (MED)*, pages 1015–1020. IEEE, 2020.

Ali Agha, Kyohei Otsu, Benjamin Morrell, David D Fan, Rohan Thakker, Angel Santamaria-Navarro, Sung-Kyun Kim, Amanda Bouman, Xianmei Lei, Jeffrey Edlund, et al. Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge. *arXiv preprint arXiv:2103.11470*, 2021.

Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142. IEEE, 2020a.

Lukas Schaupp, Mathias Burki, Renaud Dube, Roland Siegwart, and Cesar Cadena. OREOS: Oriented recognition of 3d point clouds in outdoor scenarios. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, nov 2019. doi:10.1109/iros40897.2019.8968094.

Juan Du, Rui Wang, and Daniel Cremers. Dh3d: Deep hierarchical 3d descriptors for robust large-scale 6dof relocalization. arXiv, 2020. doi:10.48550/ARXIV.2007.09217.

Stanford Artificial Intelligence Laboratory et al. Robotic operating system. URL `https://www.ros.org`.

Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142. IEEE, 2020b.

Wei Xu and Fu Zhang. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter, github: `https://github.com/HViktorTsoi/FAST_LIO_LOCALIZATION`. 2021.

Kenji Koide, Jun Miura, and Emanuele Menegatti. A portable three-dimensional lidar-based system for long-term and wide-area people behavior measurement. volume 16, 02 2019. doi:10.1177/1729881419841532.

Li He, Xiaolong Wang, and Hong Zhang. M2dp: A novel 3d point cloud descriptor and its application in loop closure detection. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 231–237, 2016. doi:10.1109/IROS.2016.7759060.

Giseop Kim and Ayoung Kim. Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4802–4809, 2018. doi:10.1109/IROS.2018.8593953.

Kavisha Vidanapathirana, Peyman Moghadam, Ben Harwood, Muming Zhao, Sridha Sridharan, and Clinton Fookes. Locus: Lidar-based place recognition using spatiotemporal higher-order pooling. arXiv, 2020. doi:10.48550/ARXIV.2011.14497.

Mikaela Angelina Uy and Gim Hee Lee. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. arXiv, 2018. doi:10.48550/ARXIV.1804.03492.

Kavisha Vidanapathirana, Milad Ramezani, Peyman Moghadam, Sridha Sridharan, and Clinton Fookes. Logg3d-net: Locally guided global descriptor learning for 3d place recognition. arXiv, 2021. doi:10.48550/ARXIV.2109.08336.

Renaud Dubé, Andrei Cramariuc, Daniel Dugas, Juan Nieto, Roland Siegwart, and Cesar Cadena. SegMap: 3d segment mapping using data-driven descriptors. In *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation, jun 2018. doi:10.15607/rss.2018.xiv.003.

Nir Shlezinger, Jay Whang, Yonina C. Eldar, and Alexandros G. Dimakis. Model-based deep learning: Key approaches and design guidelines. In *2021 IEEE Data Science and Learning Workshop (DSLW)*, pages 1–6, 2021. doi:10.1109/DSLW51110.2021.9523403.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

Spyros Gidaris and Nikos Komodakis. Locnet: Improving localization accuracy for object detection. arXiv, 2015. doi:10.48550/ARXIV.1511.07763.

Huan Yin, Li Tang, Xiaqing Ding, Yue Wang, and Rong Xiong. Locnet: Global localization in 3d point clouds for mobile vehicles. arXiv, 2017. doi:10.48550/ARXIV.1712.02165.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. volume 2, pages 359–366, 1989. doi:https://doi.org/10.1016/0893-6080(89)90020-8.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. arXiv, 2016. doi:10.48550/ARXIV.1612.00593.

Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. arXiv, 2015. doi:10.48550/ARXIV.1511.07247.

Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278, 2016. doi:10.1109/ICRA.2016.7487258.

Radu Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. pages 3212 – 3217, 06 2009. doi:10.1109/ROBOT.2009.5152473.

Anders Glent Buch, Dirk Kraft, Joni-Kristian Kamarainen, Henrik Gordon Petersen, and Norbert Kruger. Pose estimation using local structure-specific shape and appearance context. In *2013 IEEE International Conference on Robotics and Automation*. IEEE, may 2013. doi:10.1109/icra.2013.6630856.

Heng Yang, Jingnan Shi, and Luca Carlone. Teaser: Fast and certifiable point cloud registration. volume 37, pages 314–333, 2021. doi:10.1109/TRO.2020.3033695.

Zhengyou Zhang. *Iterative Closest Point (ICP)*, pages 433–434. Springer US, Boston, MA, 2014. ISBN 978-0-387-31439-6. doi:10.1007/978-0-387-31439-6_179.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv, 2014. doi:10.48550/ARXIV.1409.1556.

Srikar Appalaraju and Vineet Chaoji. Image similarity using deep cnn and curriculum learning. arXiv, 2017. doi:10.48550/ARXIV.1709.08761.

Elad Hoffer and Nir Ailon. Deep metric learning using triplet network, 2014.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv, 2014. doi:10.48550/ARXIV.1412.6980.

Anton Koval, Samuel Karlsson, Sina Sharif Mansouri, Christoforos Kanellakis, Ilias Tevetzidis, Jakub Haluska, Ali akbar Agha-mohammadi, and George Nikolakopoulos. Dataset collection from a subt environment. *Robotics and Autonomous Systems*, page 104168, 2022a. ISSN 0921-8890. doi:https://doi.org/10.1016/j.robot.2022.104168. URL `https://www.sciencedirect.com/science/article/pii/S0921889022000951`.

Boston Dynamics and Clearpath Robotics. Spot, the agile robot, spot ros package, `https://www.bostondynamics.com/products/spot`, `https://clearpathrobotics.com/spot-robot/`.

Anton Koval, Samuel Karlsson, and George Nikolakopoulos. Experimental evaluation of autonomous map-based spot navigation in confined environments. *Biomimetic Intelligence and Robotics*, page 100035, 2022b.

Björn Lindqvist, Christoforos Kanellakis, Sina Sharif Mansouri, Ali-akbar Agha-mohammadi, and George Nikolakopoulos. Compra: A compact reactive autonomy framework for subterranean mav based search-and-rescue operations. *arXiv preprint arXiv:2108.13105*, 2021.

Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. Hard negative mining for metric learning based zero-shot classification. *CoRR*, abs/1608.07441, 2016.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Juyong Zhang, Yuxin Yao, and Bailin Deng. Fast and robust iterative closest point. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. doi:10.1109/tpami.2021.3054619.