



Towards Semi-autonomous Robotic Inspection and Mapping in Confined Spaces with the EspeleoRobô

Héctor Azpúrua^{1,2} · Adriano Rezende³ · Guilherme Potje¹ · Gilmar Pereira da Cruz Júnior³ · Rafael Fernandes³ · Víctor Miranda³ · Levi Wellington de Resende Filho² · Jacó Domingues² · Filipe Rocha² · Frederico Luiz Martins de Sousa² · Luiz Guilherme Dias de Barros² · Erickson R. Nascimento¹ · Douglas G. Macharet¹ · Gustavo Pessin² · Gustavo M. Freitas³

Received: 2 July 2020 / Accepted: 18 January 2021 / Published online: 25 March 2021
© The Author(s), under exclusive licence to Springer Nature B.V. 2021

Abstract

Autonomous mobile devices operating in confined environments, such as pipes, underground tunnel systems, and cave networks, face multiple open challenges from the robotics perspective. Those challenges, such as mobility, localization, and mapping in GPS denied scenarios, are receiving particular attention from the academy and industry. One example is the Brazilian mining company Vale S.A., which is employing a robot – EspeleoRobô (SpeleoRobot) – to access restricted and dangerous areas for human workers. The EspeleoRobô is a robot initially designed for natural cave inspection during teleoperated missions. It is now being used to monitor other types of confined environments, such as dam galleries and other restrained or dangerous areas. This paper describes the platform in its current version and the pipeline used for semi-autonomous inspection in confined environments. The pipeline includes photorealistic mapping techniques, Simultaneous Localization and Mapping (SLAM) with LiDAR, path planning based on mobility optimization, and navigation control using vector fields to reduce operator dependency of the robot operation. The proposed concept was validated in simulations with a realistic underground tunnel system and in representative real-world scenarios. The results endorse the viability of using the proposed concept for real deployments.

Keywords Subterranean exploration with mobile robots · 3D reconstruction and mapping · GPS-denied localization · path planning in rugged terrains · Vector field control

1 Introduction

Exploration of confined and risk areas is a dangerous task, even for specialized personnel. Industrial maintenance teams need to perform routine inspections of structures such as pipes, small tunnels, and dams regularly despite the inherent risks of such activities. Underground tunneled systems are usual in many industrial situations. In the mining scenario, other types of confined scenarios such as natural caves are recurrent and the assessment of those natural structures is a legal requirement to determine their preservation or exploitation. Some risks related to the mentioned environments are the presence of venomous animals, noxious gases, bat excrement, getting stuck in small spaces, lack of oxygen, and the chance of roof collapse or falling from elevated grounds.

Given those hazardous conditions, the use of a remote robotic device that could enter into those complex environments and perform inspection tasks in a fast and reliable

The authors would like to thank Ramon Araújo and the speleology team of Vale S.A. due to the support in developing this project. The authors would also like to thank the colleagues from Mina do Veloso for their help in accessing the area to perform experimental validation. This work was supported by Instituto Tecnológico Vale (ITV), Vale S.A., Universidade Federal de Minas Gerais (UFMG), FAPEMIG, Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq), and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Dr. Gustavo Pessin acknowledges MCTIC/CNPq-Brazil processes 429096/2018-6 and 308425/2017-0. Dr. Gustavo M. Freitas also acknowledges MCTIC/CNPq-Brazil processes 443209/2015-4 and 402725/2018-2.

✉ Héctor Azpúrua
hector.azpuru@dcc.ufmg.br; hector.azpuru@itv.org

Extended author information available on the last page of the article.

way is a desirable solution to improve safety. However, from the robotics point of view, many of those scenarios, such as caves and tunnels, present challenging operational conditions. Irregular terrains, closed places, poor wireless communication, magnetic interference, lack of GPS signal, and slippery grounds are common in many of those environments. These hard restrictions are gaining attention from the robotic academy and industry; even the current DARPA (SubT) Challenge (2018–2021), a high-profile robotic competition, is now focused on underground environments [1].

A particular challenge for ground robots in subterranean or enclosed scenarios lies in the terrain topography, which is commonly complex and unstructured, presenting a mix of flat and rugged areas. These particular characteristics force any exploring device to have an efficient and adjustable locomotion system, allying obstacle transposing, energy consumption efficiency, and payload capabilities. In this sense, the Vale S.A. company designed the EspeleRobô (illustrated in Fig. 1a), which is a man-packable robotic device designed for inspecting confined spaces in mining operations. The EspeleRobô has six motors with versatile locomotion configurations, also including multiple embedded sensors such as cameras, Light Detection And Ranging (LiDAR), and Inertial Measurement Unit (IMU) that allow mapping the environment as a colored 3D point cloud. Although the initial designs of the robot focused on teleoperated natural cave inspection, the platform also extends to inspect other industrial confined spaces.

Despite teleoperation is one of the most used methods for robot navigation in inspection, search and rescue operations [2], a significant communication dilemma arises with these manual approaches. Since the platform must traverse unknown terrains and obstacles, the robot may be

operated without Line of Sight (LoS), meaning that any wireless communication will have degraded performance until full disconnection renders the robot unable to continue. A usual solution is to use tether cables whenever possible. On the flip side, the weight and resistance for dragging a cable for long ranges and even the risk of cord-cutting or entanglement with obstacles could make the tether cable approach unfeasible for several situations.

Given those challenges, the development of methods that allow autonomous or semi-autonomous operation is beneficial for the continuous operation of the device. In this sense, this paper presents the current developments aiming at the autonomous operation of the EspeleRobô platform. We further extend the robotic system presented in our previous work [3] by describing the current state of the platform and the pipeline used for autonomous inspection in confined environments. The pipeline includes Simultaneous Localization and Mapping (SLAM) with LiDAR, photorealistic mapping, path planning based on mobility optimization, and navigation control using vector fields to reduce operator dependency for commanding the robot. The proposed concept was validated in simulations with a realistic underground tunnel system and in representative real-world scenarios. The results endorse the viability of using the proposed concept for real deployments.

The remainder of this paper is structured as follows: Section 2 details the EspeleRobô robotic platform. Section 3 shows a high-level overview of the proposed methodology. In Sections 4 and 5 we present the proposed methodology for online and offline mapping, and in Section 6 the pipeline for autonomous inspection is described. Section 7 shows the results in representative simulated and real scenarios. Finally, Section 8 summarizes the contributions of this work and points to future research directions.

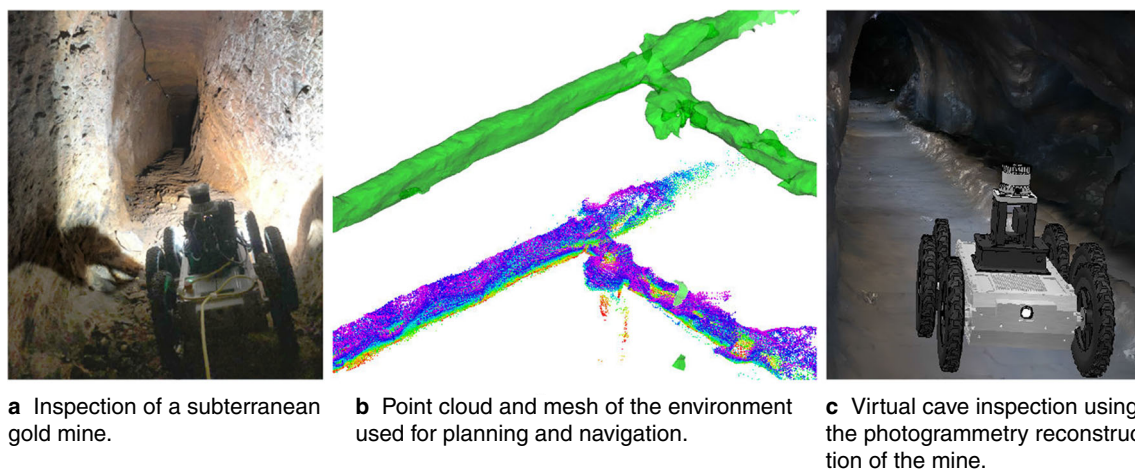


Fig. 1 EspeleRobô with the modular mapping unit exploring a mining cave in a: **a** real-world scenario (Mina du Veloso, Minas Gerais — Brazil), **b** mapping results and reconstructions of the environment

used for inspection, path planning and navigation, and **c** the photogrammetry reconstruction of the same environment used for simulations

2 Robotic Platform

In this section, we present the EspeleoRobô robotic platform, a small robotic device for inspecting confined spaces. We start with a brief explanation of the mechanical design, locomotion capabilities, and platform components. Then, we discuss the design choices for waterproofing the device and the platform's communication capabilities.

The EspeleoRobô uses a rigid chassis and six independent motors for thrust. The robot can move with different locomotion mechanisms that can be easily replaced through quick release pins. Some of the locomotion configurations include legs such as the Boston Dynamics' multi-task robot RHEX [4], circular and star-shaped wheels, and also tracks. The original hexapod architecture was chosen due to its good mobility capability, even over non-structured surfaces.

The different locomotion configurations allow the robot to adapt to various types of terrain and obstacles, allowing displacements over a higher variability of environments better than if there were only one locomotion mechanism available, such as wheels. The assembly pattern permits hybrid configurations. More details about the interchangeable locomotion system are described in the patent application [5]. Some of the locomotion's configuration can be observed in Fig. 2.

The platform was initially designed as a teleoperated tool for cave inspection using only the six legs configuration with no embedded computer, thus limiting any onboard processing and autonomous capabilities. Also, the original platform did not have any waterproof protection, making it prone to water damage during partially flooded areas, or in regions with high humidity like dam galleries and pipes. In this sense, the platform has now evolved from the original design to use lighter materials, improved onboard

sensors, processing capabilities, and now it has IP67 waterproof protection. Figure 3 depicts the evolution of the EspeleoRobô.

To make the device lighter, the internals of the robot needed mechanical structural adaptations. Some parts are now manufactured in polyacetal, aluminum 6061, magnesium and titanium alloy on the most robust components, such as drive shafts. In this sense the reduced dimensions of the robot ($0.55 \times 0.25 \times 0.14$ meters) and its weight (19 kg) facilitate the logistics in field operations, being transported inside a modified backpack or in a travel case.

2.1 Waterproofing

Inspections in partially flooded or high humidity scenarios are frequent for confined spaces. Our robot had a series of mechanical modifications to allow waterproofing. It has IP67 protection, making it possible to stay underwater up to 1 meter deep for half an hour and be sealed against dust.

The external mechanics of the robot were simplified to facilitate the waterproofing process. We reduced the number of top covers for robot upgrades from three to one to minimize the surfaces sealed with o-rings. In practice, this did not reduce capabilities for future robot upgrades, since the middle cover has access to all required internal connections. However, it significantly decreased the complexities of sealing the robot. The lower battery covers received a similar treatment, and are now manufactured on aluminum 6061 and use o-rings for sealing.

The remaining parts of the robot were sealed using mainly three methods: guarnital hydraulic sealing paper with 0.6 mm of thickness, liquid gasket seals (Loctite 518), and rubber seal cord (o-ring) as observed in Fig. 4. A particular case is the front camera and LED covers, which are fixed and sealed using liquid silicone.

Fig. 2 Possible configurations for the robot locomotion systems: circular wheels, star-shaped wheels, legs, tracks, and hybrid configurations

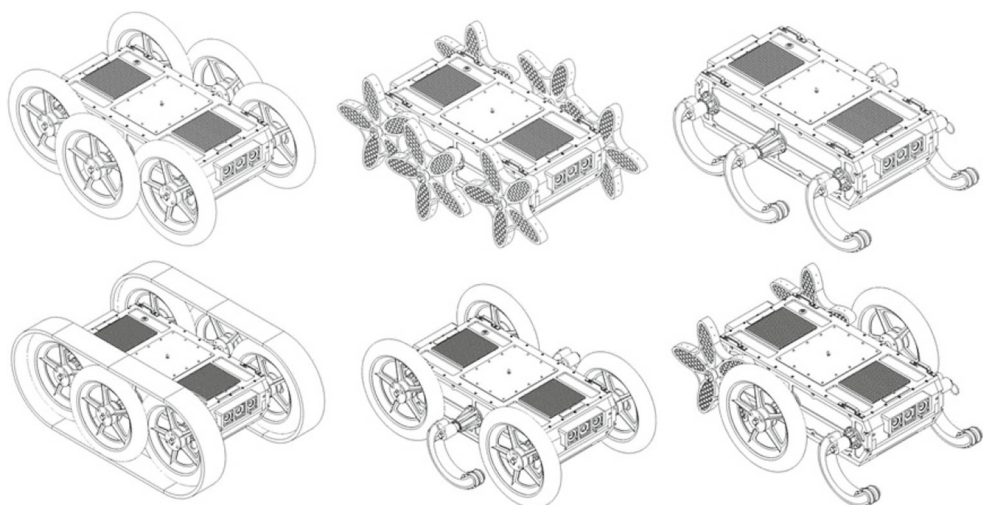


Fig. 3 EspeleoRobô's evolution, from left to right, starting at the first version in 2015 and ending at the current waterproof version of the platform



2.2 Embedded Equipment

The robot is currently equipped with six sets of reduction gears, motors and power drivers from Maxon motors (MCD EPOS), high-density energy military batteries from Bren-Tronics, a mini PC Intel NUC Core I5 running Robot Operating System (ROS) [6] with Ubuntu 16.04, high density LED illumination system from Cree (XP-G) and Stratus (100W module), and a pair of Axis-P12 cameras. Figure 5 depicts an overview of the system architecture inside the robot.

With a modular architecture, the robot can be easily modifiable to add new sensors by replacing the upper lid and perform the connection to the network switch, Arduino, or NUC embedded computer. Typical sensing set up with the platform is the Ouster OS 1 LiDAR, the RealSense T265 and D435i, and an Xsens MTI-G-710 IMU.

When operating in confined or unstructured environments, the communication link's reliability is crucial, meaning that the lack of connection during teleoperated missions can increase the risks of accidents with the robot. Thus, the platform has both wired and wireless modular communication systems that can adapt to the mission's requirements. The Ubiquiti Rocket M900 radio setup is the preferred wireless communication method used in the platform. The Rocket M900 provides long-range communication in the 902 Mhz to 928 Mhz frequencies and data rates up to 150 Mbps [7]. A Ubiquiti Airmax Amy 9m16 directional antenna is used at the base station, and two ASA-900CI 2 dBi omnidirectional antennas are embedded in the robot [8]. Optionally we also use Reseiwe WL24 modules for robust WiFi telemetry data. The Reseiwe WL24 is a wireless communication module that allows stable, low bandwidth, and low latency network connectivity using the ReWiLink system [9].

For wired communications, we use the Fathom-X Tether communication system that allows more than 350 meters of

high-speed Ethernet (up to 80 Mbps) over traditional two-wire power cables using the HomePlug AV (IEEE-1901) standard. This communication method is preferred for pipe or dam inspections, where obstacles are not common, and there is a low risk for entanglement of the communication cable.

2.3 Graphical User Interface

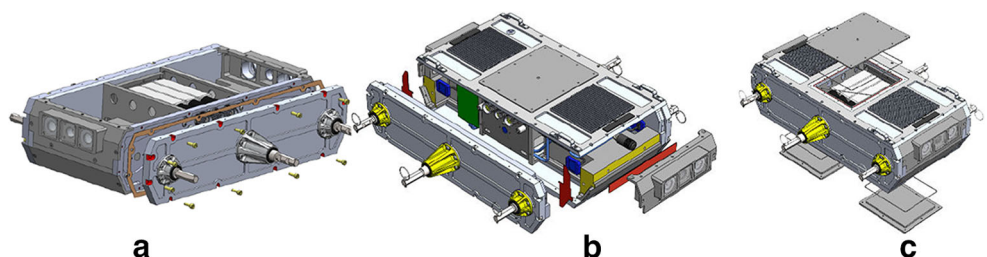
Studies on search and rescue or inspection robots for critical environments establish the Graphical User Interface (GUI) as an essential tool for mission success [10]. The GUI of EspeleoRobô was developed as an informative but easy to use tool to aid the pilot in operating the robot even in unfavorable conditions, to recognize the robot's environment, and to verify the state of the platform directly in one unified graphical environment.

The graphical interface was developed under the Qt framework using Python, integrated with the control and embedded systems inside the robot using ROS (Fig. 6). The information presented to the user is speed and current of each motor, real-time video streaming, odometry estimation, tip-over and radio signal quality alerts, lighting control, and easy access to different system procedures such as taking pictures, video recording and the toggle switch for the autonomous functionalities among other robot parameters.

3 Inspection Pipeline Overview

Autonomous navigation and exploration are still a challenge for mobile robots. Confined scenarios bring particular daring problems such as lack of global positioning, uneven terrains, wheel slip, and possible lack of visual features due to poor illumination. To overcome some of those problems, we propose a semi-automated navigation pipeline for exploration and mapping of confined spaces with the

Fig. 4 EspeleoRobô sealing methods: **a** Guarnital hydraulic sealing paper positioned between the side plates (brown), **b** Loctite 518 liquid sealing (yellow and red), and **c** O' rings at the upper and lower lids



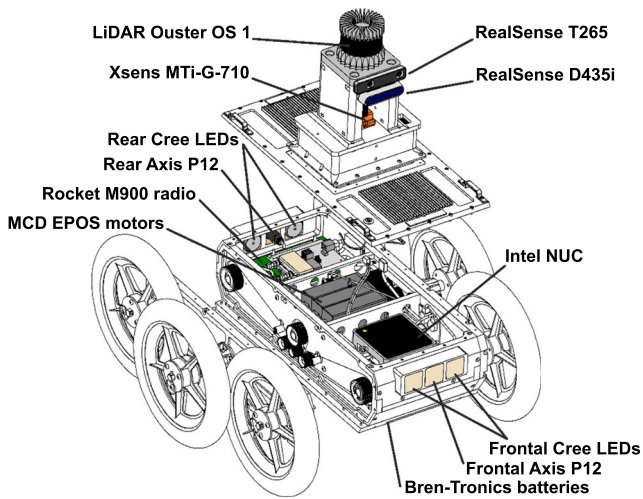


Fig. 5 EspeleoRobô platform: diagram of the proposed robot architecture and electronic components

EspeleoRobô. Our method comprises a modular architecture that deals with simultaneous localization and mapping, path planning, and navigation control, as depicted in Fig. 7.

The pipeline starts with the generation of online high-quality 3D maps of the environment using LiDAR or visual

sensors. After manually defining the next best point to visit, the path planner uses the point-cloud extracted from the LiDAR or Visual SLAM and generates a set of feasible robot paths given the selected locomotion metrics. Finally, the robot executes the generated optimal path by an Artificial Vector Field control algorithm. This process repeats until there are no more places to explore. At the end of the exploration, a photorealistic offline reconstruction is generated from the image logs captured by the RGB cameras at the previous stages. The following sections describe the sub-modules in detail.

4 Online Localization and Mapping

Autonomous navigation is a fundamental problem of mobile robotics that involves merging localization, planning, and control techniques working concurrently. The environment’s structure determines the proper use of these methods, especially when accurate localization is critical. In confined environments, the localization can be affected by the sensor’s interference and noise, lack of GPS and radio signals, and wheel odometry failures due to slip caused by rugged, wet, and slippery terrains.

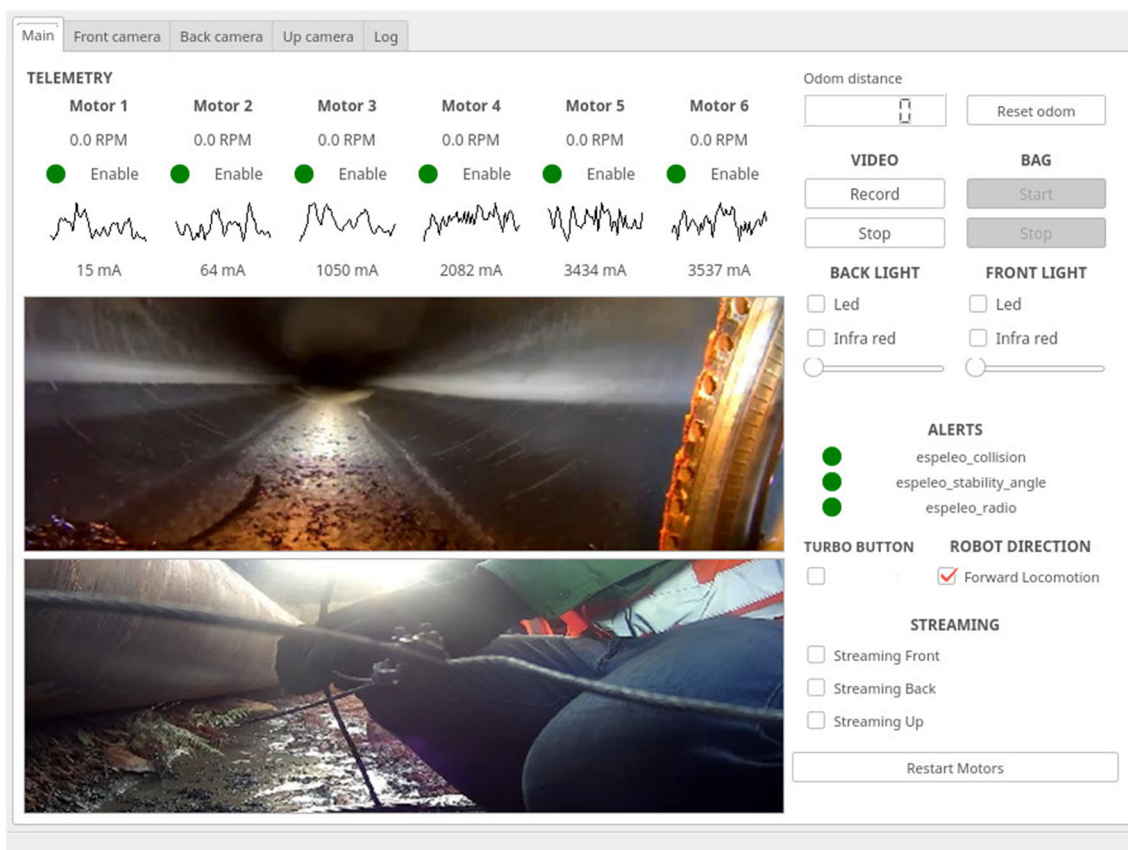


Fig. 6 EspeleoRobô’s Graphical User Interface, showing the state of several sensors, real-time video streaming, and the robot’s telemetry

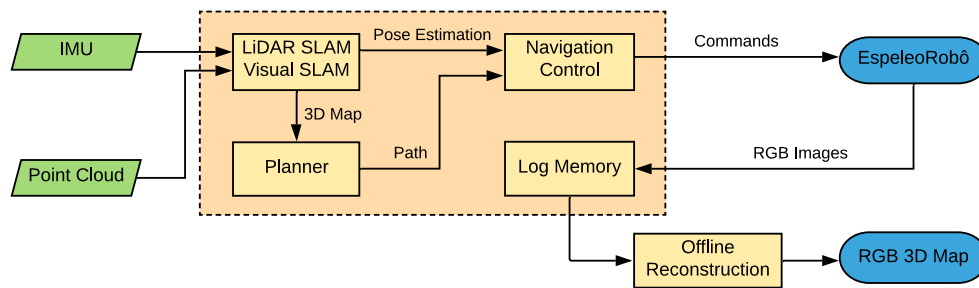


Fig. 7 Block diagram that shows the information flow of the navigation pipeline for inspection and mapping

In the last decades, localization and mapping have been some of the most studied subjects in mobile robotics [11]. While the term localization represents the process of estimating the pose of a robot, mapping involves the construction of models to represent the environment [12]. In many cases involving navigation of robots, it is necessary to estimate their poses and map the environment simultaneously, in a process known as simultaneous localization and mapping (SLAM) [13]. Different techniques and methodologies have been developed over years to solve the localization problem in an unknown environment from a map generated with information of sensors, as Visual-SLAM [14, 15] and LiDAR-SLAM [16, 17].

4.1 Visual-SLAM with RTAB-Map

A widely adopted technique is the Visual-SLAM, which uses image processing to obtain both localization and mapping. In this sense, camera-based sensors are common in SLAM applications since they are lightweight, energy-efficient, and provide information to perceive the shape, color and texture of the objects present in the environment [14].

The Visual-SLAM algorithms are categorized into two groups: direct, which uses the whole image to perform the SLAM, and feature-based, which extracts features from the images [15]. One of the most known methods for direct Visual-SLAM is the Large-Scale Direct SLAM (LSD-SLAM) [18], which uses intensity gradient and image fitting to predict the pose and build a dense map. In contrast, the ORB-SLAM is a popular feature-based technique that provides a sparse map and a robust pose estimation [19], using the Oriented FAST and Rotated BRIEF (ORB) descriptor.

To perform Visual-SLAM, we use the open-source algorithm called Real-Time Appearance-Based Mapping (RTAB-Map) [20]. RTAB-Map was developed by the Interdisciplinary Robotics Laboratory at the University of Sherbrooke, Canada, and was initially proposed as a loop closure detection method with memory management for large environments [21]. The algorithm uses images from RGB-D or stereo cameras as the main input information for pose

estimation and mapping. Also, it admits external data from IMU, wheel odometry and LiDAR to complement the procedures.

The RTAB-Map is widely used in Visual SLAM applications, standing out among the conventional SLAM methods found in the literature. The algorithm presents satisfactory results for localization and a faithful semi-dense reconstruction [15], which contributed to the choice of the RTAB-Map method instead of other popular algorithms. Figure 8 presents a schematic representation of the RTAB-Map SLAM process.

The main issue addressed by RTAB-Map is memory management, which can be divided into three types: Short-Term Memory, accessed during the odometry process; Long-Term Memory, which stores data related to the global map; and Working Memory. This last one involves all the local SLAM processes, which consist of synchronization, odometry, loop closure, proximity detection, graph optimization, and construction of the global map.

The localization process computes the robot's pose by registering the transformation of the current frame with respect to the local map of features (Map-To-Frame approach) or the last keyframe (Frame-To-Frame approach). In both approaches, the odometry procedure follows a sequence of steps including feature detection and matching, motion estimation and prediction, and update of the pose and the map.

The map structure corresponds to a graph that contains vertices and edges created at a fixed time interval. Each node holds local information (such as odometry pose, sensor data, visual words, and local occupancy grid), while the edges correspond to a transformation between two nodes. When a process (such as a loop closure or memory management) modifies a node, the graph optimization process takes place to correct the nodes' positions and minimize errors on the map.

4.2 LiDAR-SLAM

Visual-SLAM methods may be impaired by low lightness, which difficult the perception of features in the environment.

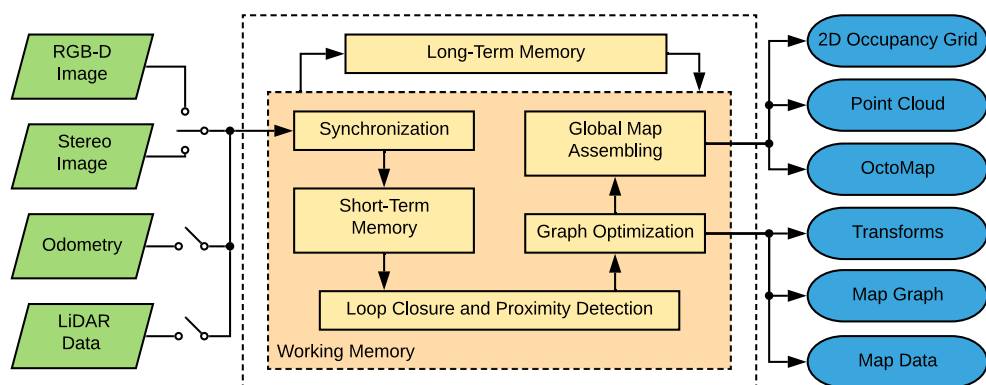


Fig. 8 Block diagram indicating the RTAB-Map operating process

A suitable solution is the use of LiDAR sensors for SLAM. The LiDAR provides robust measures of distances from the objects in the environment regardless of lighting conditions, as shown in [22].

There are several algorithms and techniques for LiDAR-SLAM that consider 3D [17] or 2D maps [23]. One of the most popular techniques currently used for 3D SLAM, based on Iterative Closest Point (ICP), is LiDAR Odometry And Mapping (LOAM) [24], which divides the SLAM problem into two algorithms: one estimating odometry at a high frequency by calculating the ego-motion between two LiDAR scans with low fidelity; and other for fine estimation of pose and map computation at a lower magnitude frequency by recording multiple scanning in a K-dimensional tree structure. The pose generated in an odometry algorithm is merged with the pose computed in the mapping algorithm, enabling the approach to run in real-time. Another method is HDL-Graph-SLAM [25], which uses the Normal Distribution Transformation (NDT) to estimate the displacement between two scans and the Graph-SLAM algorithm to estimate the pose with respect to a map and register the point cloud. This technique also tracks people in the sensor view to improve the estimation of three-dimensional pose and map. Although the people detection feature could improve the point cloud registration in crowded environments, it has little use in the hostile confined environments that this work focuses on, such as subterranean caves.

For the LiDAR-SLAM, we use the Lightweight and Ground-Optimized Lidar Odometry And Mapping (LeGO-LOAM) methodology [26], which is a light-weight version of the LOAM technique, optimized for land vehicles. Moreover, this method has a good performance running on embedded systems, which motivated us to consider it to perform the robot's localization. Figure 9 shows the overview of the LeGO-LOAM SLAM, subdivided into five modules: *Segmentation*, which projects the point cloud into the image range for segmentation in clusters; *Feature Extraction*, that obtains planar and edge points of the point cloud adjusted

by the IMU; *LiDAR Odometry*, which applies a two-step Levenberg-Marquardt optimization method to determine the components of a homogeneous transformation $\mathcal{H} \in SE(3)$ between two consecutive scans; The *LiDAR Mapping* that records points on a point cloud map by merging multiple scans into tree data structures, while the Levenberg-Marquardt optimization computes the device pose that is saved in a pose graph; Finally, the *Integration Transform*, which merges the poses returned by the LiDAR Odometry with the LiDAR Mapping module, and the algorithm outputs the final pose estimation.

The LeGO-LOAM approach uses the point cloud map to correct the pose of the robotic device, characterizing a complete problem of SLAM. The method also has the option of enabling the loop closure technique, which corrects possible deviation errors in the pose estimation and the map's construction.

We implemented the methodology by making adaptations in the LeGO-LOAM ROS package enabling it to run on a robot with the Ouster OS1-16 LiDAR. It was necessary to configure the LiDAR parameters and the sensor's pose with respect to the robot's base frame. Also, we parametrized some parts of the loop closure technique and LiDAR mapping to enable an easy configuration.

5 Photorealistic Mapping

Videos, photos, and realistic 3D maps are the main product of inspections in confined spaces. Despite the usefulness of video or monocular images, accurate three-dimensional representations improve understanding of challenging or hazardous situations for remote inspectors. Realistic maps can also be used as virtual training scenarios for robot operators or in simulations. In this section, we propose an approach for realistic offline maps using SLAM and photogrammetry. We also propose a method for generating fast and accurate 3D meshes from point clouds.

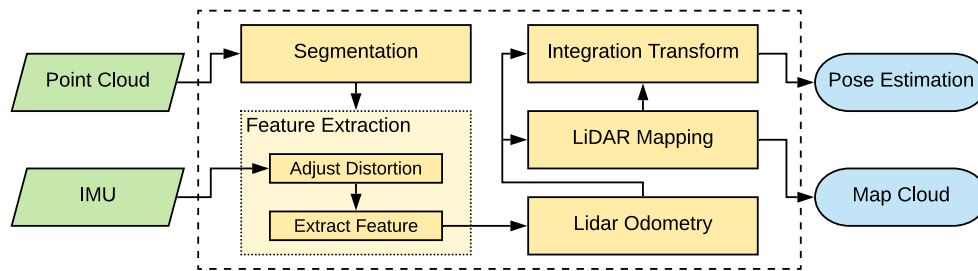


Fig. 9 System overview of the LeGO-LOAM package, adapted from [26]

5.1 Photogrammetry Reconstruction

In general, photogrammetry can be divided into two distinct groups: (i) metric photogrammetry and (ii) interpretative photogrammetry, where (i) consists of performing precise measurements and (ii) deals mainly in recognizing and identifying objects and environments [27]. Figure 10 illustrates the workflow of the interpretative photogrammetry that we use to perform a 3D reconstruction.

Thanks to recent advances in computer vision methods, photogrammetric techniques have received numerous improvements. State-of-the-art systems based on local feature matching across images [28, 29] and specialized optimization frameworks for large-scale non-linear problems [30, 31] significantly enhanced modern software's capability to generate accurate 3D reconstructions from large unordered image collections automatically. We use a workflow composed of seven steps: feature extraction [28]; image matching [32]; features matching [28]; Structure from Motion (SfM) [33]; depth estimation [34]; meshing [35]; and texturing [36]. Photogrammetry has many industrial and academic applications, and its commonly used in topography applications for generating soil, forest, geological, and city maps [27]. Thanks to the sensory capacities of the EspeleoRobô, we can apply a photogrammetry step to create realistic reconstructions of the environment using the onboard high-definition cameras. Our photogrammetry step is based on the AliceVision Meshroom open-source reconstruction software [37, 38]. This software implements the state-of-the-art reconstruction pipeline

and provides photorealistic 3D models with high-resolution textures.

Reconstruction methods that use RGB images are sensitive to illumination changes, lack of texture in images, and occlusion. In this sense, to perform a correct reconstruction in typical subterranean environments, the platform needs powerful external light modules to overcome the lack of proper illumination. Figure 11 shows the comparison between a photo used as input for the photogrammetry reconstruction algorithm and a snapshot of the reconstructed corridor section of a subterranean mine. It can be observed that the reconstruction yields realistic results for the mine's ground and walls. Holes in the corridor's uppermost region happen due to the restricted camera's field of view and could be improved by using wider angle lenses or a sensor directly aimed at the offending areas. Nevertheless, maps created with this technique can be used inside the robot's simulation environment to improve realistic robot behaviors and path planning.

5.2 Mesh Generation

Given an unstructured 3D point cloud, which is a discrete and noisy measurement of real surfaces in the 3D world, surface reconstruction methods estimate a bi-dimensional manifold that approximates the real 3D surfaces composing the scene, which is required for many practical applications in robotics. The most common representation used by many applications is the triangular mesh [39]. In the past years, considerable advancements have been achieved regarding

Fig. 10 Workflow of the photogrammetry pipeline

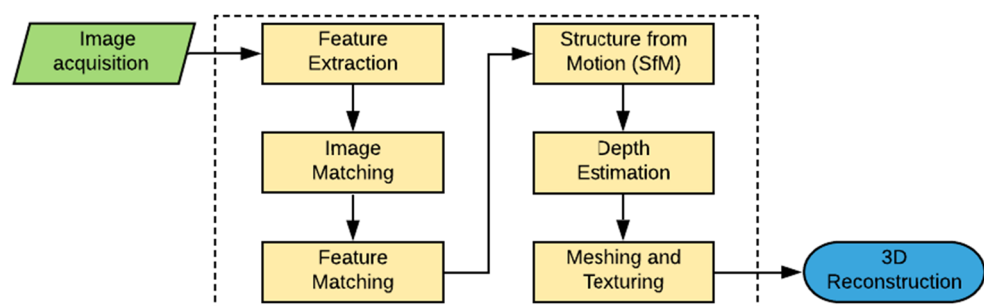
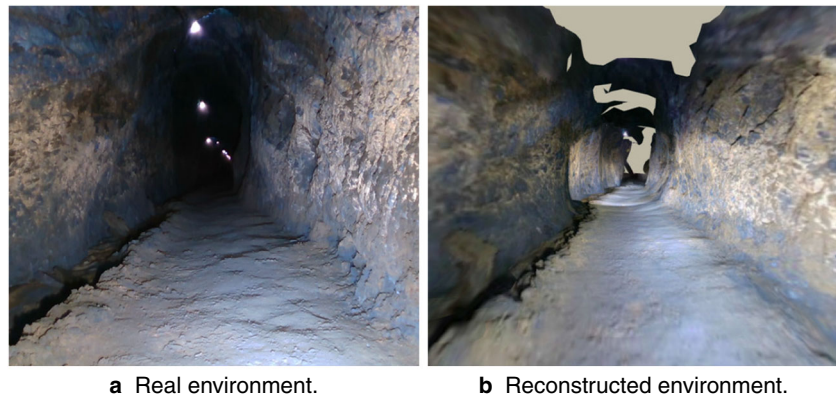


Fig. 11 Photogrammetry reconstruction example in a representative subterranean mine environment: **a** An image of the environment during the inspection, and **b** a snapshot of the 3D reconstruction



the accuracy, robustness, and efficiency of the surface reconstruction methods. Among representative methods are: the Marching Cubes algorithm [40, 41], Radial Basis Functions (RBFs) [42] and more recently, Poisson's reconstruction [43]. Nevertheless, state-of-the-art methods still require extensive parameter tuning to generate good results from real-world data.

The output of several mapping algorithms used in the EspeleoRobô comes in the form of an unstructured, colored or monochromatic point cloud. This input data will most likely contain noise, outliers, misalignment, a non-uniform point sampling, and missing parts. In order to tackle these problems, we propose a robust and automatic mesh processing pipeline to generate realistic meshes. The resulting 3D mesh can be used in a myriad of applications in Robotics, Computer Vision, and Virtual/Augmented Reality (VR/AR). Although an experienced 3D expert can produce visually compelling, high-quality meshes using advanced software pipelines, it comes at the price of spending plenty of time tweaking a large number of parameters. Our mesh processing pipeline has been designed to provide an optimized textured mesh ready to be used in simulation and virtual/augmented reality platforms. The algorithm provides an intuitive set of parameters available for non-expert users to adjust and, in most cases, works out-of-the-box. It is composed of four main steps: (1) Normal estimation; (2) Surface reconstruction; (3) Hole filling; and (4) Color embedding. Figure 12 illustrates the complete pipeline.

Preprocessing Most of surface reconstruction methods require the normals of each point to be estimated beforehand. Thus, for each 3D point, we fit a local plane using the singular value decomposition (SVD) of the covariance matrix of nearby points. After estimating the normals for all points, we orient the normals using the method presented by Hoppe et al. [44]. We compute a minimum spanning tree (MST) over the Riemannian graph and a seed orientation is chosen and propagated throughout the MST.

In order to enforce uniformity in the spatial distribution of the points, we apply a grid-based downsampling strategy to the raw pointcloud. This step increases the stability of the surface reconstruction algorithm. The grid-based downsampling strategy requires only the cell size as parameter input that can be extracted from the data by considering the average distance between the k -NN points. In this paper, the six-ring average distance between the 6-NN points is defined by the parameter d_{avg} and is estimated from the input point cloud.

Surface Reconstruction Since we work with the arbitrary shapes of caves and other confined spaces, a suitable method for reconstructing the surfaces is the Poisson Surface Reconstruction (PSR) [43]. In our pipeline, we compute the implicit function using the PSR method and also extract an isosurface and triangulate the final mesh [45]. We used a minimum triangle size and surface approximation error *w.r.t* to the average point spacing that provides a balanced trade-off between accuracy and mesh complexity.

Mesh Trimming Due to the cavities of arbitrary shape, it is necessary to trim some parts of the mesh. The PSR assumes that the object has a closed form, which leads it to over-extrapolate the opening of the cave when estimating the implicit function. To remove the over-extrapolated portion, we use a threshold distance $\tau = \alpha d_{avg}$, where α is a parameter that controls the maximum allowed distance in six-ring average distance units. All vertices of the mesh having a distance from the nearest neighbor point in the original point cloud bigger than τ are removed as well as all associated edges and faces connected to them.

Hole-filling Algorithm When scanning a cave, several holes naturally arise in the point cloud because of the shadows created when the laser rays are blocked by the cave structure. To solve the problem of holes on the mesh, we propose to use the mesh trimming algorithm first to obtain

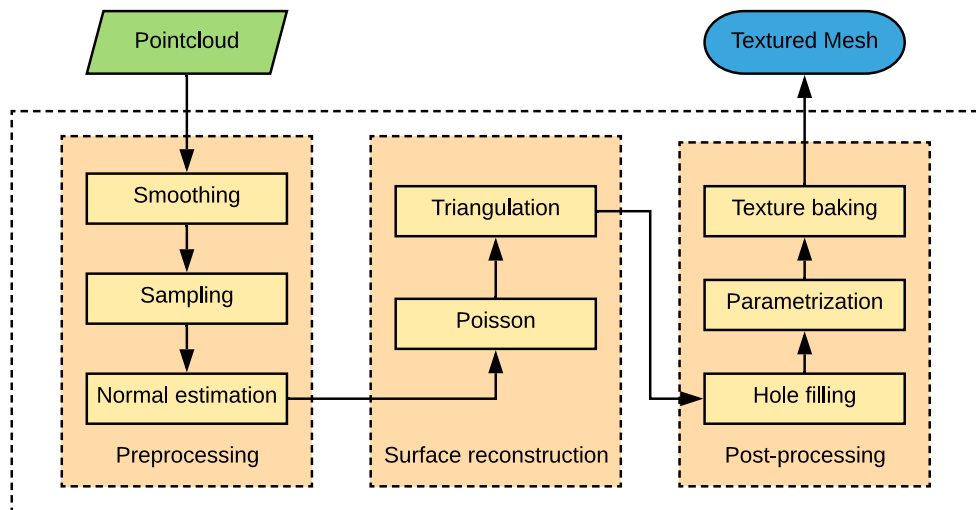


Fig. 12 Automatic mesh reconstruction pipeline overview

a well-conditioned reconstructed surface with reduced artifacts but with holes. Second, a hole-filling algorithm adapted from [46] is used to consistently fill all holes. A perimeter classification threshold based on τ is defined to avoid filling, *e.g.*, the entrance of the cave and other parts that do not represent actual holes.

Mesh Parametrization A simple but efficient per-face parametrization was chosen in our pipeline. The per-face parametrization seamlessly handles any surface topology and does not require solving linear systems in contrast to most per-vertex algorithms, which can be ill-conditioned in some cases. A disadvantage of the per-face parametrization is that it distorts the texture from triangles having very acute angles. Considering this issue, in a previous step, we optimize the triangles' shape to have similar internal angles (minimum angle value of 20°). We extend this technique to consider the size of the triangles in the parameter space to minimize loss of resolution in texture. Our implementation of the per-face parametrization is based on packing the triangles in the parameter space in a sorted manner, from the largest to the smaller ones, aiming to reduce the loss of resolution in large triangles. The arrangement of the triangles uses a parameter space described by squares of size $l \times l$. Each square parametrizes two or more faces as shown in Fig. 13a.

Texture Baking After per-face parametrization is performed, we use the inverse distance weighting (IDW) scheme [47] to bake the texture map into an image using the original RGB point cloud. To convert the image coordinates $p(x, y) \in \mathbb{R}^2$ into 3D coordinates $P(x, y, z) \in \mathbb{R}^3$, we borrow the shading idea from Phong [48]. A bijective mapping function $\phi: \mathbb{R}^2 \rightarrow \mathbb{R}^3$ is defined by linearly interpolating positions (instead of normals) along segments, and

then linearly interpolating the values between the interpolated values of the segments. This approach is illustrated in Fig. 13b.

Using the 3D point found in the mesh space, a KD-tree is employed to query the k nearest neighbors of this point ($k = 6$ in our approach). The R, G and B values of each p point in the texture map are obtained according to the k-NN IDW interpolator:

$$c(P) = \begin{cases} \frac{\sum_{i=1}^k w_i(P)c_i}{\sum_{i=1}^k w_i(P)} & \text{if } d(P, P_i) \neq 0 \text{ for all } i, \\ c_i & \text{if } d(P, c_i) = 0 \text{ for some } i, \end{cases} \quad (1)$$

where $w_i(P) = 1/d(P, P_i)^n$ is the inverse weighted distance powered by n , which in this paper was set to 1.5, $P = \phi(p)$ is the position of the 3D point mapped from a 2D texture coordinate, $d(\cdot, \cdot)$ returns the euclidean distance between two points, c_i is a known value for the nearest neighbor query, and k the number of the nearest neighbors used by the algorithm. A real sample of the generated texture map can be seen in Fig. 13a.

Figure 14 shows a reconstruction example from a point cloud of a representative subterranean mine scenario. We can see that the input cloud contains challenging misalignments and noise from the RGB-D sensor, and the final textured mesh does not present visible misalignments or artifacts. We make our implementation available as open-source code¹. The output reconstruction follows standard 3D model formats, and can be easily integrated with several other mapping algorithms and directly imported on rendering platforms.

¹<https://github.com/verlab/mesh-vr-reconstruction-and-view>

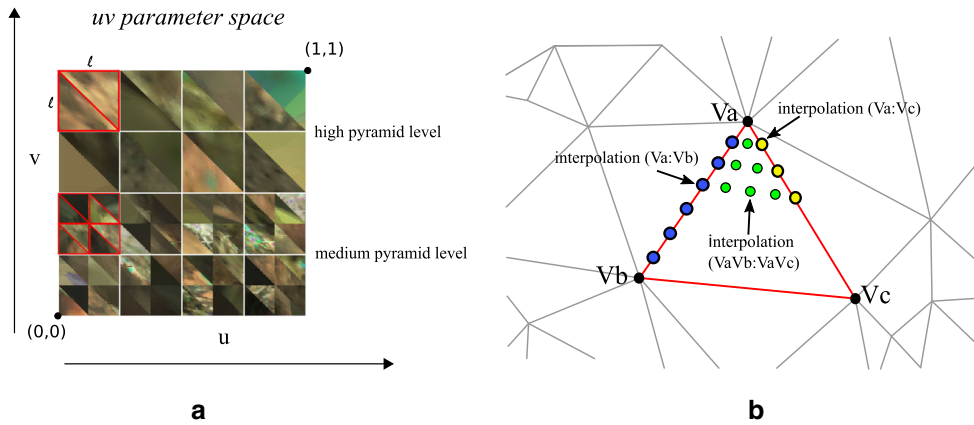


Fig. 13 **a** Proposed parametrization scheme. We divide the parameter space into a grid and pack the faces in the respective pyramid level based on their area. We use three pyramid levels in our implementation, but for sake of simplicity, we depict an example with two pyramid levels. **b** The mapping function ϕ maps pixels coordinates into 3D points

6 Autonomous Navigation

For autonomous navigation, robots need a reference path and a control system working in conjunction with a localization system. This section presents our planning approach to generate appropriate terrain-aware reference paths based on mesh reconstructions and the Dijkstra Algorithm. Also, we present a control method based on artificial vector fields capable of making the robot follow this planned path efficiently.

As the robot will need to traverse complex 3D environments, traditional planning techniques that rely only upon 2D information are not adequate [49]. In this sense, a graph representation allows increased flexibility in representing terrain topography and multi-level scenarios. There are several graph-based algorithms to generate the graph and find a path already used for robotic navigation [50]. One of the most popular algorithms, RRT* (Optimum Rapidly-exploring random tree), adopts a probabilistic strategy that generates a tree from a set of sampled vertexes. In this work, we use a mesh to represent a complex environment. Thus, probabilistic path generation algorithms may not be the most appropriate since small deviations of the best path could cause robot failure. Therefore, we opted to use the

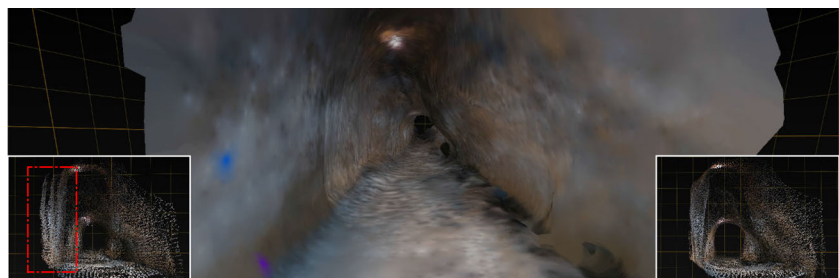
Dijkstra Algorithm; this way, we define a terrain cost for each edge of the graph and then use Dijkstra to search for the best path.

Regarding the control module, trajectory tracking can be considered as a possible solution, which is already used by several papers in the literature [51, 52]. In our case, the planning algorithm generates a path, not a trajectory, which means that there is no bound between the points in the path and time. Several path-following algorithms have been proposed in the literature, such as the ones based on virtual target points [53], and artificial vector fields [54]. This work takes upon the vector field approach for path-following since it has demonstrated to be less susceptible to practical motion failures such as slipping of the wheels [55].

6.1 Path Planning

After the definition of the target points where the robot should move to explore the uneven environment, it is necessary to compute feasible paths connecting the robot's current and goal positions. In the current stage of our exploration pipeline, the target points are defined by the human operator, while the optimum paths are computed autonomously.

Fig. 14 Subterranean mine mesh reconstruction. At the bottom-left, the input cloud contains visible noise and misalignment, marked in dashed red. After the preprocessing step, the raw cloud is filtered (bottom-right). At the middle, the final textured mesh



Natural caves commonly present rough terrain, which is challenging for path planning. Complex terrains require a full three-dimensional map for planning a safe and efficient robotic locomotion. Three-dimensional meshes are highly descriptive environment models since they can represent any 3D shape. Unlike 3D point clouds, they have face information that is useful for terrestrial robot mobility. Other representations such as elevation maps have difficulties in representing some cave structures, e.g. arcs and narrow tunnels. In this sense, we extend the path planning pipeline presented in [3, 56] by integrating a mesh generation process in the planner, performing pre-processing filtering of non-reachable regions. This pipeline, in contrast to our previous works, can be executed online at the exploration mission and mapping phases. Figure 15 illustrates the new path planning pipeline.

The mesh generation process for ambient reconstruction combines local point clouds provided by the LiDAR-SLAM methodology presented in Section 4.2 and the triangulation procedure described in Section 5.2.

The path planning pipeline models the 3D mesh representing the environment as a graph $G = (E, V)$, where the faces centroids correspond to the vertices V , and the edges E connecting the nodes have an associated cost regarding the adopted metric. Over this graph, we use the well-established Dijkstra algorithm to calculate optimal paths according to different metrics, such as distance, traversability, energy consumption, or a combination of them. Consider a route $p = \{n_1, n_2, \dots, n_i, \dots, n_{goal}\}$ consisting of neighboring nodes and $\bar{p} = p \setminus n_1$. In the following we specify how each metric is computed for this route.

Traveled Distance Metric This metric aims to find the shortest path from the robot’s current start position to a goal. The cost function using this metric is defined as follows:

$$C_1(p) = \sum_{n \in \bar{p}} D(n), \tag{2}$$

where $D(n)$ is the 3D euclidean distance between the points n and $n - 1$.

Terrain Traversability Metric The terrain traversability metric helps to find the flattest path from start to goal positions. This metric defines the cost function $C_2(p)$ based on the

positive angle $T(n)$ between the mesh normal vector (\mathbf{N}_i) and the canonical Z-axis (\mathbf{Z}), such as:

$$T(n) = \arccos\left(\frac{|\mathbf{N}_i \cdot \mathbf{Z}|}{\|\mathbf{N}_i\| \|\mathbf{Z}\|}\right), \quad C_2(p) = \sum_{n \in \bar{p}} T(n). \tag{3}$$

Energy Consumption Metric This metric aims to compute the path that leads to the minimum energy consumption from the start to the goal. For simplification purposes, we considered that the robot moves with constant velocity; this means that the robot spends energy while accelerating and braking to keep the uniform movement.

The energy $E(n)$ required to move the robot from neighboring nodes $n - 1$ and n is estimated as a linear regression of the battery consumption and terrain inclination, the friction coefficient, robot mass, angle θ between the vector linking the mesh centers, resulting in the cost function $C_3(p)$. E_{mean} is the mean battery consumption while turning 2π rad and a, b are the linear regression parameters. The quantities $\delta(n)$ and $\alpha(n)$ are estimates of the angular and linear displacements, respectively, when the robot moves between nodes $n - 1$ and n . Finally we define $C_3(n)$ as:

$$E(n) = \left(\frac{E_{mean} \alpha(n)}{2\pi}\right) + (a\theta(n) + b)\delta(n), \quad C_3(p) = \sum_{n \in \bar{p}} E(n). \tag{4}$$

Combined Metrics Considering the conflicting objectives, we propose a cost function based on all the multiple metrics previously mentioned during the path planning, where the robot operator can set a trade-off between them through weights. Therefore, the algorithm evaluates the relevance of each metric while finding paths connecting the start and goal positions. Thus, the cost function is given as follows:

$$C_4(p) = \sum_{n \in \bar{p}} [P_d N_d D(n) + P_t N_t T(n) + P_e N_e E(n)], \tag{5}$$

where P_d, P_t and P_e are the weights that set the metric priorities related to distance traveled, terrain traversability, and robot energy consumption, respectively. The scalars N_d, N_t e N_e are normalization coefficients calculated according to the neighboring nodes. When exploration time is a priority, the operator can set P_d with a higher value than the others. In environments with a high risk of tipping over, P_t

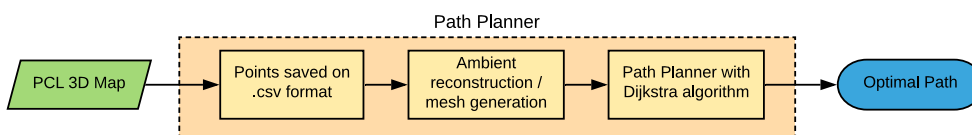
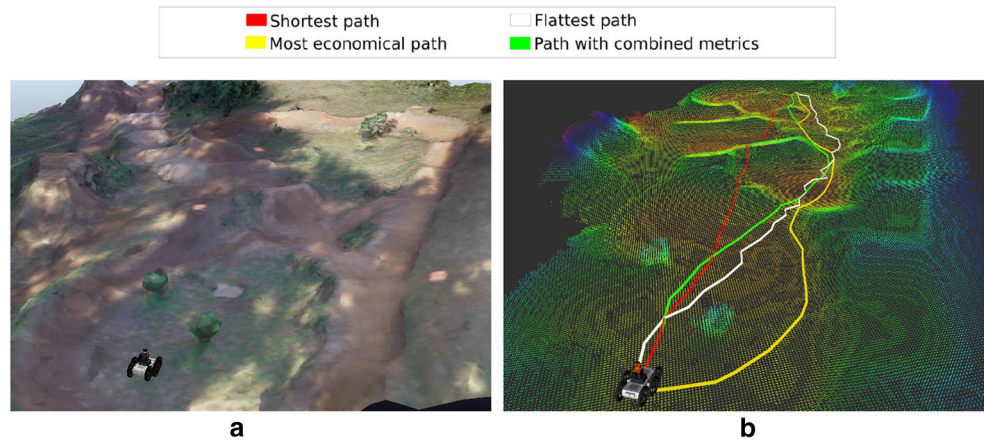


Fig. 15 Path planner workflow

Fig. 16 Path planning validation in simulated reconstruction of a rugged environment. **a** Simulated environment in CoppeliaSim. **b** Point cloud and paths according to each metric



requires a higher priority. On the other hand, in situations where energy consumption is critical, P_e should be greater.

The path planner works as a finite state machine, where the robot needs to reach the final path waypoint (or another terminal state) to plan a new path. The mesh reconstruction algorithm must get the most extended point cloud possible – generally from the final location of the previous path – to allow the mesh to cover the robot’s maximum reachable range and maximize the next waypoint’s distance from the current robot position. This prevents generating short paths for a limited view of the environment, and allows to estimate fewer and longer paths.

To validate the path planner, we used a rugged representative simulated environment to compute paths connecting multiple points through obstacles and uneven terrains. Figure 16a shows the CoppeliaSim² scene considered for the simulation. The terrain corresponds to a Motocross field in the city of Ouro Preto (MG-Brazil), whose model was obtained with the photogrammetry reconstruction techniques presented in Section 5.

Figure 16b shows the point cloud used to compute the cost functions of each metric described. The paths in red, white, and yellow correspond to the shortest, flattest, and most economical paths, respectively. The green path was obtained with the combined cost function. The paths behave as expected, as the shortest path tends to perform as a direct line to the goal, and the most economical and flat ones tend to minimize terrain roughness and high slopes.

6.2 Navigation Control

Given that the path is already defined, it is necessary to compute control signals to make the robot follow this reference. The control problem is solved with the use of artificial vector fields. This methodology consists in a function $\mathbf{F}(\mathbf{p}) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ that defines a reference velocity

F for the robot at each point \mathbf{p} it may be. This velocity is responsible for guiding the robot towards a reference path. Since the EspeleoRobô is not holonomic, an additional technique is necessary in order to impose $\dot{\mathbf{p}} = \mathbf{F}(\mathbf{p})$. The Feedback Linearization [57] is able to solve this problem by computing linear v and angular ω velocities given the translation velocity reference $\mathbf{F}(\mathbf{p})$. This strategy consists in attributing the reference velocity to a virtual point located at a distance d , in the forward direction, from the robot’s rotational center. In the experiments performed, we choose the parameter $d = 0.2\text{m}$. Figure 17 shows the control structure used for the autonomous navigation control.

The methodology to construct the vector field $\mathbf{F}(\mathbf{p})$ given a reference path is presented in [54]. The reference path \mathcal{C} must be represented as the zero level set of a scalar function $\alpha(\mathbf{p})$, i.e. $\mathcal{C} = \{\mathbf{p} \in \mathbb{R}^2 : \alpha(\mathbf{p}) = 0\}$. It is also necessary that the gradient $\nabla\alpha$ does not vanish at the curve, in other words, $\nabla\alpha(\mathbf{p}) \neq \mathbf{0} \forall \mathbf{p} \in \mathcal{C}$. Similar to [54], the vector field is computed as:

$$\mathbf{F}(\mathbf{p}) = v_d k_G(\alpha) \frac{\nabla\alpha}{\|\nabla\alpha\|} + v_d k_H(\alpha) \frac{R_{90^\circ} \nabla\alpha}{\|\nabla\alpha\|}, \tag{6}$$

in which v_d is the desired speed of the robot, $\nabla\alpha$ is the gradient of α and R_{90° is a rotation matrix of 90° . Given a convergence gain $k_f > 0$, function k_G is defined as $k_G = -(2/\pi)\text{atan}(k_f\alpha)$ and $k_H = \sqrt{1 - k_G^2}$. An example of an α function with parameters r, c_x, c_y and γ is:

$$\alpha(\mathbf{p}) \equiv \alpha(x, y) = ((x - c_x)^\gamma + (y - c_y)^\gamma)^{\frac{1}{\gamma}} - r. \tag{7}$$

Figure 18 shows two examples of vector fields generated by Eq. 6, both of them can be obtained with the α function described in Eq. 7 with $r = 2.8\text{m}$, $c_x = 1.0\text{m}$, $c_y = 1.5\text{m}$. The circular path on the left corresponds to the parameter $\gamma = 2$ while the square like curve on the right to $\gamma = 4$.

In order to compute the vector field, it is necessary to have a function α whose zero level set is the desired path. In the general case, when the paths are represented as a sequence of points, such as the ones provided by the

²<http://www.coppeliarobotics.com>

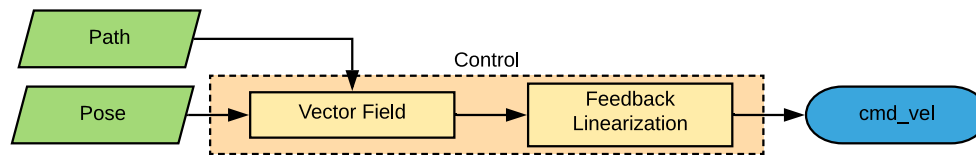


Fig. 17 Control diagram including vector field and feedback linearization

techniques presented in Section 6.1, a numerical method is needed to define the function α . In [54], the authors present a solution to construct an α function from a set of points. The method uses radial basis functions [58] and relies on the solving of a linear system. It is perfectly applicable to the paths generated by the path planner presented in Section 6.1.

A Python implementation, compatible with ROS, of the vector field-based control is available [online](#)³. It subscribes to the robot's pose topic from the LiDAR-SLAM system, and to the path topic from the planner system. This information is used to compute the robot's control signals.

7 Experiments and Results

The proposed mapping and navigation pipeline was evaluated through simulated and real-world experiments with the EspeleoRobô in representative scenarios of subterranean caves and indoor spaces.

7.1 Simulated Results

To validate the pipeline in a safe scenario, we used the subterranean cave environment from the DARPA Subterranean Challenge⁴. This environment has obstacles, uneven terrain, and realistic subterranean cave geometry (Fig. 19). All experiments were performed using the virtual version of the EspeleoRobô inside the CoppeliaSim simulator, executed with ROS Kinetic and Ubuntu 16.04.

7.1.1 Mapping in a Simulated DARPA Cave Scenario

The three methods for mapping and reconstruction were validated in a sub-section of the cave map. The online visual mapping methodology was tested using an RGB-D device with similar configurations to the robot's embedded Intel RealSense Depth D435i. As this camera has a narrow field of view, the robot moved through a route that maximizes the explored space. In this sense, it is possible to capture information from all the environment, maximizing the map's completeness. Also, the route used enforces local loop closures to reduce localization errors and deformations on

the map. The LiDAR algorithm was evaluated with simulated versions of a Velodyne VLP-16 (16 lines) and a Xsens IMU. The photogrammetry reconstruction was performed with the RGB images acquired by the simulated RGB-D sensor.

Figure 20 presents inner and external views of the final map obtained with the online Visual and LiDAR SLAM algorithms, including the estimated odometry. Despite the limited sensor range and other challenges for Visual-SLAM methods, such as reduced lighting and texture similarity, the resulting model estimated with Visual-SLAM (Fig. 20a) looks very similar to the original cave, preserving rich geometric and texture features. The point-cloud estimated by the LiDAR-SLAM, observable in Fig. 20b, is also faithful to the real cave scenario and contains fewer errors and holes than the Visual-SLAM estimation, which is understandable given the higher range and field of view of the LiDAR.

The odometry estimated online by the SLAM methods can be observed in Fig. 20c, showing a significantly reduced localization error of the LiDAR-SLAM algorithm compared to the ground truth odometry. In this sense, given the structural properties of the studied scenarios, the LiDAR-SLAM algorithm showed up as a more suitable option for navigation and localization.

A comparison of the point clouds generated by the LiDAR-SLAM, Visual-SLAM, and photogrammetry, using the DARPA map ground truth, is depicted in Fig. 21. The top row presents the point cloud generated by each method, the middle is the comparison between the clouds and the DARPA map, and the bottom row shows the error histogram of the point clouds. The LiDAR-SLAM map presented the best results with a map composed of 31,239 points, with 95% of the cloud points exhibiting errors smaller than 0.115 m, a 3σ (99.7% of the steady-state) interval smaller than 0.258 m, and a maximum total error of 0.33 m. The Visual-SLAM algorithm achieved intermediate results, estimating a cloud with 383,862 points and an error of less than 0.185 m in 95% of these points, a 3σ interval of 0.48 m and a maximum error of 0.8 m. Although the photogrammetry obtained good performance for 95% of the 79,404 cloud points, with errors smaller than 0.054 m, the 3σ of 0.332m associated to a 1.85 m maximum error configure the worst results.

The photogrammetry reconstruction was performed offline and used 405 images of size 640×480 taking a total time

³ https://github.com/adrianomcr/vector_field_control

⁴“cave_02” from <https://github.com/osrf/subt>

Fig. 18 Illustration of the vector field (blue) that guides the robot to the curve (in black). Real experiment results are shown in red, tracking the vector field, using LiDAR-SLAM information as pose feedback

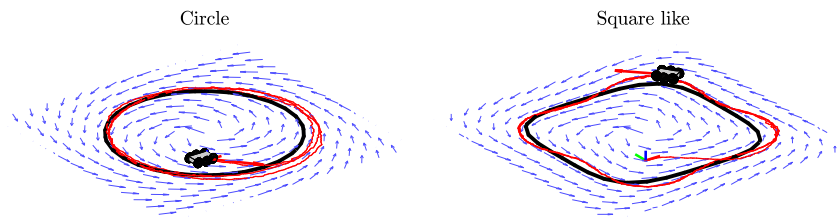
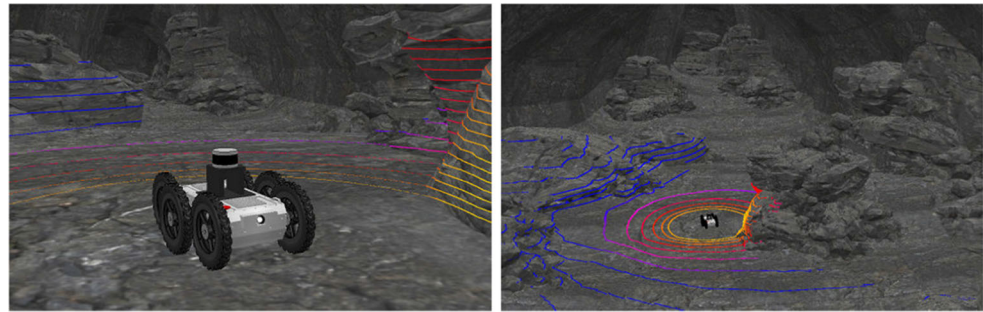
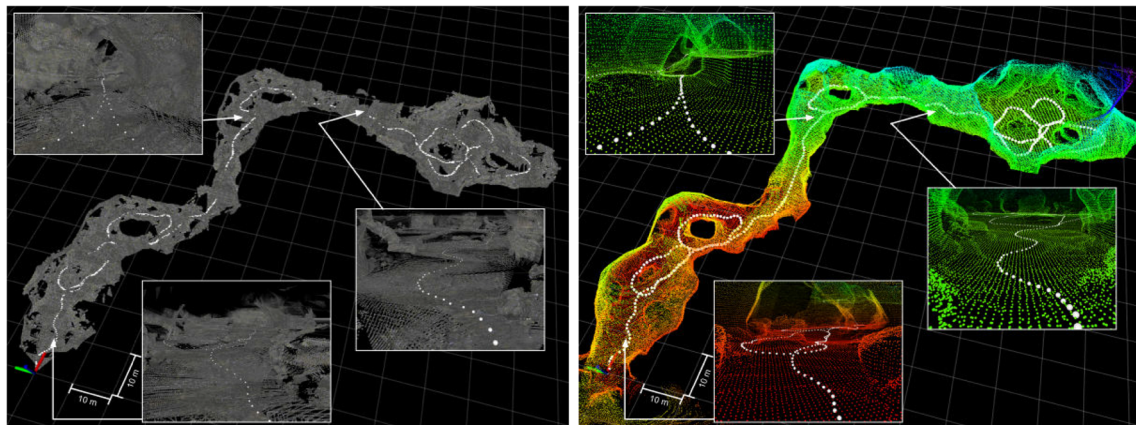


Fig. 19 Experimental setup with the CoppeliaSim simulator: (a) Simulated EspeleoRobô, and (b) inside view of the cave



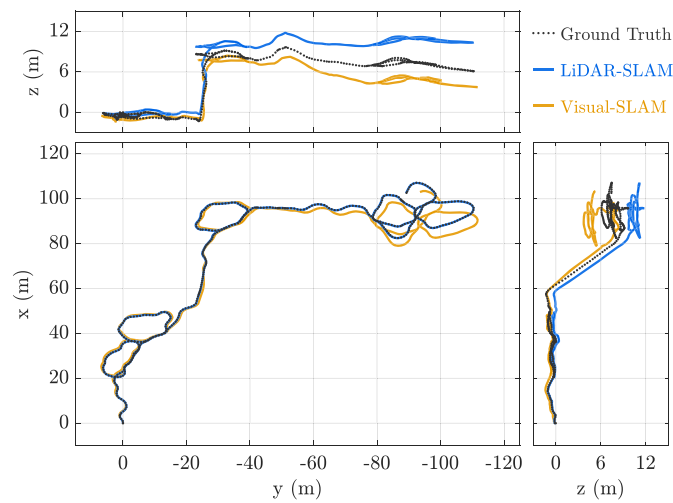
a Simulated EspeleoRobô with LiDAR.

b Inside view of the cave environment.



a Visual-SLAM point cloud.

b LiDAR-SLAM point cloud.



c Odometry estimation. Odometry estimation.

Fig. 20 Point cloud mapping results and visualization with the proposed online SLAM methods in a simulated DARPA cave scenario: **a** Visual-SLAM and **b** LiDAR-SLAM. In **c**, the odometry for both methods is depicted in blue and yellow solid lines

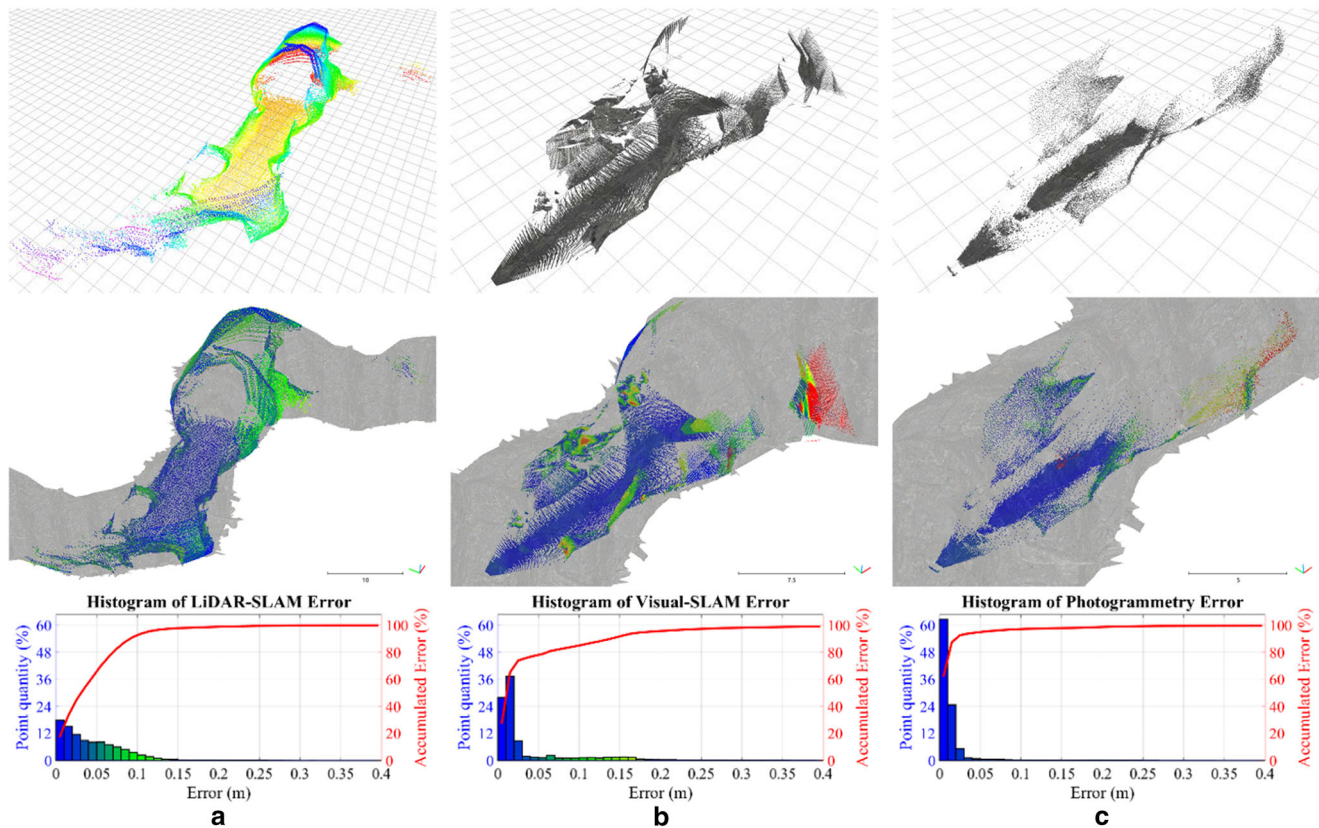


Fig. 21 Point cloud error analysis. Comparison between the ground truth map and partial maps obtained with: **a** LiDAR-SLAM, **b** Visual-SLAM, and **c** Photogrammetry in the simulated DARPA cave scenario

of one hour and 10 minutes in a laptop equipped with an i7-10875H CPU, with 32GB of RAM and an Nvidia RTX 2070 video card. The results showed that, despite the Visual-SLAM online method was faster to compute (56 ± 12 ms per frame and 486 ± 123 ms per keyframe) and more efficient than photogrammetry, the latter method for reconstruction generated a point cloud 99.7% more faithful to the original cave, but with a few larger outliers with a greater maximum error. However, the camera used as the source for the photogrammetry reconstruction had certain limitations in capturing the surrounding environments only using RGB data (the depth sensor has an increased FOV), yielding a denser and detailed reconstruction only of the ground where the robot went.

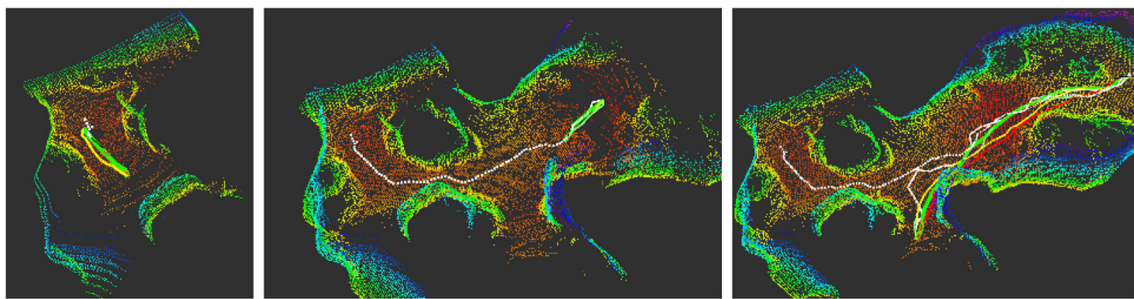
On the other hand, the LiDAR SLAM provides the robot odometry for the remainder robot modules at 0.1 s and the map at 5 s. The processing time for all steps in simulation is $1,239 \pm 226$ ms, calculated from the entry of the point cloud scan provided by the sensor until the map's update. This time depends on the number of loop closure events and the submap size updated on the map. In the simulation, LiDAR SLAM identified many loop closures, which increased the processing time. Finally, the LiDAR-SLAM algorithm

presented a reduced and more consistent error rate than the other approaches while having a similar processing time than Visual-SLAM, although at an increased range.

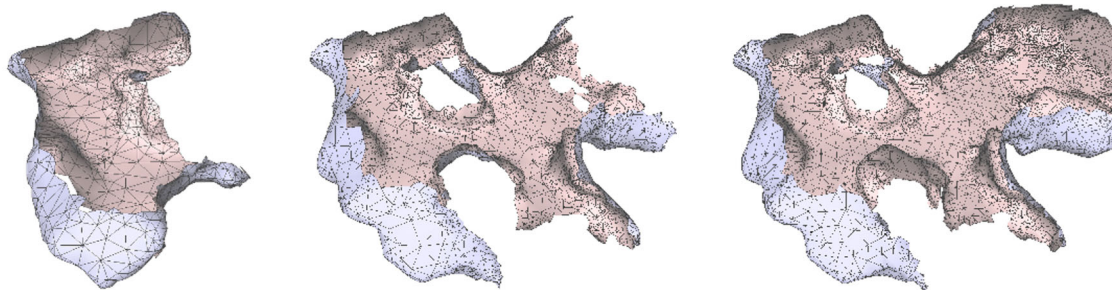
7.1.2 Autonomous Navigation in a Simulated Scenario

The complete pipeline of navigation, mapping, localization, and control was evaluated on a simulated exploration mission, as shown in Fig. 22. In this experiment, the robot performed path planning using the point cloud generated by the LiDAR mapping algorithm. This point cloud is converted to a 3D mesh, and the paths are calculated using this mesh.

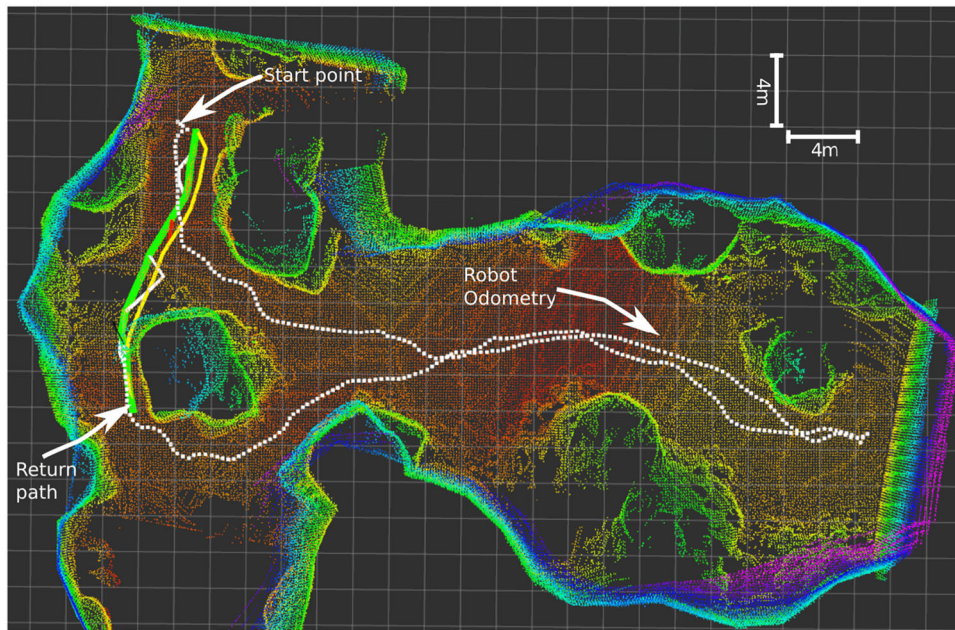
The sequential evolution of the map and the 3D meshes can be observed in Fig. 22a and b. Finally, in Fig. 22c, the final reconstructed environment is depicted with the odometry data estimated by the LiDAR-SLAM algorithm (white dotted line), and an example of the different path planning metrics in solid colors. During the planning step of the exploration, the robot used the combined metrics strategy defined in Eq. (5) to navigate (solid green line) with $P_d = 0.25$, $P_l = 0.50$ and $P_e = 0.25$. The density of the LiDAR-SLAM point cloud was defined as 20cm.



a Sequential process of mapping and navigation.



b Resulting map meshes for path planning at the previous steps.



c Top view of the complete map, depicting one instance of path planning (colored lines). Odometry is signaled as white dots.

Fig. 22 Inspection pipeline experiment in a simulated DARPA cave environment. Top: sequential coverage and the corresponding point clouds (a) and map meshes (b) used to calculate the paths. Bottom: top view (c) of the complete map. The robot odometry is depicted in

a white dotted line. The routes for every step are denoted in solid red (distance), yellow (energy consumption), white (traversability), and green (combined metrics) lines

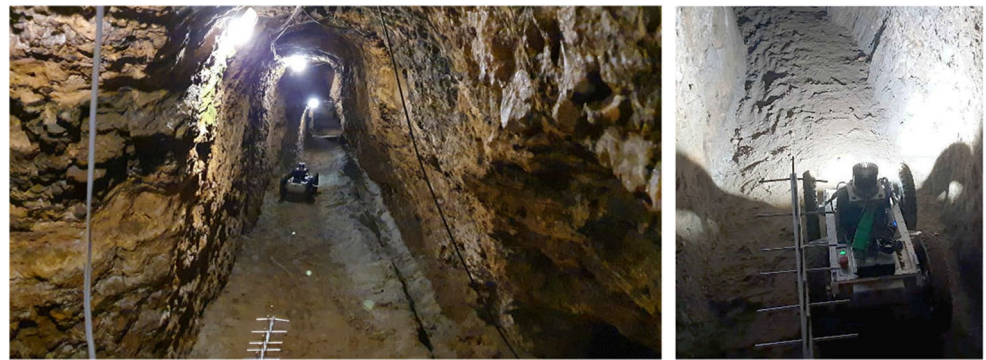
7.2 Real World Results

Real experiments were also performed with the Espele-oRobô platform in two representative scenarios: (i) a subterranean gold mine and (ii) an indoor multi-level scenario. In both experiments, we use a high range directional 900MHz

antenna for communication. The robot was equipped with an Ouster OS1 LiDAR (16 lines), an Xsens MTI-G-710 IMU, and an Intel RealSense Depth D435i camera.

The robot followed the combined metrics path (green path in Figs. 26c and 27c) and the weights were defined with $P_d = 0.80$, $P_t = 0.10$ and $P_e = 0.10$. The density of the

Fig. 23 Experimental setup of the EspeleRobô at the Mina du Veloso gold mine



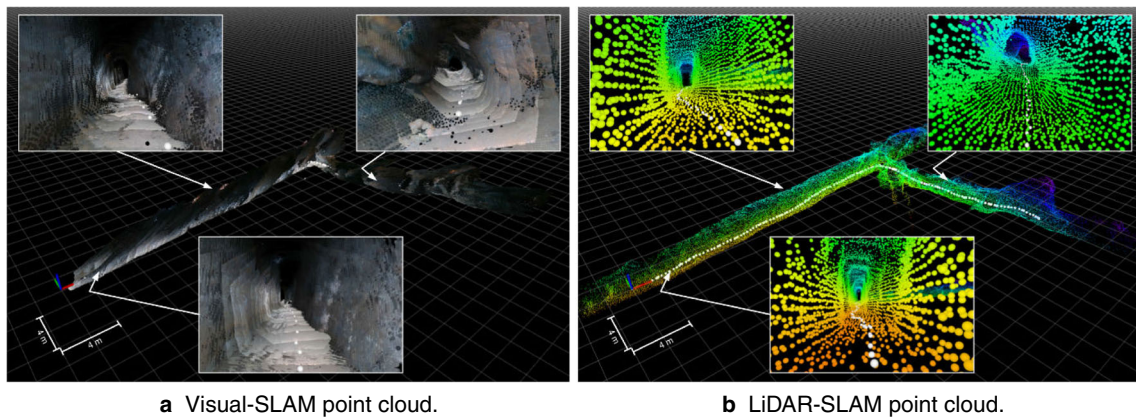
point-cloud for the LiDAR-SLAM algorithm was defined at 5 cm.

7.2.1 Subterranean Mine Mapping

The subterranean mine experiment was performed at the Mina du Veloso gold mine, located in Ouro Preto – MG, Brazil (20°22'34''S, 43°30'57''W). In Fig. 23 the environment and robot setup can be observed. Mina du Veloso is a

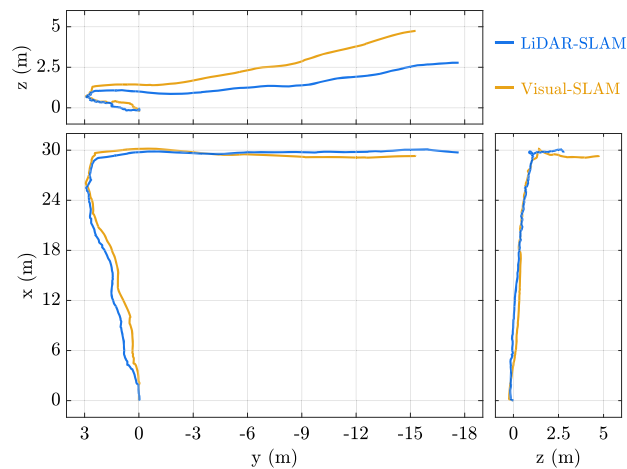
400-year-old colonial gold mine with almost 300 meters of narrow multi-level corridors, with rugged terrain and strong magnetic interference.

Figure 24 presents the final map’s inner and external views obtained with the online Visual and LiDAR SLAM algorithms, including the estimated odometry. A 100 W LED module from StratusLEDS adapted to the robot platform compensated for the lack of proper illumination inside the mine. The real-time odometry estimated by the SLAM



a Visual-SLAM point cloud.

b LiDAR-SLAM point cloud.



c Odometry estimation.

Fig. 24 Point cloud mapping results and visualization with the proposed online SLAM methods in a real subterranean Gold Mine (Mina du Veloso): **a** Visual-SLAM and **b** LiDAR-SLAM. In **c**, the odometry for both methods is depicted in blue and yellow solid lines

methods (Fig. 24c) shows similar accuracy at the X and Y axis with a more significant variation on the Z-axis.

A comparison of the point clouds generated by the three mapping methods is depicted in Fig. 25. In this case, given the precision and range of the LiDAR point cloud, we use it as the ground truth to compare the photogrammetry and Visual-SLAM methods. Different from the simulated results, the photogrammetry obtained the best performance: for 95% of the 154,881 cloud points, the errors were smaller than 0.29 m, the 3σ interval was 0.5 m with a maximum error of 2.06 m. The Visual-SLAM algorithm achieved less precise results, estimating a cloud with 1,235,708 points and an error of less than 1.276 m in 95% of these points, a 3σ interval of 1.896 m and a maximum error of 2.56 m.

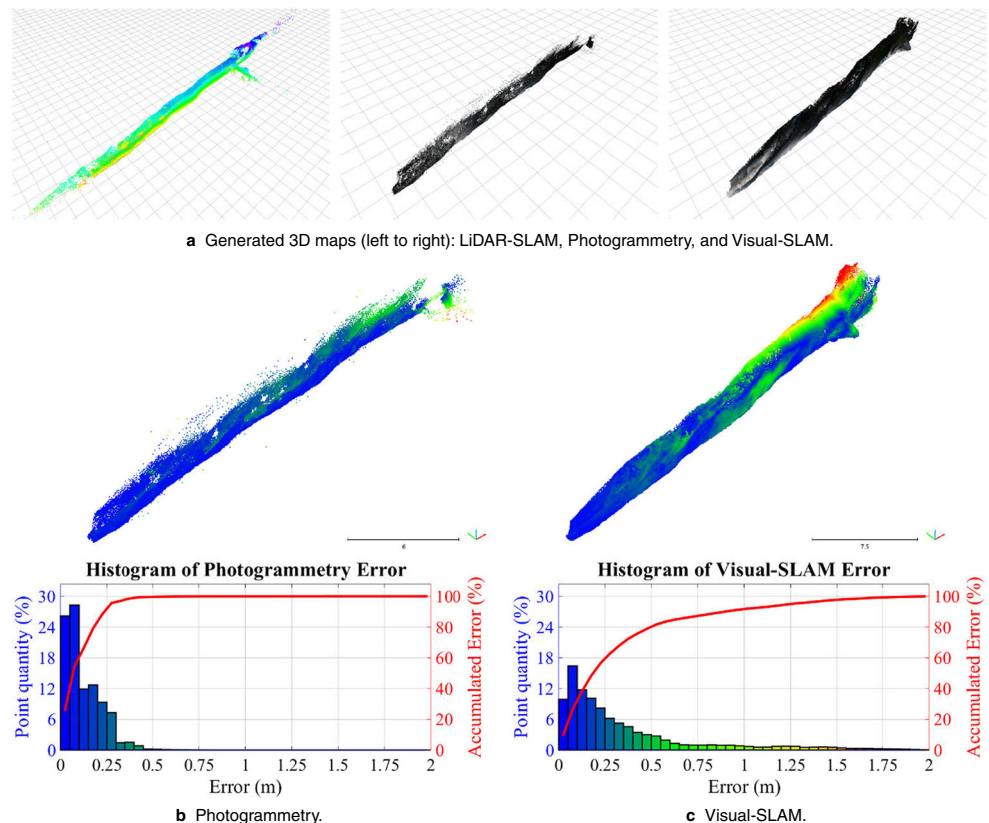
Although the photogrammetry technique presented better results than the Visual-SLAM in the real scenario, it is important to mention that a precise scale cannot be estimated correctly using only RGB images without an external reference. Thus, the resulting cloud was rotated and translated manually to match the LiDAR’s point-cloud and estimate an approximate comparison. The photogrammetry technique is more prone to errors due to rotational movements than the other technique and depends only on visual features, while the Visual-SLAM with RGB-D cameras could also use depth information to aid registration.

The photogrammetry reconstruction used 103 images of size 640x480 taking a total time of 22:18 minutes, on the other hand the Visual-SLAM spent 59 ± 41 ms per frame or 763 ± 200 ms per keyframe. The LiDAR SLAM took 715 ± 69 ms to process a sensor scan and generate the map update.

7.2.2 Autonomous Navigation in a Subterranean Mine

The navigation and inspection experiment results inside the Mina du Veloso gold mine can be observed in Fig. 26, where the robot performed mapping over ≈ 55 m of connected cave tunnels, autonomously. Figure 26a depicts the sequential point clouds generated by the LiDAR-SLAM algorithm, including a picture of the current scenario taken from the frontal RGB camera of the robot. Figure 26b shows the sequential mesh generation used for navigation and path planning. An overview of the complete map, including odometry, can be observed in Fig. 26c. Only the combined metric path (solid green line) was estimated in this experiment, given the previous validations’ results. This scenario was particularly challenging as the terrain was rugged and slippery, with narrow corridors including multiple small bumps, holes, and rocks. In this sense, the planning algorithm’s weights were defined to prioritize straighter collision-free paths to prevent any extra in-place rotations. The

Fig. 25 Point cloud error analysis. Comparison between the ground truth map and partial maps obtained with: **a** Photogrammetry, and **b** Visual-SLAM in the Mina du Veloso, using LiDAR SLAM map as reference



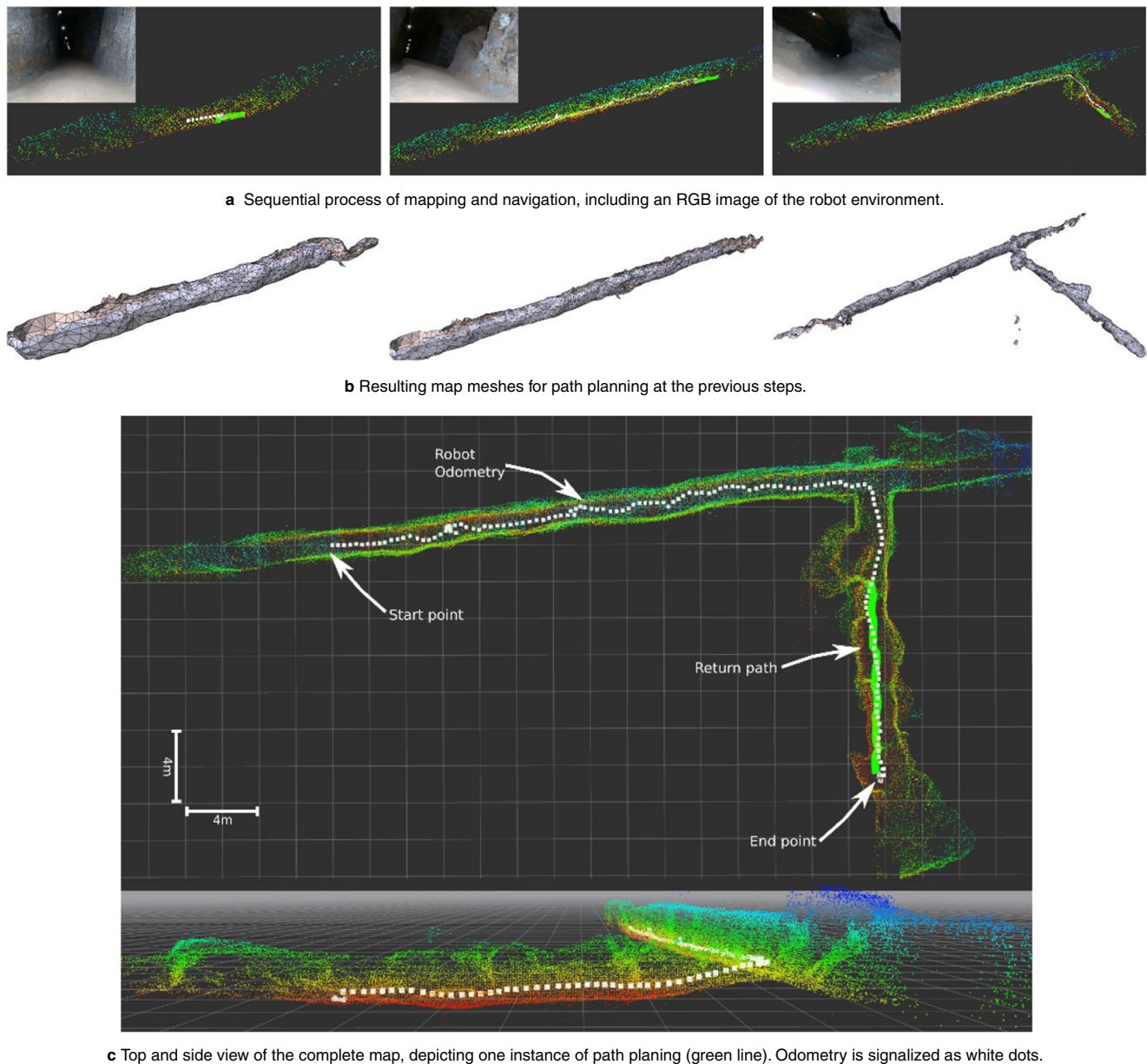


Fig. 26 Inspection pipeline real experiment at the Mina du Veloso gold mine. Top: sequential coverage (a) and the corresponding map meshes, (b) used to calculate the paths, with corresponding RGB images of the environment. Bottom (c): final point cloud of the

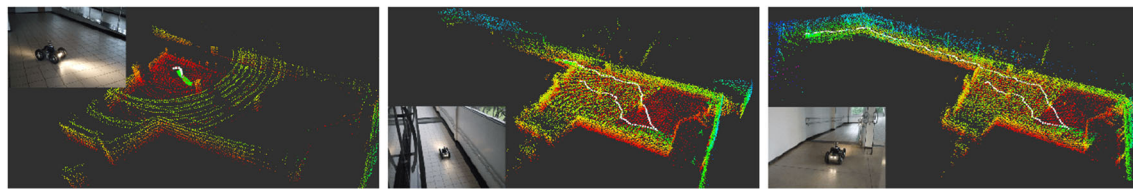
exploration pipeline (top and lateral view). In c the LiDAR odometry of the robot for the entire inspection mission is depicted in a white dotted line, and the navigation path for the combined metric is shown in green

increased cloud resolution (5 cm) used in this experiment helped detect and reconstruct the complex terrain more reliably, at the cost of an increased point cloud size.

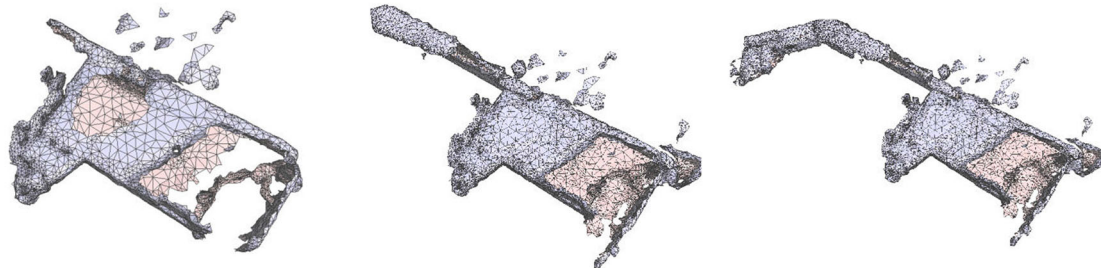
7.2.3 Autonomous Navigation in an Indoor Multi-level Scenario

The inspection and navigation results in an indoor scenario can be observed in Fig. 27. In this scenario, the robot

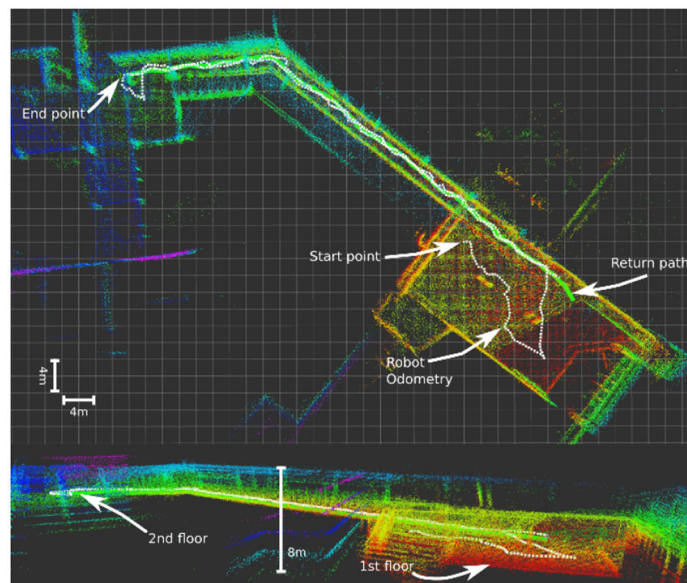
explored two floors joined by an inclined corridor. The robot performed over ≈ 80 m of autonomous navigation. This scenario was challenging due to the small and inclined corridor the robot must take to reach another floor, which validates the capacities of the 3D navigation of the proposed pipeline. The scenario presented flat and shiny surfaces with multiple windows and glass doors, which generated noise in the LiDAR sensor readings; however, the meshing algorithm correctly reconstructed the navigable areas.



a Sequential process of mapping and navigation, including an RGB image of the robot environment.



b Resulting map meshes for path planning at the previous steps.



c Top and lateral view of the final map, depicting one instance of path planning returning to the home base (green line). Odometry is signalized as white dots.

Fig. 27 Mapping, navigation, and control pipeline real experiment at an indoor multilevel scenario. Top: sequential coverage (a) and the corresponding map meshes (b) used to calculate the paths, with corresponding RGB images of the environment. Bottom (c): final

point cloud of the exploration pipeline (top and lateral view). In (c) the LiDAR odometry of the robot for the entire inspection mission is depicted in a white dotted line, and the navigation path for the combined metrics is shown in green

7.3 Planning Performance Analysis

To evaluate the navigation pipeline’s performance, we measured the mesh reconstruction time for multiple point clouds extracted iteratively from the LiDAR-SLAM algorithm when performing an exploration mission using a laptop with an Intel i7-4810MQ CPU with 16GB of RAM. Figure 28 shows the mesh generation times for an indoor scenario that extends to over ≈ 80 meters. Every data point is the mean of 3 reconstructions, and the standard deviation is depicted as a light blue shadow. It is possible to observe that the first meshes are generated relatively fast

in less than 20 seconds. The largest cloud takes 96 seconds, allowing the algorithm’s execution directly on the robot’s embedded computer online. The results show a near-linear relationship between the cloud size and the time to perform the reconstruction.

The planning step, performed after the mesh reconstruction from the point cloud, transforms the mesh on a graph where the Dijkstra search algorithm finds the more suitable path from the current robot’s position to the desired end-point considering multiple terrain metrics. Figure 29 shows the time spent searching the path over an increasing graph size at variable path lengths. Results showed that the time

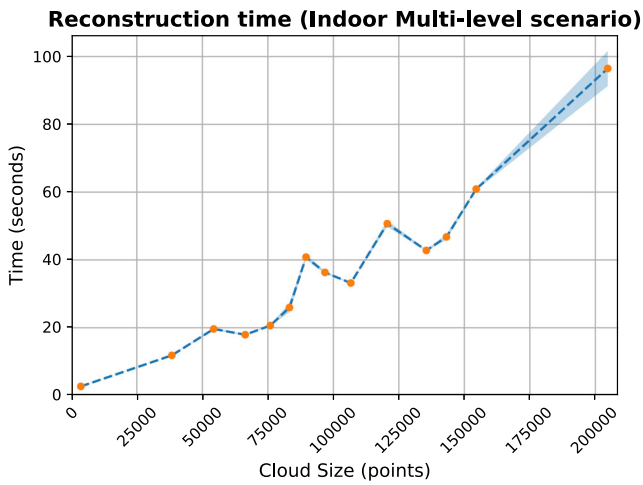


Fig. 28 Mean reconstruction time for the mesh algorithm over multiple iteratively generated point clouds for a real exploration experiment on an indoor scenario that extends for over ≈ 80 meters. Every data point is the mean time of 3 runs. The standard deviation is depicted by the shadow in light blue

spent in planning is more related to the graph’s size than the path size, as shown by the linear time growth despite the variable path size (color of the dots). This procedure was relatively fast, at a maximum of ≈ 18 seconds for the worst-case on a complete exploration experiment. The time spent on path planning could be increased or decreased, given the graph density, original point cloud size, or mesh filtering.

7.4 Discussion

The proper selection of one mapping or SLAM method over another is particularly relevant in the proposed system. In this sense, photogrammetry reconstruction in this work’s context is a computationally intensive procedure that

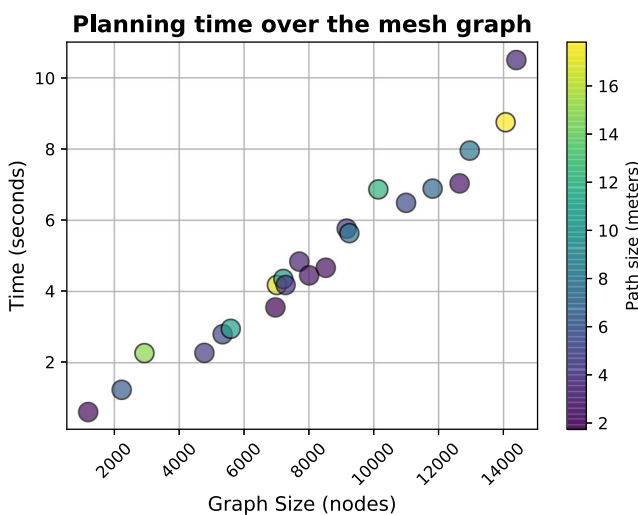


Fig. 29 Mean planning time for the navigation algorithm over multiple iteratively generated environment graphs. Every data point is the mean time of 3 runs. The color bar represents the path size

generates point clouds only from RGB images. This method is more suitable for offline processing after the exploration mission is already performed and the robot is back safely into the base. Since the computational resources embedded in the robot are limited in terms of speed, space, thermal, and power consumption, performing intensive and long computations directly on the embedded computer is not recommended. Also, the point-cloud generated by the photogrammetry for this application, does not have a reference scale, so we need to translate the cloud to match real-world proportions and position.

On the other hand, Visual-SLAM is a more optimized method for visual map generation and localization that could be executed directly on the robot. Visual-SLAM and the photogrammetry methods use visual features to recreate a realistic map of the environment in the sense of colored point clouds. Both methods are highly dependent on good illumination to be able to extract features and perform registration. However, Visual-SLAM is not limited to only RGB images and generally uses RGB-D sensors generating more dense clouds, including sensors that are less dependent on external illumination, such as structured light sensors.

The LiDAR-SLAM method uses a 3D 360 depth sensor that is less affected by illumination changes and has a significantly increased depth range. Since the LiDAR-SLAM method is also paired with an IMU, its more robust to the robot’s rotational motions than the other visual mapping algorithms. Nevertheless, this method relies only on 3D structural features to perform the cloud registration, meaning that very similar structures (such as the ones found inside pipes and galleries, for example) could not perform correct registration even in the presence of rich visual features such as joints, small defects or markings on the walls.

Generally, the LiDAR-SLAM is preferred in most of the scenarios studied in this work since the range, precision, and illumination robustness helps with the path planning and navigation. In locations where the LiDAR-SLAM is prone to failure, the Visual-SLAM method could be used given that the local has enough illumination or visual features to track. However, in both cases, it is generally desirable to perform a photogrammetry reconstruction with the RGB images captured by the robot to improve the structures’ analysis, since the LiDAR alone can not provide colored point clouds or texture.

Although the path planning successfully generated collision-safe paths at all of the tested environments, a particularly critical step of the pipeline is the meshing procedure. The mesh reconstruction algorithm could generate unusable mesh results depending on sensory noise or inaccurate measurements (as in indoor scenarios with glossy walls and floors). Thus, a filtering step could be necessary to increase robustness for noisy scenarios where the sensor

could not perform efficiently. For considerably larger maps, the pipeline could use only a sub-segment of the point cloud to allow the visitation of nearby regions quickly and prevent wasting computing time for parts of the map that are not intended for visitation in that timestep.

8 Conclusions

This paper describes a mobile robot for inspection services in confined areas. The EspeleoRobô has already executed several inspections and mapping services in the field through wireless and wired communication. Despite being initially designed for inspection of natural caves, EspeleoRobô proved to be a highly versatile device, inspecting other confined areas, including drain ducts and dam galleries. Aiming autonomous operations, we proposed a set of strategies for localization and mapping, path planning, and navigation control.

Since image data and 3D maps are the main product of inspections in confined spaces, we proposed both offline (photogrammetry) and online (RTAB-Map and LeGO-LOAM) reconstruction methodologies that work in these types of GPS denied scenarios. We validated our methodology in simulated experiments with a virtual cave (Darpa SubT subterranean mine), in a real subterranean mine scenario (Mina du Veloso), and an indoor multi-level environment. Results demonstrate the applicability of those solutions showing that adequate reconstruction can be achieved with proper embedded illumination systems even in poor illumination conditions. Those realistic maps can also be used as virtual training scenarios for robot operators or in simulations. Some novel uses for three-dimensional maps for inspections include virtual reality immersion devices.

The LiDAR-SLAM experiments showed the most reliable and precise maps of the three methods, considering the shortest steady-state error interval and maximum errors. This algorithm was also able to give a precise online localization to feedback the navigation strategy with a Vector-Field control.

Based on the terrain map, we can plan optimal routes for the robot. We adopt multiple metrics for path generation such as traveled distance, terrain traversability, energy consumption, and a trade-off between the conflicting objectives. The proposed navigation method is evaluated through simulations and real experiments of subterranean mines and indoor scenarios, corresponding to an initial required step for autonomous operations. The proposed artificial Vector field-based control was efficient in guiding the robot through the planned paths and the LiDAR localization was used to feedback the system.

The results obtained demonstrated the applicability of a semi-autonomous pipeline for inspection and mapping

confined spaces with a ground robot. The presented methodology and validations allow the robot platform to have an initial autonomy level by performing a considerable part of the inspection mission in an automated fashion.

8.1 Future Work

Future work will focus on the EspeleoRobô autonomous operation. Exploration is a requirement for autonomous operation in unknown environments. In this sense, we plan to adopt autonomous exploration methods for the current navigation pipeline, such as using information gain or other related metrics [59]. Optimization methods could also be applied to determine the best set of navigation weights for each particular environment. In this sense, other optimizations could also be implemented to improve the pipeline's processing time. For example, the use of an intelligent selection of which portions of the point cloud are used for mesh reconstruction. Also, simpler environments could benefit from using a sparser point cloud without compromising the terrain reconstruction faithfulness. An algorithm that dynamically changes the point cloud's density given the environment state could also improve processing times. Adaptations could reduce the final navigation graph size by exploring particular robot architectures and dynamics.

Robot control in subterranean environments is very challenging due to wet and slippery terrains. In this sense, the use of dynamic control methods, which account for the effects of these terrain features on the robot and compute the motors' torque to compensate for these disturbances, can improve the robot's locomotion. The robotic platform could also benefit from hardware improvements such as selecting the best locomotion mechanism – wheel type and diameter – for the mission's expected terrain. Teleoperation could also benefit from multiple cameras located carefully to visualize critical areas of the robot prone to collisions.

Other research lines, such as robotic cooperation, can also enhance the platform's current mapping capabilities by subdividing the inspection tasks among a team of robots. Heterogeneous cooperation could benefit from using robots with different capabilities, i.e., aerial and terrestrial, to explore areas unreachable to one type of robot.

Author Contributions General work conduction: H.A. and G.M.F.; Conceptualization: H.A., G.P. and G.M.F.; Robot platform: L.G.D.B., H.A., J.D., F.R. and G.M.F.; Online SLAM: G.P.C.J., R.F. and G.M.F.; Photogrammetry and mesh reconstruction: G.P., H.A., L.W.R.F. and E.R.N.; Navigation: H.A., F.L.M.S., G.M.F. and D.G.M.; Control: A.R., V.M., H.A. and G.M.F.; Experimental methodology: H.A., A.R., G.P., G.P.C.J., R.F., V.M., L.W.R.F., J.D., F.R., F.L.M.S. and G.M.F.; Conceived images and graphics: H.A., A.R., G.P., G.P.C.J., R.F., V.M., L.W.R.F., F.L.M.S. and L.G.D.B.; Work supervision: G.P. and G.M.F.; All authors reviewed and wrote the manuscript.

Funding

- Instituto Tecnológico Vale (ITV);
- Vale S.A.;
- Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq);
- Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG);
- Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

Availability of Data and Material Algorithms are already published on GitHub. Data-sets can be made available upon acceptance and publication.

Declarations

Conflict of Interests The authors declare no competing interests.

References

1. Booz, A.: Unearthing the subterranean environment. <https://www.subtchallenge.com/> (2018)
2. Murphy, R.R.: Disaster robotics (2014)
3. Azpurua, H., Rocha, F., Garcia, G., Santos, A.S., Cota, E., Barros, L.G., Thiago, A.S., Gustavo P., Gustavo M.F.: EspeleoRobô - a robotic device to inspect confined environments. In: 2019 19th Int. Conf. on Advanced Robotics (ICAR), IEEE (2019)
4. Saranlı, U., Buehler, M., Koditschek, D.E.: Rhex: A simple and highly mobile hexapod robot. *Int. J. Robot. Res.* **20**(7), 616–631 (2001)
5. Freitas, G.M., Rocha, F.A.S., Torre, M.P., Frederico, F.J.A., Ramos, V.R., Nogueira, L.E.C., Santos, A.S., Cota, E., Miola, W., Reis, M.A.D., Costa, B.L.S., Ledezma, L.C.M., Evangelista, R.P., Alcantara, P.X., Lima, R.T., De Souza, T.P., Brandi, I.V., Araujo, R.N., Gomes, M.F.M., Garcia, G.C.: Multi-terrain inspection robotic device and methods for configuring and guiding the same - pct/br2018/050025 (2018)
6. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: Ros: an open-source robot operating system. In: ICRA Workshop, vol. 3, pp. 5. Kobe (2009)
7. RocketM Datasheet: Ubiquiti Networks Inc. https://dl.ubnt.com/datasheets/rocketm/RocketM_DS.pdf (2015)
8. Antena móvel ASA-900CI: ARS Eletrônica Industrial Ltda. <http://www.arseletronica.com.br/fileuploader/download/download/?d=0&file=custom%2Fupload%2FFile-1562681992.pdf> (2018)
9. WiFi for Industrial Internet of Things – Reseiwe: RESEIWE A/S, <https://reseiwe.com/wifi-for-industrial-internet-of-things-2/> (2017)
10. Murphy, R.R., Tadokoro, S., Kleiner, A.: Disaster robotics. In: Springer Handbook of Robotics, pp. 1577–1604. Springer (2016)
11. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **32**(6), 1309–1332 (2016)
12. Siciliano, B., Khatib, O.: Springer handbook of robotics. Springer, Berlin (2016)
13. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping: part i. *IEEE Robot. Automat. Mag.* **13**(2), 99–110 (2006)
14. Fuentes-Pacheco, J., Ascencio, J., Rendon-Mancha, J.: Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev.* **43**, 11 (2015)
15. Filipenko, M., Afanasyev, I.: Comparison of various slam systems for mobile robot in an indoor environment. In: Int. Conf. on Intelligent Systems (IS), pp. 400–407. IEEE (2018)
16. Hess, W., Kohler, D., Rapp, H., Andor, D.: Real-time loop closure in 2d lidar slam. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1271–1278. IEEE (2016)
17. Droschel, D., Behnke, S.: Efficient continuous-time slam for 3d lidar-based online mapping. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–9. IEEE (2018)
18. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: European Conf. on Computer Vision, pp 834–849, Springer (2014)
19. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. *IEEE Trans. Robot.* **31**(5), 1147–1163 (2015)
20. Labbé, M., Michaud, F.: Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *J. Field Robot.* **36**(2), 416–446 (2019)
21. Labbe, M., Michaud, F.: Appearance-based loop closure detection for online large-scale and long-term operation. *IEEE Trans. Robot.* **29**(3), 734–745 (2013)
22. Rezende, A.M.C., Júnior, G.P.C., Fernandes, R., Miranda, V.R.F., Azpurua, H., Pessin, G., Gustavo, M.F.: Indoor localization and navigation control strategies for a mobile robot designed to inspect confined environments. In: 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), pp. 1427–1433. IEEE (2020)
23. Yagfarov, R., Ivanou, M., Afanasyev, I.: Map comparison of lidar-based 2d slam algorithms using precise ground truth. In: 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 1979–1983. IEEE (2018)
24. Ji, Z., Singh, S.: Low-drift and real-time lidar odometry and mapping. *Auton. Robot.* **41**(2), 401–416 (2017)
25. Koide, K., Miura, J., Menegatti, E.: A portable three-dimensional lidar-based system for long-term and wide-area people behavior measurement. *Int. J. Adv. Robot. Syst.* **16**(2), 1729881419841532 (2019)
26. Tixiao Shan, Brendan Englot: Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In: 2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp 4758–476., IEEE (2018)
27. Wolf, P.R., Dewitt, B.A., Wilkinson, B.E.: Elements of photogrammetry with applications in GIS, 4th ed. 696 pp. ISBN 9780071761123 (2014)
28. Low, D.G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 91–110. <https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf> (2004)
29. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: Speeded-up robust features (surf). *Comput. Vis. Image Underst.* **110**(3), 346–359 (2008)
30. Wu, C., Agarwal, S., Curless, B., Seitz, S.M.: Multicore bundle adjustment. In: CVPR 2011, pp 3057–306. IEEE (2011)
31. Agarwal, S., Mierle, K., et al.: Ceres solver. <http://ceres-solver.org>
32. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 2, pp. 2161–2168 (2006)
33. Moulon, P., Monasse, P., Marlet, R.: Global fusion of relative motions for robust, accurate and scalable structure from motion. In: 2013 IEEE International Conference on Computer Vision, pp. 3248–3255 (2013)
34. Hirschmuller, H.: Accurate and efficient stereo processing by semi-global matching and mutual information. In: 2005 IEEE

- Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol 2, pp 807–814 (2005)
35. Jancosek, M., Pajdla, T.: Exploiting visibility information in surface reconstruction to preserve weakly supported surfaces. *International Scholarly Research Notices*, 2014: 1–20. ISSN 2356-7872 (2014)
 36. Bruno, L., Sylvain, P., Nicolas, R., Maillot, J.: Least squares conformal maps for automatic texture atlas generation. In: *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '02*, pp. 362–371 (2002)
 37. AliceVision: Meshroom: A 3D reconstruction software. <https://github.com/alicevision/meshroom> (2018)
 38. Moulon, P., Monasse, P., Renaud, M.: Adaptive structure from motion with a contrario model estimation. In: *Proceedings of the Asian Computer Vision Conference (ACCV 2012)*, pp. 257–270. Springer, Berlin (2012)
 39. Shewchuk, J.R.: Delaunay refinement algorithms for triangular mesh generation. *Comput. Geom.* **22**(1-3), 21–74 (2002)
 40. Shekhar, R., Fayyad, E., Yagel, R., Cornhill, J.F.: Octree-based decimation of marching cubes surfaces. In: *Proceedings of the 7th conference on Visualization'96*, pp. 335–ff IEEE Computer Society Press (1996)
 41. Kazhdan, M., Klein, A., Dalal, K., Hoppe, H.: Unconstrained isosurface extraction on arbitrary octrees. *Symp. Geom. Process.* **7**, 256–263 (2007)
 42. Carr, J.C., Beatson, R.K., McCallum, B.C., Fright, W.R., McLennan, T.J., Mitchell, T.J.: Smooth surface reconstruction from noisy range. In: *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pp 119–ff ACM (2003)
 43. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. *ACM Trans. Graph. (TOG)* **32**(3), 29 (2013)
 44. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. *26, ACM* (1992)
 45. Fabri, A., Sylvain, P.: CGAL The Computational geometry algorithms library. In: *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pp. 538–53. ACM (2009)
 46. Liepa, P.: Filling holes in meshes. In: *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on geometry processing*, pp. 200–205. Eurographics Association (2003)
 47. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: *Proceedings of the 1968 23rd ACM national conference*, pp. 517–524. ACM (1968)
 48. Phong, B.T.: Illumination for computer generated pictures. *Commun. ACM* **18**(6), 311–317 (1975)
 49. Macenski, S., Martin, F., White, R., Clavero, J.G.: The marathon 2: A navigation system. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020)
 50. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **30**(7), 846–894 (2011). <https://doi.org/10.1177/0278364911406761>
 51. Miranda, V.R.F., Mozelli, L.A., Neto, A.A., Freitas, G.M.: On the robust longitudinal trajectory tracking for load transportation vehicles on uneven terrains. In: *2019 19th International Conference on Advanced Robotics (ICAR)*, pp 320–325. IEEE (2019)
 52. Sousa, R.L.S., Forte, M.D.N., Nogueira, F.G., Torrico, B.C.: Trajectory tracking control of a nonholonomic mobile robot with differential drive. In: *IEEE Biennial Congress of Argentina (ARGENCON)*, pp. 1–6. IEEE, p. 2016 (2016)
 53. Cho, N., Kim, Y., Park, S.: Three-dimensional nonlinear differential geometric path-following guidance law. *J. Guid. Control Dyn.* **38**(12), 2366–2385 (2015). <https://doi.org/10.2514/1.G001060>
 54. Gonçalves, V.M., Pimenta, L.C.A., Maia, C.A., Dutra, B.C.O., Pereira, G.A.S.: Vector fields for robot navigation along time-varying curves in n-dimensions. *IEEE Trans. Robot.* **26**(4), 647–659 (2010). ISSN 1552-3098
 55. Rezende, A.M.C., Júnior, G.P.C., Fernandes, R., Miranda, V.R.F., Azpúrua, H., Pessin, G., Freitas, G.M.: Indoor localization and navigation control strategies for a mobile robot designed to inspect confined environments. In: *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pp. 1427–1433 (2020)
 56. Santos, A.S., Azpúrua, H.I.P., Pessin, G., Freitas, G.M.: Path planning for mobile robots on rough terrain. In: *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and Workshop on Robotics in Education (WRE)*, pp. 265–270. IEEE, p. 2018 (2018)
 57. Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: *Robotics: modelling, planning and control*. Springer Science & Business Media (2010)
 58. Chaimowicz, L., Michael, N., Kumar, V.: Controlling swarms of robots using interpolated implicit functions. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. pp. 2487–2492 (2005)
 59. Carrillo, H., Dames, P., Kumar, V., Castellanos, J.A.: Autonomous robotic exploration using a utility function based on rényi's general theory of entropy. *Auton. Robot.* **42**(2), 235–256 (2018)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Héctor Azpúrua is currently pursuing a Ph.D. in the Computer Vision and Robotics Laboratory (VeRLab) at the Department of Computer Science (DCC) of the Universidade Federal de Minas Gerais (UFMG) in Brazil and is an assistant researcher at the Instituto Tecnológico Vale (ITV). He has a bachelor's in Computer Science from Universidad Nueva Esparta (2011) and has an MSc in Robotics from UFMG (2015). His research interest includes Robotics, Artificial Intelligence, and Digital Security, emphasizing mobile robotics, machine learning, and real-time strategy games. He has worked in diverse research projects between academia and industry.

Adriano Rezende received the B.S. in electrical engineering from the Universidade Federal de Minas Gerais (UFMG) in 2016, Brazil, with one year in the Colorado State University (UFMG), USA. At UFMG, he completed a master's degree in electrical engineering in 2019, and he is currently working toward a Ph.D., also in electrical engineering. His current research interests include ground and aerial mobile robotics, control theory, path planning, and optimization.

Guilherme Potje received his B.Sc. degree in Computer Science from the University of Western São Paulo, Brazil (2013), and his M.Sc. degree, also in Computer Science, from the Federal University of Minas Gerais (UFMG), Brazil (2016). Currently, he is pursuing a Ph.D. degree in Computer Science at UFMG. His interests include Artificial Intelligence, Machine Learning, and Computer Vision. His main research topics are Image Registration & Matching and 3D Reconstruction.

Gilmar Pereira da Cruz Júnior has a Bachelor's degree in Control Automation Engineering from the Federal University of Pelotas (2018) in Brazil, focused on Stewart Platform Design and Modeling adapted for delivery services use. He is currently finishing his M.Sc in Electrical Engineering from the Department of Electrical Engineering (DEE) of the Federal University of Minas Gerais. He has a professional formation in Instrumentation and Process Control Technical from the Federal Institute of Minas Gerais (2008). His research interest includes mobile robotic, simultaneous localization and mapping, robot movements planning, adaptive filter, and robotic manipulator.

Rafael Fernandes is an MSc student in Control, Robotics, and Automation at the Graduate Program of Electrical Engineering (PPGEE) of the Universidade Federal de Minas Gerais (UFMG) in Brazil. He has worked with methods for visual reconstruction of confined environments using RGB-D cameras. Rafael has a bachelor's degree in Electrical Engineering from the Universidade Federal de Viçosa, Brazil (2017), when he participated in robot soccer competitions, acquiring experience in image processing and artificial intelligence. His research interests include computer vision, digital image processing, mobile robotic, simultaneous localization and mapping, and artificial intelligence.

Victor Miranda is a Ph.D. student in the graduate program in Electrical Engineering (PPGEE) of the Federal University of Minas Gerais (UFMG). He has a bachelor's in Mechatronics Engineering from the Federal University of São João Del Rei (UFSJ) in 2017 and has an M.Sc. in Electrical Engineering from UFMG in 2019. His main research interests include nonlinear and robust control theory, mobile robotics, multi-agent systems, motion planning, localization, and filtering.

Levi Wellington de Resende Filho is currently a Researcher Engineer working in robotics projects by the Federal University of Rio de Janeiro in Brazil and is finishing his master's degree in the Instituto Tecnológico Vale and the Federal University of Ouro Preto. His research interest includes Mobile Robotics, Computer Vision, Industry 4.0, Remote Sensing and Project Management, with emphasis on applied research projects for the mining and oil and gas industries.

Jacó Domingues is an MSc student in Instrumentation, Control, and Automation of Mining Processes (PROFICAM) of University Federal of Ouro Preto (UFOP) and Vale Institute of Technology (ITV). He has a bachelor's in Control and Automation Engineering from UFOP (2020). He is currently a technician in the Robotics Laboratory of ITV, where he works developing robots for industrial robot inspection.

Filipe Rocha has M.E. in Instrumentation, Control and Automation of Mining Processes (Escola de Minas, UFOP / Instituto Tecnológico Vale - 2018), and B.E. in Control and Automation Engineering at (Escola de Minas, UFOP - 2016) with foreign period at École Nationale d'Ingénieurs de Brest (ENIB, Brest, France). He is currently a Ph.D. student at COPPE Electrical Engineering Program (Universidade Federal do Rio de Janeiro).

Frederico Luiz Martins de Sousa Currently, he is a Computer Science B.S. undergraduate student at Universidade Federal de Ouro Preto. His current research interests include Deep Learning, Computer Vision, Robotics, and Industry 4.0.

Luiz Guilherme Dias de Barros is a Mechanical Engineer graduated from Universidade Federal de Ouro Preto (UFOP) and a Master's degree student in Instrumentation, Control, and Automation of Mining Process at Instituto Tecnológico Vale (ITV) and UFOP. Luiz is also a Researcher at ITV focused on the development of industrial applications using robotic devices.


Erickson R. Nascimento is an Associate Professor in the Department of Computer Science at the Federal University of Minas Gerais (UFMG), Belo Horizonte, Brazil. He holds B.S., M.Sc., and D.Sc. degrees in Computer Science, all from UFMG. He was also a visiting researcher at The Berkeley Artificial Intelligence Research (BAIR) of the University of California Berkeley. His research interests include Computer Vision and Pattern Recognition. In particular, egocentric video processing, three-dimensional reconstruction and modeling, keypoint descriptors, and underwater vision.

Douglas G. Macharet is an Associate Professor at the Department of Computer Science (DCC) of the Universidade Federal de Minas Gerais (UFMG). He received an M.Sc. and a D.Sc. degree in Computer Science from the same university in 2009 and 2013, respectively, and in 2019 he was a visiting researcher at the GRASP Laboratory at the University of Pennsylvania. He is with the Computer Vision and Robotics Laboratory (VeRLab), and his main research interests are in mobile robotics, considering topics such as motion planning and autonomous navigation; multi-robot systems and swarm robotics; and human-robot interaction.

Gustavo Pessin got his D.Sc. in Computer Science at the University of Sao Paulo as a member of the Mobile Robotics Lab. During his D.Sc. Pessin carried out research with the Robotics Lab, at the Heriot-Watt University, Edinburgh, UK and the Communication and Distributed Systems Group, at the Universität Bern, Switzerland. In 2015, Pessin had a Visiting Scholar position within the Media Lab at the Massachusetts Institute of Technology. Currently, Pessin is an Associated Researcher within the Robotics Lab, at the Vale Institute of Technology. The bulk of his research is related to intelligent systems and mobile robots.

Gustavo M. Freitas received his B.Sc. degree in Control and Automation Engineering from the Universidade Federal de Santa Catarina and the M.Sc. and D.Sc. degrees in Control, Automation, and Robotics from the Universidade Federal do Rio de Janeiro, Brazil. His career has focused on robotics applied to environmental monitoring, agriculture, mining, and oil and gas production systems through his work at the Carnegie Mellon University - Field Robotics Center, Petrobras Robotics Laboratory, and Instituto Tecnológico Vale Robotics Laboratory, Brazil. Currently, Dr. Freitas is a professor at the Electrical Department (DEE) of the Universidade Federal de Minas Gerais, Brazil.

Affiliations

Héctor Azpúrua^{1,2}  · Adriano Rezende³ · Guilherme Potje¹ · Gilmar Pereira da Cruz Júnior³ · Rafael Fernandes³ · Victor Miranda³ · Levi Wellington de Resende Filho² · Jacó Domingues² · Filipe Rocha² · Frederico Luiz Martins de Sousa² · Luiz Guilherme Dias de Barros² · Erickson R. Nascimento¹ · Douglas G. Macharet¹ · Gustavo Pessin² · Gustavo M. Freitas³

Adriano Rezende
adrianomcr@ufmg.br

Guilherme Potje
guiopotje@dcc.ufmg.br

Gilmar Pereira da Cruz Júnior
gilmarpcjunior@ufmg.br

Rafael Fernandes
rafaelfgs@ufmg.br

Victor Miranda
victormrfm@ufmg.br

Levi Wellington de Resende Filho
levi.filho@pq.itv.org

Jacó Domingues
jaco.domingues@itv.org

Filipe Rocha
filipe.rocha@itv.org

Frederico Luiz Martins de Sousa
frederico.sousa@pq.itv.org

Luiz Guilherme Dias de Barros
luiz.barros@pq.itv.org

Erickson R. Nascimento
erickson@dcc.ufmg.br

Douglas G. Macharet
doug@dcc.ufmg.br

Gustavo Pessin
gustavo.pessin@itv.org

Gustavo M. Freitas
gustavomfreitas@ufmg.br

¹ Computer Science Department, Universidade Federal de Minas Gerais, Belo Horizonte, 31270-901, Brazil

² Instituto Tecnológico Vale (ITV), Ouro Preto, 35400-000, Brazil

³ Electrical Engineering Department, Universidade Federal de Minas Gerais, Belo Horizonte, 31270-901, Brazil