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

Nikolaos Stathoulopoulos *, Anton Koval, George Nikolakopoulos

Robotics and AI Group, Department of Computer, Electrical and Space Engineering, Luleå University of Technology, 971 87 Luleå, Sweden

ARTICLE INFO

ABSTRACT

Localization algorithms that rely on 3D LiDAR scanners often encounter temporary failures due to various factors, such as sensor faults, dust particles, or drifting. These failures can result in a misalignment between the robot's estimated pose and its actual position in the global map. To address this issue, the process of global re-localization becomes essential, as it involves accurately estimating the robot's current pose within the given map. In this article, we propose a novel global re-localization framework that addresses the limitations of current algorithms heavily reliant on scan matching and direct point cloud feature extraction. Unlike most methods, our framework eliminates the need for an initial guess and provides multiple top-k candidates for selection, enhancing robustness and flexibility. Furthermore, we introduce an event-based re-localization trigger module, enabling autonomous robotic missions. Focusing on subterranean environments with low features, we leverage range image descriptors derived from 3D LiDAR scans to preserve depth information. Our approach enhances a state-of-the-art data-driven descriptor extraction framework for place recognition and orientation regression by incorporating a junction detection module that utilizes the descriptors for classification purposes. The effectiveness of the proposed approach was evaluated across three distinct real-life subterranean environments.

1. Introduction

In recent years, there has been a growing emphasis on the exploration of GPS-denied environments using autonomous robots. These environments pose unique challenges, including harsh conditions, poor illumination, lack of structure, and uncharted territories. Consequently, there is an increasing demand for robust algorithms that can effectively navigate and explore these challenging environments while ensuring the safety of human operators (Lindqvist, Karlsson, et al., 2022; Nikolakopoulos & Agha, 2021). In the context of exploration or navigation missions, having access to a reliable global map is crucial. Such a map provides valuable information for tasks such as path planning, coordination of multiple robots, and localization of objects and survivors in Search And Rescue (SAR) missions. However, even with a global map, localization algorithms often encounter temporary failures due to various factors, including sensor faults, dust particles, or drifting. These failures can result in a misalignment between the robot's estimated pose and its actual position in the global map. To address this issue, the process of global re-localization becomes essential, as it involves accurately estimating the robot's current pose within the given map. This enables the resumption of missions in previously mapped environments or the

correction of misalignment issues, ensuring the reliability and accuracy of the robot's navigation. While traditional approaches for place recognition heavily rely on camera images due to their rich and descriptive information, they often struggle with environment changes and are not well-suited for low-light applications (Agha et al., 2022; Kominiak, Mansouri, Kanellakis, & Nikolakopoulos, 2020). In contrast, LiDAR sensors offer distinct advantages in challenging environments. They are immune to appearance changes and variations in illumination, making them highly reliable and robust sensing devices (Shan et al., 2020). Moreover, recent advancements in deep learning techniques have facilitated the development of efficient data representations and feature descriptors for LiDAR point clouds. These advancements have greatly improved the performance of LiDAR-based methods in computer vision tasks, including global re-localization (Du, Wang, & Cremers, 2020; Schaupp, Burki, Dube, Siegwart, & Cadena, 2019). By leveraging these LiDAR-based approaches, the limitations associated with environment changes and low-light conditions can be overcome, making them a viable and promising choice for addressing the challenges of global re-localization and other computer vision tasks in complex and unexplored environments (see Fig. 1).

---

* Corresponding author.
E-mail addresses: niksta@ltu.se (N. Stathoulopoulos), antkov@ltu.se (A. Koval), geonik@ltu.se (G. Nikolakopoulos).
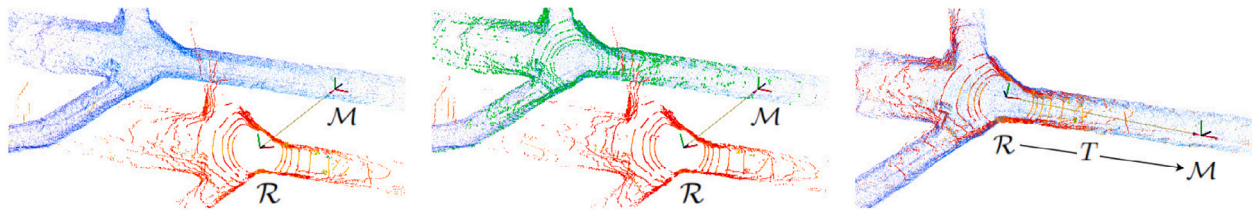
Fig. 1. (a) Initially, the LiDAR scan of the robot frame $\mathcal{R}$ and the point cloud map of the map frame $\mathcal{M}$ are not aligned. (b) During the re-localization process, the nearest submap based on the indexes of the vectors in the k-d tree, highlighted in green, is selected to find the transform T. (c) Finally, the LiDAR scan of the robot frame is aligned with the map frame.

1.1. Related work

Predominantly, the global re-localization problem consists of two stages: (a) place recognition, which identifies the frame in the map that is topologically close to the current frame, and (b) pose estimation, which calculates the relative pose from the map frame to the robot's current frame. Our proposed framework serves as a bridge between descriptor extraction for place recognition and global pose estimation within a prebuilt point cloud map. Therefore, we structure this section into discussions of related work concerning learned descriptors for place recognition and pose estimation, as well as available global re-localization solutions in a point cloud map.

1.1.1. Learned descriptors

Descriptors can be categorized as handcrafted (He, Wang, & Zhang, 2016; Kim & Kim, 2018), learned-based (Du et al., 2020; Dubé et al., 2018; Schaupp et al., 2019; Uy & Lee, 2018; Vidanapathirana, Ramezani, Moghadam, Sridharan, & Fookes, 2021), or hybrid (Vidanapathirana et al., 2020). Handcrafted methods have the advantage of not requiring re-training to adapt to different environments or platforms. While methods like the well-known ScanContext by Kim and Kim (2018) have demonstrated reliable performance across varying scenarios, the discriminatory ability of such methods remains limited. More recently, in the context of urban autonomous driving, Jiang and Shen (2023) introduced Contour Context, a novel approach for topological loop closure detection and accurate 3-DoF metric pose estimation. The proposed method leverages the layered distribution of structures within Cartesian birds' eye view (BEV) images, obtained from 3D LiDAR points. By extracting contour information from these images and assessing their geometric consistency and similarity, the approach achieves effective place recognition while optimizing relative transforms.

Learned-based methods have shown promising results with the universal approximation function properties of neural networks (Hornik, Stinchcombe, & White, 1989). In recent years, CNNs have become the state-of-the-art method for generating learning-based descriptors due to their ability to find complex patterns in data (Krizhevsky, Sutskever, & Hinton, 2012). PointNetVLAD, proposed by Uy and Lee (2018), pioneered the use of an end-to-end trainable global descriptor for 3D point cloud recognition. Extracted local features from PointNet (Qi, Su, Mo, & Guibas, 2016) are deployed to the NetVLAD aggregator (Arandjelović, Gronat, Torii, Pajdla, & Sivic, 2015) to form a global descriptor of the scene. SegMap (Dubé et al., 2018) employs CNNs to encode small-dimensional representations and decode them back to the original input, as part of its core modules: segment extraction, description, localization, map reconstruction, and semantic extraction, all contributing to 3D point cloud localization and mapping. LoGG3D-Net (Vidanapathirana et al., 2021), for the first time, addressed the limitations of first-order aggregation by introducing a training signal to the local features and using differentiable second-order pooling for global descriptor generation. High-order aggregation methods demonstrated superior performance in visual recognition (Koniusz, Yan, Gosselin, & Mikolajczyk, 2017; Li, Xie, Wang, & Zuo, 2017), previously applied to 3D place recognition (Vidanapathirana et al., 2020), though not in a

trainable architecture. In DH3D (Du et al., 2020), a hierarchical 3D descriptor learning approach was presented. A hierarchical network, operating directly on a point cloud, delivers local descriptors, a keypoint score map, and a global descriptor in a single forward pass. The success of deep learning is particularly noticeable in 2D images, where convolutional kernels can be easily applied to the 2D grid structure of the image. Schaupp et al. (2019), with OREOS, takes advantage of this success by projecting a 3D point cloud into spherical coordinates, yielding a 360-degree range image. The learned data-driven descriptors are then used for fetching the nearest neighbor place and estimating yaw discrepancy. Similarly to the preceding approaches, yet harnessing a diverse array of cues encompassing range, normals, intensity, and semantic classes, the OverlapNet framework by Chen et al. (2022) effectively exploits spherical images derived from point cloud data, resulting in a notable enhancement of its performance. Expanding upon this foundational work, the advanced OverlapTransformer (Ma et al., 2022) iteration introduces the integration of rotation-invariant features and expedited inference capabilities. Achieving this is facilitated through the incorporation of the attention mechanism derived from the Transformer (Vaswani et al., 2017) and the NetVLAD head (Arandjelović et al., 2015).

Finally, as the name suggests, hybrid methods aim to unite mathematical models with data-driven models to benefit from both (Shlezinger, Whang, Eldar, & Dimakis, 2021). First demonstrated by Locus (Vidanapathirana et al., 2020), an approach for LiDAR-based place recognition mathematically models 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, Lenz, Stiller, & Urtasun, 2013), it struggles to adapt to environments where the extracted segments are structurally different from its training data. With LocNet (Gidaris & Komodakis, 2015), Yin, Tang, Ding, Wang, and Xiong (2017) used semi-handcrafted range histogram features as input to a 2D Convolutional Neural Network (CNN), demonstrating the power of Deep Neural Networks (DNNs) to learn suitable data representations and exploit the most relevant cues in the input data.

1.1.2. Global re-localization

Currently, only a limited number of ROS packages support global re-localization in a 3D point cloud map. Koide, Miura, and Menegatti (2019) has provided a series of packages that include global re-localization as part of the localization and mapping process. The localization process employs an Unscented Kalman Filter-based pose estimation, fusing IMU and 3D LiDAR data. Subsequently, the scheme performs Normal Distribution Transform (NDT) scan matching between the global map and the input scan to correct the estimated pose. For global re-localization, it offers three engines: Branch and Bounce Search (BBS) (Hess, Kohler, Rapp, & Andor, 2016), FPFH+RANSAC (Buch, Kraft, Kamarainen, Petersen, & Kruger, 2013; Rusu, Blodow, & Beetz, 2009), or FPFH+Teaser++ (Yang, Shi, & Carlone, 2021).

In the case of LIO-SAM (Shan et al., 2020), re-localization based on LIO-SAM employs multiple factors, including IMU data, 3D LiDAR data, and the loop closure process, to jointly optimize the factor graph. This

Table 1

Comparison of the proposed framework against other available place-recognition solutions and re-localization frameworks.

	OREOS	PointNetVLAD	3DEG (OURS)	LIO-SAM	FAST-LIO	HDL Localization
Input format	Data sequences	Data sequences	PCD file	PCD file	PCD file	PCD file
Initial guess	Not required	Not required	Not required	Required	Required	Required
Yaw regression	✓	✗	✓	NA	NA	NA
Top-k candidates	✓	✓	✓	✗	✗	✗
Event-triggered	NA	NA	✓	✗	✗	✗

(NA: Not Applicable).

approach introduces key-frames and a sliding window scan-matching strategy, where new key-frames are selectively registered to a fixed-size set of prior sub-key-frames to enhance real-time performance. The most recent package developed is based on FAST-LIO (Xu & Zhang, 2021). It provides global pose estimation in a pre-built point cloud map by combining low-frequency global localization and high-frequency odometry. The feature extraction process involves extracting edge and planar features from the input 3D LiDAR scan, which, along with IMU measurements, are fed into the state estimation module.

1.2. Overview of the proposed approach

Our proposed re-localization framework, referred to as 3DEG, acts as a pivotal link between existing place recognition methods and available integrated re-localization solutions, as illustrated in Table 1. We address the limitations of conventional place recognition methods, which often depend on sequential data rather than a global point cloud map. Within the context of subterranean environments, we expand the applicability of our framework by introducing a modular architecture capable of adapting to specific environments based on their unique features. This adaptability is crucial to tackle challenges posed by subterranean settings, which include reduced lighting, confined spaces, and irregular structural elements. Furthermore, our framework overcomes the constraints of typical re-localization methods reliant on a single global point cloud map file as input. These methods typically necessitate a manual initial guess or suffer from extended computational times. In contrast, our framework eliminates the need for a manual initial guess and significantly reduces computational time. Of particular importance, our proposed approach acknowledges the critical role of junctions, a key factor in establishing a complete autonomous pipeline. Junctions play a pivotal role in autonomously triggering the re-localization process, ensuring seamless navigation in complex subterranean environments. Through these advancements and adaptations, our proposed re-localization framework offers a comprehensive and efficient solution. It combines the strengths of place recognition methods and integrated re-localization solutions while mitigating their respective limitations. This holistic approach contributes to improved accuracy, flexibility, and computational efficiency across various environments, ultimately leading to enhanced re-localization performance.

1.3. Contributions

The contributions of this work can be summarized as follows:

1. We present a comprehensive framework for re-localization within a given global 3D point cloud map. Our approach capitalizes on data-driven descriptors to facilitate efficient place recognition. Significantly, as depicted in Table 1, our framework integrates the capability to explore the top-k candidates, thereby augmenting the overall system's resiliency.
2. By conducting comprehensive real-world field experiments, we showcase the strong performance of our framework in achieving reliable re-localization within demanding subterranean environments. Notably, our approach sets itself apart by eliminating the requirement for a manual initial estimate, a feature that distinguishes it from other ROS-based packages utilizing comparable

input formats (Stanford Artificial Intelligence Laboratory et al., 2018), including relocalization methods like LIO-SAM (Shan et al., 2020), FAST-LIO (Xu & Zhang, 2021), and hdl global localization (Koide et al., 2019).

3. We present a novel modular architecture that empowers descriptors to adapt to specific events or tasks. This is accomplished by integrating a classification module, which enables the robot to autonomously initiate the re-localization process through event detection, such as recognizing a junction. This strategy enhances the probability of successful outcomes and extends the robot's exploration autonomy.
4. We showcase a data-handling process that facilitates the transition from large public datasets and place recognition solutions to learning from a limited amount of field mission data. Additionally, we propose a direct semantic global re-localization approach, which, to the best of the authors' knowledge, is absent from the current literature. We describe various techniques employed in this process, including joint training, negative mining, fine-tuning and label smoothing.

Overall, these contributions provide a comprehensive and innovative framework for re-localization, addressing key challenges in the field and demonstrating improved performance in challenging environments.

2. The proposed approach

The goal of this article is to introduce a global re-localization algorithm that is able to yield a rigid transform $T \in SE(3)$ so that the current robot frame $\mathcal{R}$ is transformed to the global map frame $\mathcal{M}$. Considering a robot r operating in $\mathbb{R}^3$ space, it generates 3D point cloud LiDAR scans, $\mathcal{P} \in \mathbb{R}^3$, with respect to the robot frame $\mathcal{R}$. Given a known point cloud map M in the global map frame $\mathcal{M}$ and its corresponding trajectory Tr, denoted as:

$$
\begin{aligned}
M &= \{m_1, m_2, \ldots, m_n\}, \\
Tr &= \{p_1, p_2, \ldots, p_k\},
\end{aligned}
\tag{1}
$$

where $m \in \mathbb{R}^3$ are sets of points and $p \in \mathbb{R}^3$ are sets of poses $p_k = (x_k, y_k, z_k)$, we are looking for the homogeneous rigid transformation of the special Euclidean group, defined as:

$$
T = \begin{bmatrix} R & \vec{p} \\ 0 & 1 \end{bmatrix} \in SE(3),
\tag{2}
$$

where $R \in SO(3)$ is the rotational matrix and $\vec{p} \in \mathbb{R}^3$ is the translational vector. To tackle this problem and acquire the transform T, we follow the steps depicted in Fig. 2, which can be summarized as: (a) Map Partitioning, (b) Point Cloud Projection, (c) Descriptor Extraction, (d) Initial Pose Estimation, and (e) Pose Refinement.

2.1. Map partitioning

To work with a point cloud map, we must partition it into individual scans. Let the point cloud map be denoted as M, and let the trajectory of the robot be represented as a sequence of n points, $\{p_1, p_2, \ldots, p_n\}$. We create a k-d tree database of the visited places to enable efficient search using descriptors. We partition M into n scans by transforming the point clouds according to the corresponding trajectory points. Let M_i
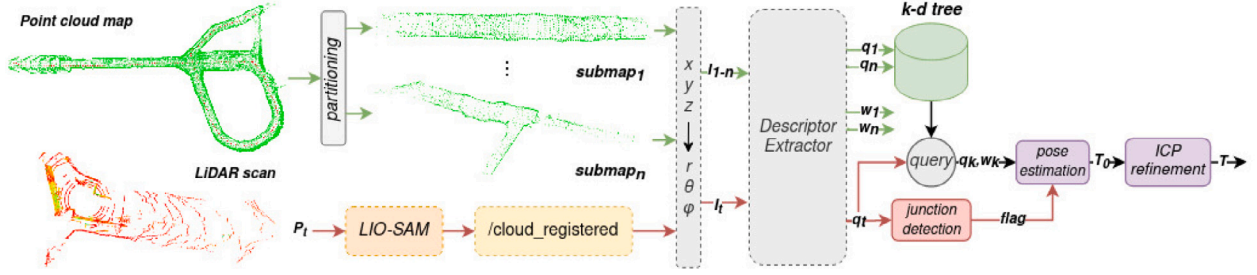
Fig. 2. The overall pipeline of the proposed architecture. The red arrows follow the workflow of the current scan $\mathcal{P}_t$, while the green arrows follow the workflow of the given point cloud map.

denote the point cloud map for the ith scan. We transform M according to p_i to obtain M_i, such that each point m in M_i is expressed with respect to the robot frame and not the map frame. Mathematically, we can write:

$$M_i = \{T_{p_i}(m) \mid m \in M\}, \tag{3}$$

where $T_{p_i}(m)$ denotes the transformation of point m according to p_i. 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.

2.2. Point cloud projection

The primary purpose of the Point Cloud Projection submodule is to convert the LiDAR point cloud scan data $\mathcal{P}_k$ at each time step k or each submap M_k, into a 2D depth image $\mathcal{I}_k$ using a spherical projection model. This transformation is accomplished by projecting a list of point coordinates p_x, p_y, and p_z onto a 2D spherical grid, as illustrated in Fig. 3. The pixel value of each point in the grid is determined by its range ρ from the sensor's frame, as described by the equations:

$$\phi = atan(\frac{p_x}{p_y})$$
$$\rho = asin(\frac{p_x^2 + p_y^2}{\sqrt{p_x^2 + p_y^2 + p_z^2}}) \tag{4}$$

Converting LiDAR point cloud scans into range images has significant advantages over using raw point cloud data. The panoramic 360° view obtained from range images enables the production of orientation-invariant descriptors that can be used for various applications, including object detection and classification. Using a 2D CNN on range images can be particularly beneficial because of their computational efficiency and ability to handle large datasets. Additionally, 2D CNNs are suitable for capturing translational invariance in the data, which is useful for detecting and classifying features from different viewpoints. However, there are some limitations to using range images. One of the main disadvantages is that they capture a less dense view of the surroundings compared to depth sensors, which can make it challenging to extract detailed features from converted range images, especially for tasks that require high accuracy and precision.

2.3. Descriptor extraction

The Descriptor Extraction module is responsible for generating a concise representation of the surrounding topological characteristics, which includes place recognition, orientation regression, and classification information. In this study, we employ a Convolutional Neural Network (CNN) that takes 2D range images, $\mathcal{I}_k$, as input and produces

2×64 vector sets, namely $\vec{q}$ and $\vec{w}$, which will be further discussed in Section 3 and are denoted as:

$$Q = \{\vec{q}_1, \vec{q}_2, \dots, \vec{q}_n : \vec{q}_i \in \mathbb{R}^{64}\}, \tag{5}$$
$$W = \{\vec{w}_1, \vec{w}_2, \dots, \vec{w}_n : \vec{w}_i \in \mathbb{R}^{64}\} \tag{6}$$

The vector $\vec{q}$ captures orientation-invariant information that is specific to each place, while the vector $\vec{w}$ serves as a compact representation of rotation-variant information. The latter is utilized for estimating the yaw discrepancy in a subsequent stage of the pipeline (Schaupp et al., 2019). To build a comprehensive understanding of the environment, we utilize the place-specific vectors Q, generated from the partitioned map scans M_i. By employing the aforementioned approach, we extract essential information from the topological characteristics, allowing for effective place recognition, orientation regression, and classification tasks.

2.4. Initial pose estimation

In the Initial Pose Estimation module, we make use of the aforementioned descriptive vectors to construct a k-d tree. Subsequently, with the current scan from the robot $\mathcal{P}_t$, we predict the vector $\vec{q}_t$ and query the k-d tree to identify nearby potential places. The querying process tries to identify the vector $\vec{q}_i$ from the vector set Q that has the minimum distance from the vector $\vec{q}_t$, in the vector space. The process is denoted as:

$$i = \arg\min_{i \in \mathbb{N}} f(q_i, q_t) \text{ where } q_i \in Q, \tag{7}$$

where $f(a, b)$ is a function that returns the Euclidean distance between two multidimensional vectors a and b, described as:

$$f(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_n - b_n)^2} = \|a - b\| \tag{8}$$

Once the k-d tree has been queried with the current vector $\vec{q}_t$, we obtain the indexes of the nearest neighbors, along with the corresponding vectors $\vec{q}_i$ and $\vec{w}_i$ from the vector set Q. This retrieval allows us to obtain the trajectory points associated with these neighbors, facilitating the acquisition of the initial translation vector $\vec{p}_0$. The top-k indexes of the nearest neighbors are returned from the search, and the index corresponding to the minimum distance is selected as the primary candidate for re-localization. This choice is based on the assumption that the nearest neighbor with the least distance is more likely to yield an accurate re-localization result. However, it is important to note that in cases where the first candidate fails to provide the desired outcome, the remaining top-k indexes can be utilized as alternative candidates, allowing for resiliency in cases of challenging localization scenarios. The subsequent step involves feeding the orientation estimation module with the rotation variant vectors $\vec{w}_t$ and $\vec{w}_i$. This module is responsible for estimating the yaw discrepancy $\delta\theta$ between the query point cloud and the nearest retrieved candidate from the partitioned map. With the calculated $\delta\theta$, we can construct the rotation matrix $R_0(\delta\theta)$, which represents the initial estimation. Combining this rotation matrix with the
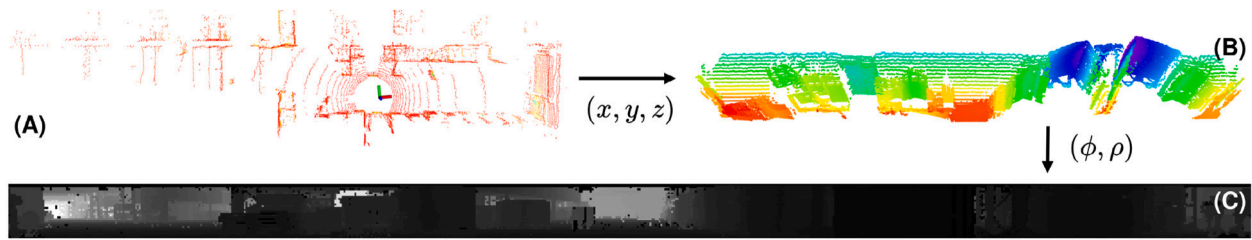
Fig. 3. (A) The original LiDAR scan $\mathcal{P}k$. (B) The result of the spherical projection. (C) The depth image $\mathcal{I}k$ obtained by mapping the range of each point to its corresponding image pixel.

initial translation vector $\vec{p}_0$, we obtain the complete initial estimation denoted as:

$$T_0 = \begin{bmatrix} R_0 & p_0 \\ 0 & 1 \end{bmatrix} \in SE(3) \tag{9}$$

This process allows us to align the query point cloud with the nearest candidate from the partitioned map, providing an initial estimation that incorporates both translation and rotation information.

2.5. Pose refinement

Using the obtained initial pose estimation, we can enhance the accuracy of the pose by utilizing it as a prior condition in the registration algorithm, Iterative Closest Point (ICP) (Zhang, 2014). This algorithm iteratively aligns the query point cloud with the nearest candidate from the partitioned map, progressively refining the pose estimation. Additionally, the distance between the two vectors $\vec{q}_t$ and $\vec{q}_k$ can be leveraged to establish a distance threshold for the registration method. By incorporating this threshold, we can aid the registration process by discarding potential matches that exceed the threshold. By integrating the initial pose estimation and utilizing the distance information between the vectors, we can achieve a more accurate and efficient refinement of the pose estimation through the ICP registration algorithm.

2.6. Event-based triggering

In order to enhance the autonomy of our robots during field exploration missions, we utilize the classification module to identify instances when the robot reaches a junction. This classification capability serves as a crucial trigger for the global re-localization process, particularly in scenarios involving multi-robot exploration, where multiple robots need to share the same map (Stathoulopoulos, Koval, Agha-mohammadi, & Nikolakopoulos, 2023). By providing the current vector $\vec{q}_t$ as input, the classifier is capable of distinguishing between different configurations, such as a straight tunnel, a junction, or a turn. The event of reaching a junction serves as the trigger point for initiating the global re-localization process. This is due to the fact that junctions tend to exhibit more distinctive features and offer a higher likelihood of success in accurately re-establishing the robot's global position within the environment.

3. 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 Appalaraju and Chaoji (2017), Simonyan and Zisserman (2014), as well as our own proposed addition for driving learned descriptors to certain features.

3.1. 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 Fig. 4. 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 $2 \times 64 \times 1$ vectors, $\vec{q}$ and $\vec{w}$ respectively. As mentioned before, the vector $\vec{q}$ encodes place dependent information and is invariant to orientation changes, while $\vec{w}$ 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 $\vec{w}$ 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 $\vec{q}$ and depending on the mission, the *classifier* can be trained on detecting other characteristics, for example pipelines or shafts.

3.2. 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 $\vec{q}$ and $\vec{w}$. These orthogonal vectors are crucial for achieving distinct objectives within the subsequent modules of the system. The *orientation estimation* module estimates the yaw difference from two compact vectors, $\vec{w}$, ensuring that this vector's orientation-specific information is separated from other components. Similarly, the *classifier* has the goal of predicting the correct class based on a descriptor vector $\vec{q}$. This concept of vector orthogonality, as introduced in the work by Schaupp et al. (2019), emerges due to the divergent objectives pursued by the two vectors, each striving for distinct and independent goals. Specifically, one vector is designed to be place-dependent while remaining orientation-invariant, whereas the other vector is orientation-specific yet place-independent. The ensuing orthogonal relationship is established as a consequence of these differing design principles. 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. The choice of orthogonal vectors aligns well with the principle that orthogonal vectors often arise when capturing different aspects or features that are independent of each other. In our case, $\vec{q}$ and $\vec{w}$ represent distinct sets of features, each encoding unique characteristics of the data. This design choice ensures that the vectors do not overlap in their information content, preventing redundancy and correlation. By making sure that these vectors are orthogonal, we enhance the ability of your system to focus on the specific aspects relevant to each module's task, thus contributing to the overall efficiency and effectiveness of your approach.
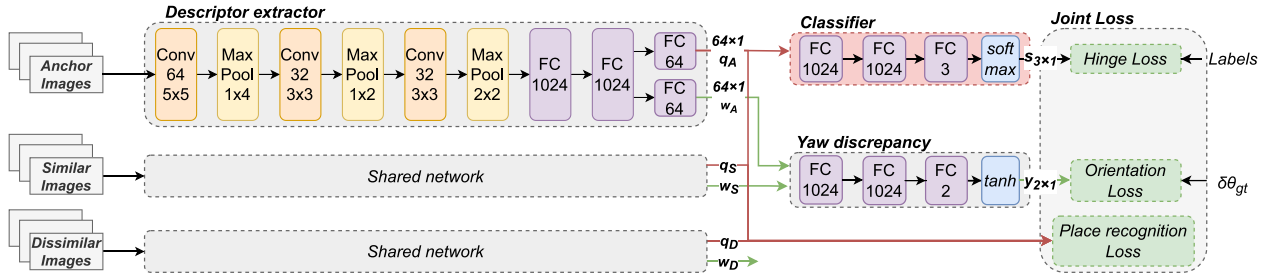
Fig. 4. The Siamese network consists of three convolutional layers followed by max pooling and then two fully connected layers, as proposed by Schaupp et al. (2019). The number of filters and the kernel size have been adjusted to the considered testing environments and sensor utilized, and in addition we have an extra classifying module that boosts the descriptor extraction process.

Starting with L_{pr}, in order to train our network for the task of place recognition, we use the triplet loss method (Hoffer & Ailon, 2014). As demonstrated in Fig. 4, we feed the neural network with three types of images: anchor images, which serve as the reference points for comparison, similar images to the anchor images, and dissimilar images, denoted as I_A, I_S and I_D respectively. These anchor images provide a fixed foundation for evaluating the similarities and differences between the other image pairs, playing a crucial role in our approach for descriptor extraction and subsequent analysis. We also define d_S as the Euclidean distances between the descriptors $\vec{q}_A$ from I_A and the descriptors $\vec{q}_S$ from I_S. The same applies for d_D and the descriptors $\vec{q}_D$ 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:

$$L_{pr}(d_S, d_D) = \underbrace{\|f(I_A) - f(I_S)\|^2}_{d_S} - \underbrace{\|f(I_A) - f(I_D)\|^2}_{d_D} + m \qquad (10)$$

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 $\vec{w}_A$ and $\vec{w}_S$. The orientation loss is defined as:

$$L_\theta(y_{yaw}, \delta\theta_{gt}) = \frac{1}{2}((y_{yaw,0} - \cos(\delta\theta_{gt}))^2 \\ + (y_{yaw,1} - \sin(\delta\theta_{gt}))^2) \qquad (11)$$

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.

$$L_c(s_j, s_{l_i}) = \sum_{j \neq s_i} \max(0, s_j - s_{l_i} + 1) \qquad (12)$$

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 jth class label (the incorrect class) and the l_i-th class (the correct class).

3.3. Training the descriptors

In the training process of our network, we employ a technique known as joint training. Joint training involves simultaneously optimizing the weights and parameters of multiple interconnected neural network modules. This approach is beneficial in machine learning training for several reasons. Firstly, joint training facilitates the creation of synergistic relationships between different modules, enabling them to collectively learn and adapt to complex patterns and interactions present in the data. By training the entire network holistically, information and insights obtained from one module can be shared and utilized

by other modules, leading to improved overall performance. Moreover, joint training promotes the development of a cohesive and unified representation of the input data. As different modules influence each other's learning processes, the network can more effectively capture intricate relationships and dependencies within the data, resulting in a more comprehensive understanding of the underlying patterns. Additionally, joint training can help mitigate issues related to overfitting, as the modules are optimized together, they collectively strive for a balanced solution that generalizes well to unseen data, reducing the risk of overfitting to individual modules. This approach also encourages the network to learn more discriminative and transferable features, enhancing its ability to handle various scenarios and data variations. In the context of our specific network, joint training enables us to achieve multiple objectives simultaneously, namely, enhancing localization recall, achieving precise yaw angle estimation, and ensuring robust classification performance. By jointly optimizing the weights of the three Neural Networks, we create a unified learning framework that leverages the strengths of each module while fostering synergies between them. This holistic optimization approach contributes to the effectiveness and efficiency of our network's performance across various tasks. For this, we combine all three loss functions, defined as L:

$$L = L_{pr} + L_\theta + L_c \qquad (13)$$

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, $(\vec{q}_A, \vec{w}_A), (\vec{q}_S, \vec{w}_S)$ and $(\vec{q}_D, \vec{w}_D)$. The three place dependent vectors are used to compute the L_{pr} loss, while $\vec{w}_A$ and $\vec{w}_S$ 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 $\vec{q}_A$ 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 & Ba, 2014) learning optimizer the weights are updated.

4. 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.
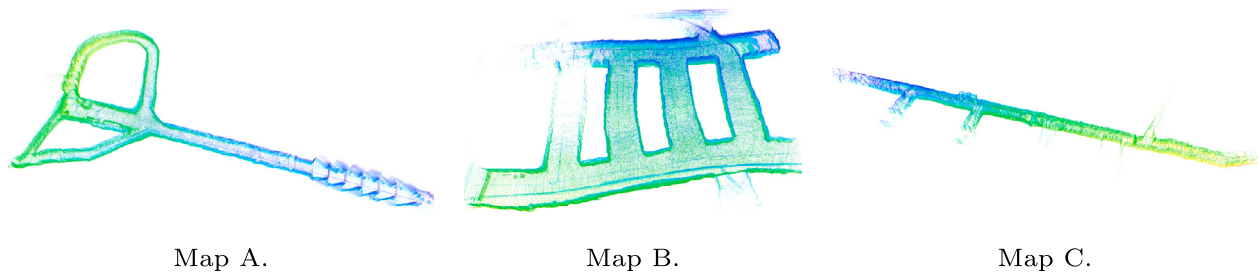
Map A.　　　　　　　Map B.　　　　　　　Map C.

Fig. 5. 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.

4.1. Neural network evaluation

4.1.1. Datasets

For the training and evaluation process, we use three dataset collections. The first dataset collection (Koval, Karlsson, Mansouri, et al., 2022) contains recordings from an underground tunnel located in Luleå, Sweden, as seen in Fig. 5. 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, Karlsson, & Nikolakopoulos, 2022), that includes the Velodyne VLP16 PuckLite 3D LiDAR, the Vectornav VN-100 IMU sensor and an Intel NUC on-board computer. The NUC has an Intel Core i5-10210U, 4-core processor with Intel UHD Graphics and 8 GB of RAM. The second robotic platform is a custom-built quadrotor (Lindqvist, Kanellakis, et al., 2022) 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 cumulative distance covered by the robots within the mapped environment spans approximately 0.5 km, signifying a considerable exploration effort. This extensive traversal yielded a comprehensive representation composed of an approximate total of $3 \cdot 10^5$ points, capturing the features and layout of the environment. The subsequent dataset, represented as Map B in Fig. 5, stems from an authentic subterranean mining site. In contrast to the initial dataset, this collection encompasses more expansive tunnels, reaching widths of up to 10 m. Within this environment, multiple intricate junctions coexist amidst a relatively featureless setting, presenting a distinct set of challenges for robotic exploration and mapping. Spanning an area of 0.5 km, this environment contributes to the creation of a map containing an estimated $1.6 \cdot 10^6$ points. The intricate complexities of the larger tunnels and junctions further augment the richness of the mapped data. This dataset serves as a valuable resource for evaluating the efficacy and robustness of our proposed approach within diverse subterranean settings, highlighting its adaptability to a spectrum of real-world environments. The third and last dataset collection is from the same underground tunnel as the first one but from a different passage, and it is depicted on Fig. 5 as Map C. Within this particular context, the environment takes on a more constrained configuration, resembling a corridor that is both narrower and more linear in nature. This corridor stretches across a length of approximately 150 m, characterized by its distinct geometric attributes and confined spatial characteristics. In terms of data density, the environment contributes a collection of data points totaling approximately $1.2 \cdot 10^5$. The intricacies of this corridor-like setting pose unique challenges for mapping and navigation, allowing us to examine the adaptability and effectiveness of our proposed approach in constrained and specialized subterranean scenarios. 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 handcrafted on all datasets. It

is important to highlight the process behind acquiring the maps A, B, and C showcased in Fig. 5, as these maps lay the foundation for the subsequent re-localization process in our experiments. In both scenarios, the generation of these maps relied on the utilization of LIO-SAM (Shan et al., 2020), a Simultaneous Localization and Mapping (SLAM) algorithm. SLAM is a fundamental capability that empowers autonomous systems to concurrently estimate their own position (localization) and construct a map of the environment they are navigating. LIO-SAM leverages both 3D LiDAR scans and IMU measurements to perform this dual task. In the case of Map A, manual control was employed to guide the robot and build the map, ensuring a meticulously crafted initial representation. Map B, on the other hand, is a product of merging multiple maps generated from distinct autonomous missions. This map merging process was facilitated by FRAME (Stathoulopoulos et al., 2023), a map-merging algorithm. Lastly, the creation of the final map C was orchestrated through the autonomy framework COMPRA (Lindqvist, Kanellakis, et al., 2022). COMPRA facilitated the autonomous exploration of the tunnel, enabling the robot to venture forth, comprehensively map the environment, and safely return. In essence, SLAM plays a pivotal role in generating accurate maps that serve as the bedrock for subsequent localization and navigation tasks. It allows to concurrently estimate the robot's location and construct a comprehensive map of its surroundings. The combined efforts of SLAM algorithms and autonomous frameworks contribute significantly to the reliable and informed navigation of robotic systems within complex environments.

4.1.2. Data sampling and training process

As mentioned in Section 3, the neural network is based on the triplet network architecture and therefore requires sampling three tuples of anchor, similar and dissimilar pairs. We consider two point clouds as similar, if their ground-truth poses, defined as p, are within 3 m, $|p_A - p_S| \leq 3$. In addressing the dissimilar pairs, our approach employs a strategy known as hard-negative mining, to enhance the performance of our network. This technique focuses on selecting dissimilar pairs, which consist of point clouds that are not related to each other. To achieve this, we employ a two-stage negative mining strategy. In the initial stage, we randomly sample point clouds that are beyond a 3-m radius from each other ($|p_A - p_D| \geq 3$), ensuring a significant level of dissimilarity between the selected pairs. As the training process advances, we move on to the later stage, where we sample point clouds within a radius of 3 to 6 meters from the anchor ($3 \leq |p_A - p_D| \leq 6$). The essence of this strategy lies in the concept of "hard-negative mining", a concept introduced by Bucher, Herbin, and Jurie (2016). By introducing progressively more challenging negative samples during the later stages of training, the network is exposed to point cloud pairs that are difficult to distinguish, pushing its boundaries and honing its ability to discern subtle differences. This process of incrementally introducing harder-to-distinguish triplets helps the network adapt and improve its performance in the advanced phases of convergence. The benefits of negative mining are twofold. Firstly, it exposes the network to a wider range of training examples, helping it learn from diverse scenarios and enhancing its generalization capabilities. Secondly, by

focusing on challenging triplets that are initially challenging to differentiate, the network becomes more robust and capable of handling complex and intricate variations in the input data. This strategic integration of negative mining contributes significantly to the network's ability to achieve higher performance levels and better representation learning. Given the constrained availability of datasets from subterranean environments, we adopt a strategic approach by utilizing data solely from the initial dataset collection for neural network training. Achieving a balanced dataset is crucial, particularly during the training process of the *classifier* module. However, the inherent nature of the dataset introduces a bias toward straight corridors due to the environmental conditions. To address this issue, we adopt the concept of fine-tuning, which proves beneficial for enhancing the network's ability to generalize across diverse scenarios. To commence this fine-tuning process, we initiate training without the inclusion of the classifying module. This initial phase allows the network to learn essential features and patterns inherent in the subterranean environments, undisturbed by the classification task. Following this, we strategically resample the dataset to achieve balance among the three distinct classes: (a) straight corridors, (b) junctions, and (c) turns. This balanced dataset enables the network to learn from a diverse set of scenarios, capturing the subtleties associated with each class. Subsequently, we initiate the fine-tuning process, this time enabling the classifying module. By fine-tuning with the balanced dataset, the network refines its understanding of different classes, effectively adapting its learned features to align with the intricacies of subterranean corridors, junctions, and turns. This fine-tuning approach ensures that the *classifier* becomes adept at identifying and classifying different environmental configurations, thereby enhancing the overall accuracy and robustness of our network's performance in real-world scenarios with limited data availability. Furthermore, given the inherent challenge of accurately determining the initiation and termination points of junctions, we employ a regularization technique known as label smoothing (Goodfellow, Bengio, & Courville, 2016; Müller, Kornblith, & Hinton, 2019) to refine our training process. Label smoothing entails modifying the target labels used in training to be more softly distributed, rather than using the conventional binary 0 and 1 labels.

$$labels_{new} = labels_{old} \cdot (1 - a) + \frac{a}{N} \qquad (14)$$

This technique serves as a powerful tool to address the issues of overfitting and excessive confidence that may arise in the *classifier* module. The mechanism of label smoothing, as described by Eq. (14), involves transforming the original hard labels ($labels_{old}$) by redistributing a portion ($\frac{a}{N}$) of the label probability mass uniformly among all classes. The parameter a governs the extent of this redistribution, while N represents the number of classes. By introducing this smoothing process, the classifier is encouraged to exhibit more cautious decision-making, mitigating the risk of extreme overconfidence in its predictions. By embracing soft labels that encapsulate the uncertainty and ambiguity inherent in labeling complex features, we foster a more balanced and adaptive learning process. In essence, when the classifier makes an

incorrect prediction, the use of soft labels results in a reduced loss compared to the conventional binary case. Consequently, the model learns from its mistakes in a more gradual manner, leading to a refined understanding of the intricate features and patterns involved. This integration of label smoothing significantly contributes to enhancing the generalization capacity of the model and its performance in scenarios with inherently complex and uncertain decision boundaries.

4.1.3. 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 in Fig. 6 and in Table 2, 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 junctions, the junction detection module provides a significant boost on the top-1, with more than 10%. The results of Table 2, 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°–25°), 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° 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.

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 2, we only present them for 3DEG since the final estimation is performed by the ICP registration.

4.2. 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 1.1. In Table 3, 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
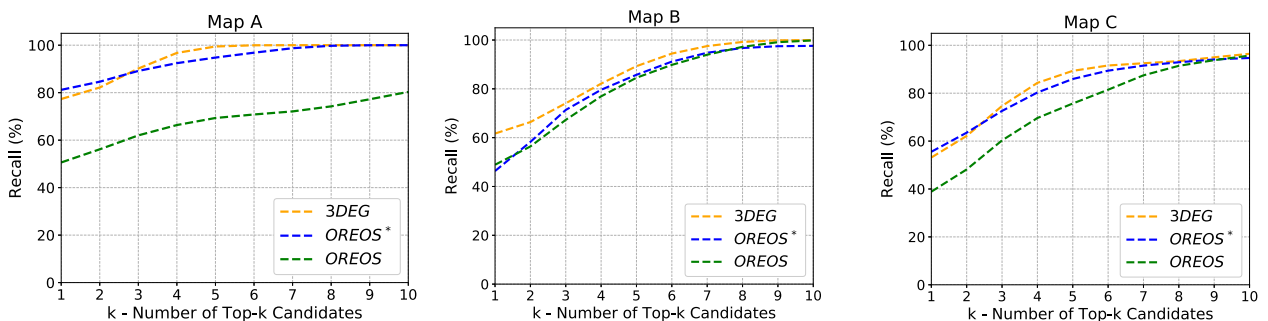

Fig. 6. Performance comparison of the proposed framework 3DEG, our implementation of OREOS, as described by Schaupp et al. and our implementation of OREOS but with adjusted network parameters as seen in Fig. 4, denoted as OREOS*.

Table 2

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.

	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 3

Experimental evaluation of the available re-localization packages, from Map A of Fig. 5, with comparison of the average computational time, CPU load, memory usage and mean error.

	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**

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, Yao, & Deng, 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.

5. 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 the implementation code should be optimized, which for the moment is not optimal and highly affects the runtime and memory usage of the algorithm.

6. Conclusions

In this article, we have introduced the 3DEG framework, which offers a comprehensive solution for global re-localization in a 3D point cloud map setting. This novel framework leverages data-driven descriptors and is designed to autonomously initiate the re-localization process upon detecting a junction within the environment. By incorporating this junction-based triggering mechanism, our framework provides an effective means to address re-localization challenges. One significant aspect of our proposed framework is its ability to provide resiliency through the inclusion of multiple candidates. This semi-autonomous operation enhances the success rate of critical missions such as search and rescue. By offering multiple candidate options during the re-localization process, our framework increases the robustness and reliability of the

system, allowing it to adapt to various scenarios and overcome potential failures. Overall, the primary objective of this paper is to present a comprehensive re-localization pipeline specifically designed for challenging tunnel environments. Our framework stands out from existing conventional methods, which often rely solely on place recognition or pose estimation approaches that may struggle in such demanding settings. Through our proposed 3DEG framework, we aim to provide an effective and reliable solution for global re-localization in challenging tunnel environments.

CRediT authorship contribution statement

Nikolaos Stathoulopoulos: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Visualization. **Anton Koval:** Conceptualization, Investigation, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition. **George Nikolakopoulos:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data

Acknowledgments

References

Agha, A., Otsu, K., Morrell, B., Fan, D., Thakker, R., Santamaria-Navarro, A., et al. (2022). Nebula: Team costar's robotic autonomy solution that won Phase II of DARPA Subterranean challenge. *Field Robotics*, *2*(1), 1432–1506. http://dx.doi.org/10.55417/fr.2022047.

Appalaraju, S., & Chaoji, V. (2017). Image similarity using deep CNN and curriculum learning. http://dx.doi.org/10.48550/ARXIV.1709.08761, arXiv.

Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., & Sivic, J. (2015). NetVLAD: CNN architecture for weakly supervised place recognition. http://dx.doi.org/10.48550/ARXIV.1511.07247, arXiv.

Buch, A. G., Kraft, D., Kamarainen, J.-K., Petersen, H. G., & Kruger, N. (2013). Pose estimation using local structure-specific shape and appearance context. In *2013 IEEE international conference on robotics and automation*. IEEE, http://dx.doi.org/10.1109/icra.2013.6630856.

Bucher, M., Herbin, S., & Jurie, F. (2016). Hard negative mining for metric learning based zero-shot classification. CoRR, abs/1608.07441. arXiv:1608.07441.

Chen, X., Läbe, T., Milioto, A., Röhling, T., Behley, J., & Stachniss, C. (2022). Overlap-Net: a siamese network for computing LiDAR scan similarity with applications to loop closing and localization. *Autonomous Robots*, *46*, http://dx.doi.org/10.1007/s10514-021-09999-0.

Du, J., Wang, R., & Cremers, D. (2020). DH3D: Deep hierarchical 3D descriptors for robust large-scale 6DoF relocalization. http://dx.doi.org/10.48550/ARXIV.2007.09217, arXiv.

Dubé, R., Cramariuc, A., Dugas, D., Nieto, J., Siegwart, R., & Cadena, C. (2018). SegMap: 3D segment mapping using data-driven descriptors. In *Robotics: Science and systems XIV*. Robotics: Science and Systems Foundation, http://dx.doi.org/10.15607/rss.2018.xiv.003.

Dynamics, B., & Robotics, C. (2023). Spot, the agile robot, spot ROS package. https://www.bostondynamics.com/products/spot, https://clearpathrobotics.com/spot-robot/.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*.

Gidaris, S., & Komodakis, N. (2015). LocNet: Improving localization accuracy for object detection. http://dx.doi.org/10.48550/ARXIV.1511.07763, arXiv.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

He, L., Wang, X., & Zhang, H. (2016). 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* (pp. 231–237). http://dx.doi.org/10.1109/IROS.2016.7759060.

Hess, W., Kohler, D., Rapp, H., & Andor, D. (2016). Real-time loop closure in 2D LIDAR SLAM. In *2016 IEEE international conference on robotics and automation* (pp. 1271–1278). http://dx.doi.org/10.1109/ICRA.2016.7487258.

Hoffer, E., & Ailon, N. (2014). Deep metric learning using triplet network. http://dx.doi.org/10.48550/ARXIV.1412.6622, arXiv.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *2*, (5), (pp. 359–366). http://dx.doi.org/10.1016/0893-6080(89)90020-8,

Jiang, B., & Shen, S. (2023). Contour context: Abstract structural distribution for 3D LiDAR loop detection and metric pose estimation. In *2023 IEEE international conference on robotics and automation* (pp. 8386–8392). http://dx.doi.org/10.1109/ICRA48891.2023.10160337.

Kim, G., & Kim, A. (2018). Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map. In *2018 IEEE/RSJ international conference on intelligent robots and systems* (pp. 4802–4809). http://dx.doi.org/10.1109/IROS.2018.8593953.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. http://dx.doi.org/10.48550/ARXIV.1412.6980, arXiv.

Koide, K., Miura, J., & Menegatti, E. (2019). A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement. *16*, http://dx.doi.org/10.1177/1729881419841532,

Kominiak, D., Mansouri, S. S., Kanellakis, C., & Nikolakopoulos, G. (2020). MAV development towards navigation in unknown and dark mining tunnels. In *2020 28th mediterranean conference on control and automation* (pp. 1015–1020). IEEE.

Koniusz, P., Yan, F., Gosselin, P.-H., & Mikolajczyk, K. (2017). Higher-order occurrence pooling for bags-of-words: Visual concept detection. *39*, (2), (pp. 313–326). http://dx.doi.org/10.1109/TPAMI.2016.2545667,

Koval, A., Karlsson, S., Mansouri, S. S., Kanellakis, C., Tevetzidis, I., Haluska, J., et al. (2022). Dataset collection from a SubT environment. *Robotics and Autonomous Systems*, Article 104168. http://dx.doi.org/10.1016/j.robot.2022.104168, URL https://www.sciencedirect.com/science/article/pii/S0921889022000951.

Koval, A., Karlsson, S., & Nikolakopoulos, G. (2022). Experimental evaluation of autonomous map-based spot navigation in confined environments. *Biomimetic Intelligence and Robotics*, Article 100035.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, & K. Weinberger (Eds.), *Advances in neural information processing systems, vol. 25*. Curran Associates, Inc.

Li, P., Xie, J., Wang, Q., & Zuo, W. (2017). Is second-order information helpful for large-scale visual recognition? In *2017 IEEE international conference on computer vision* (pp. 2089–2097). http://dx.doi.org/10.1109/ICCV.2017.228.

Lindqvist, B., Kanellakis, C., Mansouri, S. S., Agha-mohammadi, A.-A., & Nikolakopoulos, G. (2022). COMPRA: A COMPact reactive autonomy framework for Subterranean MAV based Search-And-Rescue operations. *Journal of Intelligent and Robotic Systems*, *105*(3), 49.

Lindqvist, B., Karlsson, S., Koval, A., Tevetzidis, I., Haluška, J., Kanellakis, C., et al. (2022). Multimodality robotic systems: Integrated combined legged-aerial mobility for subterranean search-and-rescue. *Robotics and Autonomous Systems*, *154*, Article 104134. http://dx.doi.org/10.1016/j.robot.2022.104134.

Ma, J., Zhang, J., Xu, J., Ai, R., Gu, W., & Chen, X. (2022). OverlapTransformer: An efficient and yaw-angle-invariant transformer network for LiDAR-based place recognition. *IEEE Robotics and Automation Letters*, *7*(3), 6958–6965. http://dx.doi.org/10.1109/LRA.2022.3178797.

Müller, R., Kornblith, S., & Hinton, G. E. (2019). When does label smoothing help? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems, vol. 32*. Curran Associates, Inc.

Nikolakopoulos, G., & Agha, A. (2021). Pushing the limits of autonomy for enabling the next generation of space robotics exploration missions. *Computer*, *54*(11), 100–103.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2016). PointNet: Deep learning on point sets for 3D classification and segmentation. http://dx.doi.org/10.48550/ARXIV.1612.00593, arXiv.

Rusu, R., Blodow, N., & Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. In *Proc IEEE int conf robot automation* (pp. 3212–3217). http://dx.doi.org/10.1109/ROBOT.2009.5152473.

Schaupp, L., Bürki, M., Dube, R., Siegwart, R., & Cadena, C. (2019). OREOS: Oriented recognition of 3D point clouds in outdoor scenarios. In *2019 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, http://dx.doi.org/10.1109/iros40897.2019.8968094.

Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., & Rus, D. (2020). Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ international conference on intelligent robots and systems* (pp. 5135–5142). IEEE.

Shlezinger, N., Whang, J., Eldar, Y. C., & Dimakis, A. G. (2021). Model-based deep learning: Key approaches and design guidelines. In *2021 IEEE data science and learning workshop* (pp. 1–6). http://dx.doi.org/10.1109/DSLW51110.2021.9523403.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. http://dx.doi.org/10.48550/ARXIV.1409.1556, arXiv.

Stanford Artificial Intelligence Laboratory, et al. (2018). Robotic operating system. URL https://www.ros.org.

Stathoulopoulos, N., Koval, A., Agha-mohammadi, A.-a., & Nikolakopoulos, G. (2023). FRAME: Fast and robust autonomous 3D point cloud map-merging for egocentric multi-robot exploration. In *2023 IEEE international conference on robotics and automation* (pp. 3483–3489). http://dx.doi.org/10.1109/ICRA48891.2023.10160771.

Uy, M. A., & Lee, G. H. (2018). PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. http://dx.doi.org/10.48550/ARXIV.1804.03492, arXiv.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 6000–6010). Red Hook, NY, USA: Curran Associates Inc..

Vidanapathirana, K., Moghadam, P., Harwood, B., Zhao, M., Sridharan, S., & Fookes, C. (2020). Locus: LiDAR-based place recognition using spatiotemporal higher-order pooling. http://dx.doi.org/10.48550/ARXIV.2011.14497, arXiv.

Vidanapathirana, K., Ramezani, M., Moghadam, P., Sridharan, S., & Fookes, C. (2021). LoGG3D-net: Locally guided global descriptor learning for 3D place recognition. http://dx.doi.org/10.48550/ARXIV.2109.08336, arXiv.

Xu, W., & Zhang, F. (2021). FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter. github: https://github.com/HViktorTsoi/FAST_LIO_LOCALIZATION.

Yang, H., Shi, J., & Carlone, L. (2021). TEASER: Fast and certifiable point cloud registration. *37*, (2), (pp. 314–333). http://dx.doi.org/10.1109/TRO.2020.3033695.

Yin, H., Tang, L., Ding, X., Wang, Y., & Xiong, R. (2017). LocNet: Global localization in 3D point clouds for mobile vehicles. http://dx.doi.org/10.48550/ARXIV.1712.02165, arXiv.

Zhang, Z. (2014). Iterative closest point (ICP). In K. Ikeuchi (Ed.), *Computer vision: A reference guide* (pp. 433–434). Boston, MA: Springer US, http://dx.doi.org/10.1007/978-0-387-31439-6_179.

Zhang, J., Yao, Y., & Deng, B. (2021). Fast and robust iterative closest point. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1. http://dx.doi.org/10.1109/tpami.2021.3054619.