Expert Systems With Applications 241 (2024) 122561

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Multisensory integration for topological indoor localization of mobile robots in complex symmetrical environments

Sergio Lafuente-Arroyo *, Saturnino Maldonado-Bascón, Diego Delgado-Mena, Carlos Gutiérrez-Álvarez, Francisco Javier Acevedo-Rodríguez

Alcalá University, Signal Theory and Communications Department, Alcalá de Henares, 28805 Madrid, Spain

ARTICLE INFO

Keywords:
Indoor global localization
Autonomous mobile robot
Topological map
Complex symmetrical environments

ABSTRACT

Indoor localization is essential for robotic navigation by using different sensors on board. Specifically, visual localization with a single camera is a great challenge in highly symmetric environments (e.g. offices, hospitals or residences), where appearance patterns are repetitive and captures from different locations provide very similar images. To overcome this issue, in this paper, we present a method that integrates multisensory information from an RGB-D camera, a LiDAR and motor encoders. Our approach simultaneously utilizes spatial consistency from a reference topological map and temporal consistency from time-series observations. Inspired by human cognitive perception, we define a two layered topological architecture that encompasses both coarse information of object distributions and structural information with some metric references. Categories of common objects in the environments, such as fire extinguishers or doors, are used as natural beacons. We evaluated our approach in two real-world buildings based on a multi-aisle structure with corridors of very similar appearance. Results demonstrated accurate localization despite the high level of symmetry of the scenario, and how ambiguity was significantly reduced as the agent progressed along its trajectory.

1. Introduction

Mobile autonomous robots need to manage information about their workspace to perform meaningful activities. In recent years, they have attracted a great deal of interest due to their numerous applications in different sectors such as transport, logistics or healthcare. In contrast to outdoor robotics (Li et al., 2017), indoor robotics mainly needs to overcome the lack of reliable position. It is caused by the non-availability of GPS reception as well as the usually smaller error margins enforced by the smaller size of the working area.

The location of the robot, also known as its pose, is represented by a position and orientation in a coordinate frame. Autonomous navigation relies on a good localization system dealing with the kidnapped robot problem and relocating the robot in case of pose tracking failure. Robotic localization involves two steps: global localization (GL) and local pose tracking (re-localization). The GL problem aims to estimate where a robot is placed somewhere in the environment and has to localize itself from scratch. It is different from the local localization problem, which tracks the agent over time based on GL information and assumes GL is known a priori. The assumption of GL initialization by the user has led most research to focus on the re-localization problem.

As a standard framework, the Robot Operating System ROS (Maruyama et al., 2016; Quigley et al., 2009) includes the package *amcl*, which allows the robot to be able to self-relocate on a reference map but it requires a previous initialization. It is based on an Adaptive Monte Carlo Localization (AMCL) (Dellaert et al., 1999) model. This approach works by comparing LiDAR measurements to the readings that would be expected for each of the poses, according to a reference map. This algorithm generates random samples (particles) of the current state from a priori probability. These particles are updated according to sensor measurements. The AMCL algorithm has been widely used in the literature with different variations: particle swarm optimization (Zhang et al., 2019), integration of text information and laser scan data (Text-MCL) (Ge et al., 2022), and a dual-timescale approach for highly dynamic environments (Valencia et al., 2014). However, the conventional AMCL method may be sufficient for robot localization in confined environments (Talwar & Jung, 2019) but, due to the high number of required particles, it is hardly scalable

Abbreviations: GL, Global localization; ROS, Robot Operating System; AMCL, Adaptive Monte Carlo localization; GT, Ground truth; SVM, Support vector machine; AEL, Accumulative error localization; CBIR, Content-based image retrieval; MSE, Mean squared error; SSIM, Structural similarity index measure

* Corresponding author.
E-mail addresses: sergio.lafuente@uah.es (S. Lafuente-Arroyo), saturnino.maldonado@uah.es (S. Maldonado-Bascón), diego.delgadom@uah.es (D. Delgado-Mena), carlos.gutierrezalva@uah.es (C. Gutiérrez-Álvarez), javier.acevedo@uah.es (F.J. Acevedo-Rodríguez).

https://doi.org/10.1016/j.eswa.2023.122561
Received 13 July 2023; Received in revised form 10 November 2023; Accepted 10 November 2023
Available online 20 November 2023

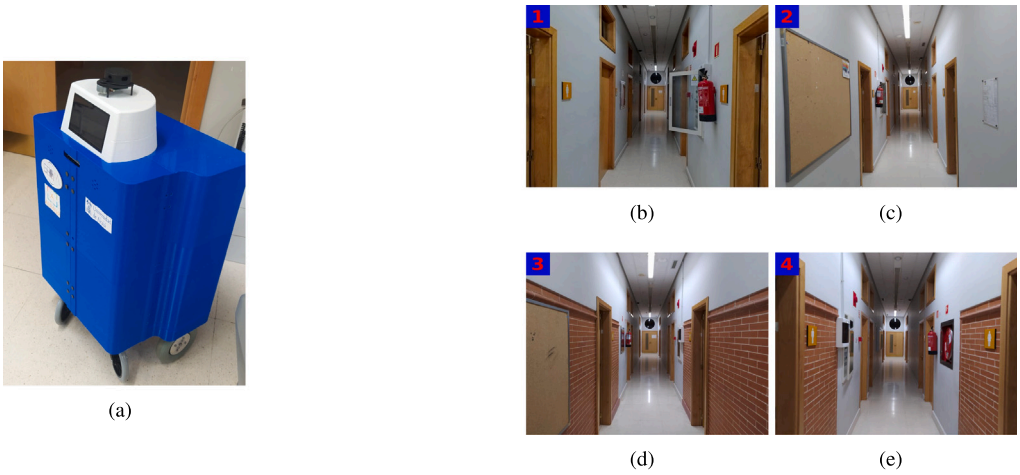


Fig. 1. Examples of highly symmetrical views in a building with numerous corridors. (a) Our own LOLA robotic platform. (b, c, d, e) Views at the entrances of four corridors captured with our own agent.

and not easily applicable in symmetric environments with repetitive perceptions (Zhang et al., 2019). Thus, we highlight the difficulties of global localization of AMCL in office corridors using solely data from distance sensors without considering visual information.

One of the greatest challenges for a mobile platform is to locate itself and navigate in complex indoor scenarios that present repetitive structures. Indoor environments generally consist of many rooms connected by corridors. Building geometry in corridor spaces presents, in general, a repeated pattern that makes it difficult to obtain specific information for global localization. This is often the case for public buildings such as hospitals, residences or care centers, where assistance platforms operate. The use of scattered beacons in these scenarios becomes unaffordable since it requires having a great number of landmarks and it is not always possible in large-scale buildings.

Fig. 1 provides an illustration of the structural symmetry of the scenarios under consideration, showcasing comparable views taken by our robot at the entrances to several hallways within the same structure. The existence and/or distribution of specific objects can help us distinguish between the views taken from various positions, which in this case exhibit a significant degree of similarity. The placement and presence of distinguishing elements, like a light box or a fire extinguisher, can serve to clear up confusion and distinguish between two identical hallways. For this reason, we emphasize the use of particular objects as natural identifiers and present a multimodal data-based localization approach that combines the perception of structural and semantic information. Stated differently, our approach is motivated by the idea of emulating cognitive perception, i.e., attempting to determine the agent's location by tracking the locations of objects and motions along the trajectory.

Motivated by this challenge, we propose an analytic localization method that utilizes spatial and temporal consistency using a reference map and time-series observations from the agent. More specifically, it is based on two intuitive ideas: (1) Spatial consistency can be analyzed to determine the similarities between sensor observation blocks and sections of a reference map, and (2) Temporal consistency is integrated over time using robot time-series observations. The robot is equipped with an RGB-D camera, a 2D LiDAR and the motor encoders. Fusion of these sources of information allows for reducing the level of ambiguity in the location. Moreover, trajectories not consistent with movement patterns are discarded.

In our approach, the system relies on a topological map as a graph-based representation of the environment defined by a set of nodes, where link edges denote environment connectivity or reachability. We formulate the GL problem as the estimation of the node closest to the robot's position and direction of movement within the last edge (edge orientation). Fig. 2 depicts the flowchart of our model, which consists of three main components: 2D Object Position Estimation, Node Detection and Topological Localization. The 2D Object Position Estimation module is able to extract the topological distribution of objects whereas the Node Detection stage identifies the type of zone in which the robot is moving. In more detail, this second module allows us to subdivide the scenario into sections corresponding to topological edges. The merging of both outputs (Object Position Estimation and Node Detection) generates the topological structure of the explored route. Finally, this output feeds the Topological Localization module, which estimates the location by comparing both spatial and temporal consistency between the topological structure generated in the route and a reference map. Further details are provided in Section 3.

The contributions of this work can be summarized as follows:

1. A new two-level structure is proposed to describe complex symmetrical environments. It includes a coarse topological approach with some metric references and comprises multisensory integration: visual information from the camera, depth information from a 2D LiDAR and motion information from encoders.
2. A novel analytic model for global localization has been proposed and implemented based on evaluating spatial and temporal consistency. It models the consistency between the readings from sensors and the ground truth (GT) combinations of routes. This contribution allows us to estimate the localization without the need of a manual initialization.
3. A visual object detector has been trained for the detection of characteristic objects in indoor environments, such as doors, windows and fire extinguishers. These objects are used as natural beacons, so no intervention is needed in the scenario.
4. A supervised learning model has been developed for the classification of nodes, which makes it possible to identify the type of zone in which the agent is located. Our system is able to distinguish whether the agent is in a corridor, at the end of a corridor, or at a junction of corridors.

Experimental results test the capabilities of the system with two real-world environments. The first presents an approximate square area of 100 × 100 meters and is organized in a ring around 16 corridors with a rate of free zones in occupancy map equal to 6.98%. It supposes a great challenge for localization due to its high level of symmetry. The second environment corresponds to a medium-sized building with a lower level of symmetry than the first. The results in both scenarios support the generalization capacity of our algorithm.

The rest of the manuscript is structured as follows. We present a brief literature review on existing methods for indoor localization in

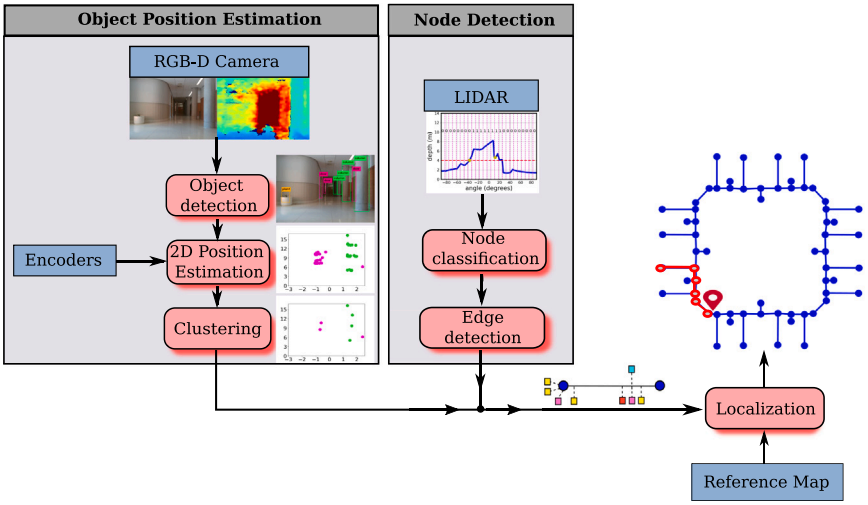


Fig. 2. Model overview of multisensory integration based topological localization. It consists of three components: 2D Object Position Estimation, Node Detection and Topological Localization.

Section 2. Section 3 presents the core of this work. It describes the architecture of our algorithm and the analytic model to evaluate spatial and temporal consistency. Section 4 describes the experimental setup details and experiments. Finally, the last section concludes this paper and suggests some future research lines.

2. Related work

The objective of indoor localization is to accurately estimate the location of an agent concerning a reference map. As explained in the introduction, the loss of GPS signal indoors due to the NLOS (Non-line-of-sight) effect involves a challenge. As a consequence, different technologies have been proposed in the scientific community in past years to find a solution to this problem: Bluetooth (Ayyalasomayajula et al., 2018), WiFi (Hernández et al., 2021; Luo et al., 2017), UWB (Ultra Wide Band) (Ridolfi et al., 2016), RFID (Diallo et al., 2019), magnetic field (Chen et al., 2021) or ultrasonic systems (Mannay et al., 2020), among others. Wireless localization based on WiFi is the most common due to its high availability and high accuracy. However, in general, signal-based methods have two primary limitations: (1) They need to have the corresponding infrastructure, such as routers, beacons, RFID tags, or ultrasound receivers; and (2) They can only output the location but not the view angle.

As a complementary alternative in localization tasks, the robotics community has also considered visual-based localization methods. Image-based algorithms do not need extra infrastructure to support indoor localization. Moreover, image-based localization methods can estimate location linking information about what is being seen in a scene to what exists in a digital information model. The most common techniques are inspired by the following approaches: image retrieval (Wolf et al., 2005), structure-based localization methods (Sattler et al., 2017; Taira et al., 2018) and learning-based localization methods (Walch et al., 2016). Image retrieval methods build a spatial repository during mapping and determine the image that is most similar to the queried one for localization. Since the queried image is not likely to have identical poses as the reference images, the system can only provide a rough estimation. Structure-based localization is based on hand-crafted or deep learning-based features (Arandjelovic et al., 2015) to estimate 2D-to −3D matches between features in a given image and points in 3D models. The camera pose is estimated from the correspondence by using epipolar geometry. Major limitations are related to camera intrinsic differences, which can provoke failures even when a small subset of mismatched features exist, and it requires a high computational burden. Finally, deep learning-based localization methods have emerged and benefited from deep-learning (DL) approaches. These algorithms either predict matches for pose estimation or directly regress the camera pose such as PoseNet (Kendall et al., 2015), PoseNet2 (Kendall & Cipolla, 2017) and VlocNet (Valada et al., 2018). However, these methods are still susceptible to different lighting conditions, are not scalable for large environments and generate ambiguity because two different locations can provide similar characteristics.

From the point of view of robotics, while classical approaches (Thrun et al., 2005) decoupled the tasks of SLAM (simultaneous localization and mapping), many models in recent years have proposed solutions that directly map sensor data to robot actions (Savinov et al., 2018; Ye et al., 2018). Most methods integrate reinforcement learning (RL) techniques with DL models for visual perception, e.g. Chang et al. (2020). Interestingly, the resulting action policy learns navigational priors that depend on the contextual scene, e.g. kitchen or bedroom. The main limitations of most previous models are as follows: (1) they are not tested on real scenarios, i.e., the experimental evaluation is mainly done on different virtual environments where the deep RL approaches are trained on Szot et al. (2021); and (2) in consequence they exhibit a low generalization capacity, making it difficult to use a trained model to navigate in a different scenario to the one used for training. For these reasons, these approaches exhibit limited capabilities to generalize to new environments or to plan long trajectories. In consequence, these works are hardly applicable to the environments of interest in our work.

Departing from the works above, we follow the traditional approach of decoupling localization from planning and execution. While end-to-end tend to struggle in large-scale environments, topological solutions have a rich history in classical literature (Kuipers & Byun, 1991; Tomatis et al., 2001). These approaches build a graph using trained models that can predict whether two images are close (Savinov et al., 2018), or regress the number of steps required to move from one image to another (Shah et al., 2021). However, few of these works propose graph update strategies that improve lifelong navigation performance over time. Researchers have also incorporated insights from classical literature into neural structures. Thus, a modular model is proposed in Chaplot et al. (2020), where each node is associated with a panoramic image. Graph localization requires comparing each node image with the given image based on a ResNet18 encoder (He et al., 2016) and a Connection model. Some similarities can be found in Wiyatno et al. (2021), where a convolutional neural network (CNN) takes two images and predicts the probability of reachability from one image to the other and their relative transformation. Even when our work builds upon the existing literature on topological maps, the

resemblance is only at high-level graph structure. In terms of our representation, the structure is not only relying on visual information but also on structural information. In contrast to Chen et al. (2019), where they work with one of the three generic node types: *room*, *hallway* and *open space*, we define nodes as points with path changes. Hence, nodes encode the types of possible trajectories in our proposal.

Most of these previous works exhibit important limitations since they are not tested on real scenarios of large scale, but in simulated environments or real-world environments of medium or small size. Therefore, our work focuses on models of localization in complex and highly symmetrical environments using multisensorial integration. To overcome the limitations of image retrieval methods, which are based on the use of only a single image; it seems reasonable to use time-series observations from perceptual localization. There is a connection between our work and the general purpose of Niwa et al. (2022). In this manuscript, the architecture is based on a recurrent-type graph neural network and it requires a sim2real transfer. However, our goal is to provide an accurate localization without the need for a long training process and the difficulties associated with transferring the learned skills to reality.

3. System description

In this section, we describe the proposed method for robot self-localization in complex symmetrical environments based on a topological map. The topological map encodes spatial information about the environment, which is represented as a directed graph denoted by $G = (N, E)$, where $N = \{n_i\}_{i=1:L}$ is the set of L nodes. Each node is defined as a location with a relevant change of trajectory that can be encoded using a LiDAR signature. Also, $E = \{(s_i, t_i)\}_{i=1:B}$ is a tuple of B edges, where each edge $e_i = (s_i, t_i)$ is connected from source node s_i to target node t_i, being both nodes considered as adjacent. Each edge stores a coarse distribution of objects (relative location of characteristic objects concerning the source node) from RGB-D camera observations and the relative pose between two adjacent nodes by adding encoder information. Assuming that in the existing topological map there are B edges $\{e_1, e_2, \ldots, e_B\}$, the localization problem is raised as the ability of the system to find the edge and the heading direction based on the sensory observations. Since there are two directions in the B edges, there are $2B$ possible locations.

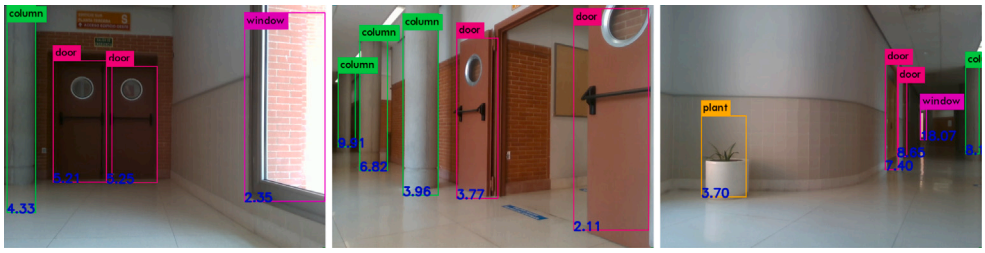
As mentioned in the introduction, baseline localization methods have difficulty accurately localizing the self-position when multiple similar sensorial measurements are contained in the topological map. To solve this problem, we propose in Section 3.3 a novel self-localization system based on the analysis of spatial and temporal consistency of sensor observations concerning a reference map. Previously, Sections 3.1 and 3.2 describe, respectively, our multisensory approach defining the environment. Because of the input to various modules of the system, as illustrated in Fig. 2, the operating frequency is not the same for all modules depending on requirements and computational load. More specifically, the Node Detection stage, which is based on LiDAR scans for delimiting the end of edges, requires a very low computational burden and is capable of working with frequencies similar to the scan rate (5–15 Hz). On the other hand, the Object Position Estimation module is triggered after the initialization of an edge and stops when the next node is detected and, finally, the localization step is only triggered when an edge is completed. That is why, in our approach, localization consists of determining the node closest to the robot's position and edge orientation. Here, sensory fusion plays a crucial role in the generation of the topological architecture: on the one hand, the LiDAR is used for the geometric characterization of the environment and the detection of obstacles during the movement of the agent, and on the other, the RGB-D camera and encoders are used to estimate the position of objects. It is crucial to emphasize that synchronization of sensor information is not a critical aspect because the Object Positioning module is tolerant to small odometry drifts that will be corrected through multiple detections of the same object. The platform speed is low enough (approximately 0.2 m/s) so that a small desynchronization between the camera and the encoders result in centimeter-level positioning errors that do not affect the performance. In this idea, our proposal does not require an exact topological description, but a coarse description is sufficient for localization.

3.1. 2D position estimation of objects

This module uses as inputs visual and depth images of an RGB-D camera as well as information from encoders. From these inputs our solution generates a map with a coarse 2D distribution of significant objects. Taking into account the most common objects in structured indoor environments, a YOLOv3 model (Redmon & Farhadi, 2018) was trained for visual detection of these objects. In particular, the following nine classes have been considered as natural beacons: window, door, elevator, fire extinguisher, plant, bench, firehose, light-box and column. Fig. 3 depicts as examples the output detections for three visual captures, where each detection is marked with its bounding box and includes the corresponding label and depth estimation in meters.

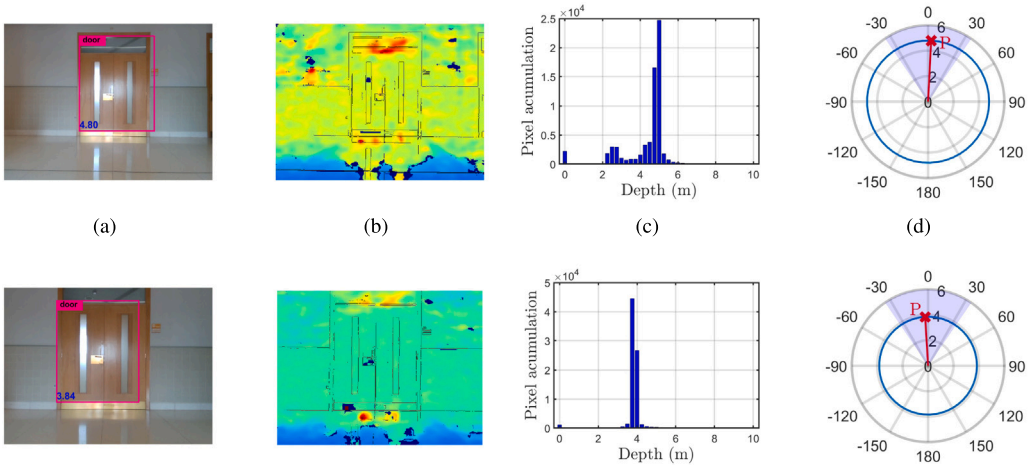
The steps taken to generate the topological distribution of objects are described below:

- Object detection. This step is based on YOLO as a real-time one-stage framework. The output of the visual detector provides the following information for each detected object: coordinates of the bounding box, the category of the detected object and the confidence score. Since this last parameter reflects how likely the box contains an object (objectness) and how accurate is the boundary box, we set empirically a threshold value of 0.8 to filter poor detections. To generate a custom model we ran a fine-tuning process using our own training dataset with a total of 3,193 labeled images of 640 × 480 pixels captured in indoor scenarios. Based on this training dataset 8,748 object instances were annotated and distributed between the mentioned nine classes, where one or more annotated objects correspond to each image.
- Object depth. The depth is estimated from the information provided by the RGB-D camera. For this purpose, the median of the depth of the pixels in the bounding box of the detected object is computed. Additionally, to maintain coherence with the RGB-D operating distance range and to avoid noisy measurements of pixels corresponding to the background of the scene or light reflections on windows, the system omits those pixels with a distance of more than 10 m and less than 1 m. As example Figs. 4(a) and 4(e) represent the output of the object detector for each of the captures, where the door has been detected during the robot's movement (each detection is marked with its corresponding category label, bounding box and depth estimation). Figs. 4(b) and 4(f) represent as examples the corresponding depth images as colormaps overlapping them with the detected edges of the scene for a better understanding, whereas Figs. 4(c) and 4(g) depict the corresponding depth histogram for both doors. As can be seen from the comparison of the two histograms, the measurement accuracy is higher, in general, as the robot gets closer to the object.
- 2D Object position. The estimation of object position is performed based on the depth obtained in the previous step and the estimation of the angle of the object centroid with respect to the optical axis. Precisely, to estimate the angle of the object we rely on the geometry of the camera using the pinhole model. Thus, we can consider that the angle of the object concerning the optical axis is approximately equal to the ratio between the angular aperture of the camera and the distance in pixels of the projected object in the image plane from the center. The geometric model to estimate object localization is depicted in Figs. 4(d) and 4(h) in polar coordinates for corresponding examples of Figs. 4(a) and



(a) (b) (c)

Fig. 3. Examples of results at the output of our YOLOv3 object detector including depth estimation.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 4. Examples of object position estimation for two captures taken on the same edge. (a,e) Visual images with object detection and depth estimation. (b,f) Depth images. (c,g) Depth histograms for detected objects. (d,h) Geometric model for object position estimation corresponding to (a,e), respectively.

4(e). Here, the position estimation of the object 'door' (denoted as point P) is determined by the intersection of the blue circle of radius equal to the depth of the object and the direction given by the angle of the object concerning the optical axis (red line). The shaded cone specifies the aperture of the camera, which in our case is equal to 69°. As the robot approaches an object, it is detected several times and the multiple detections serve to improve the accuracy of the location estimation. In Fig. 5 d_1 and d_2 represent the distances from two objects to the reference node whereas d'_1 and d'_2 are the distances to the image plane of the robot camera. From these distances and their corresponding angles α_1 and α_2, transverse and longitudinal components are calculated.

In our algorithm, the origin node **O** on each edge is considered as the longitudinal distance reference. In the considered environments based on a multi-aisle layout, we can roughly consider that the optical axis of the robot's camera moves aligned with the center of the aisle between two consecutive nodes.

- Temporal integration of detections. This module combines into a single detection several time-based detections of the same object along an edge's path. For this purpose, a separate agglomerative clustering algorithm is used for each object category with a threshold of maximum distance between detections of the same object. Owing to the nature of the objects, which can be large-area extensions like doors and windows or small objects like fire extinguishers, difference thresholds for distance have been established for each object category. The procedure is shown for one edge in Fig. 6. More specifically, the map generated at the end of the agglomerative clustering is depicted in Fig. 6(h), while the multiple estimates of localization for the identified objects are shown in Fig. 6(g). From Fig. 6(h) note that the transverse distance is significantly smaller than the longitudinal distance. For reasons of space, not all the captures along the edge have been represented, but rather each three consecutive images correspond to longitudinal distance increments in the robot's position of 1.5 m.

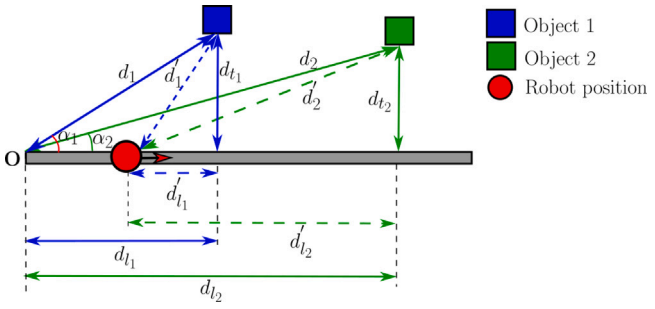


Fig. 5. Scheme for estimation of 2D object position estimation along the movement on an edge, which is represented by gray color. Note that the origin reference **O** is here placed locally at the origin node of each edge. Here, d_{t_1} and d_{t_2} denote the transverse distances, and d_{l_1} and d_{l_2} the longitudinal components. Variables denoted with hyperindex prime refer to the distances of the objects with respect to the image plane of the robot camera.

3.2. Node detection

The purpose of this stage is the detection of characteristic zones of the environment regarding to the geometrical information. In our approach nodes represent areas of the environment that force a relevant

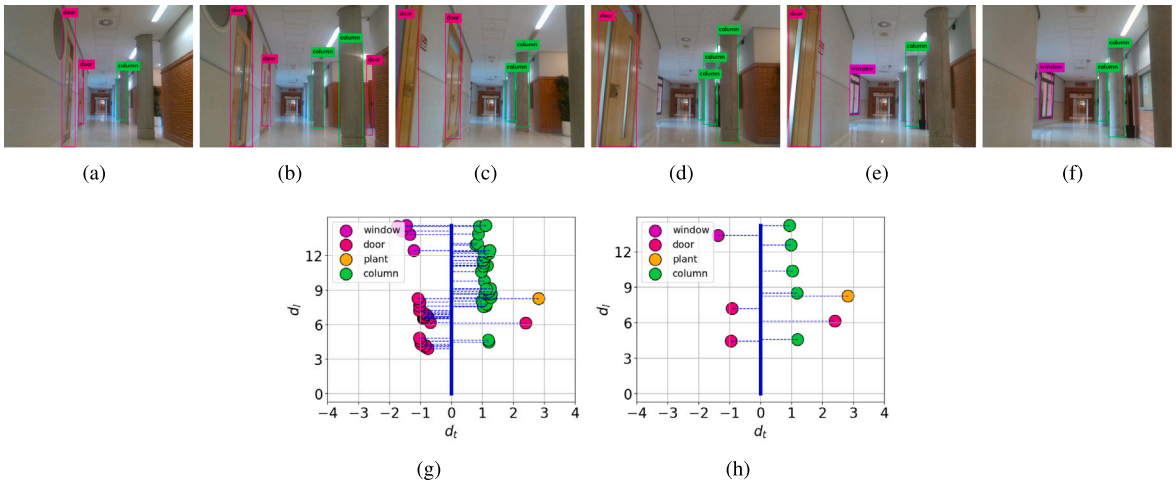


Fig. 6. Example of generation of 2D object position estimation. (a–f) Visual detections of objects. (g) Map of detections. (h) Map of object position estimation after clustering.

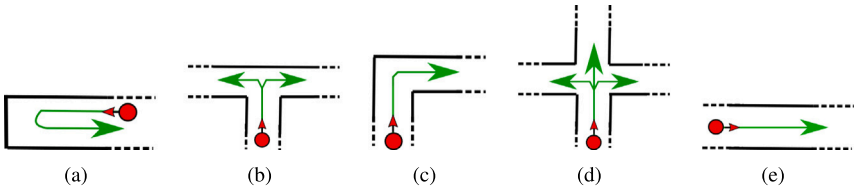


Fig. 7. Types of nodes considered in a building with dense distribution of corridors (robot position and orientation are represented in red). (a) End Node. (b) Node 'T'. (c) Node 'L'. (d) Cross node. (e) Non node.

change in the agent's trajectory. Since we consider that an edge is defined by two adjacent nodes, this sub-module allows us to delimit the beginning and end of the different edges of the explored environment. In fact, the 2D Object Position Estimation stage is only triggered when the agent enters the zone of influence of a new node and its execution stops when it detects the following node. Here, edges are considered as transitions between nodes. Fig. 7 represents the different categories of nodes considered in this work. Thus, for each type of node, we represent the location of the robot with a cross and the possible trajectories are depicted in green. Regarding this criterion, common corridor structures can be classified into the following categories:

- End node: There is no outlet at the front, neither from the left nor from the right of the corridor. The agent cannot move forward.
- Node 'T': There are two outlets for the agent since the corridor presents two lateral bifurcations.
- Node 'L': The path presents a marked change of direction. We have considered in this category changes of direction involving angles less than 40°, regardless of whether the turn is from the left or the right.
- Cross node: The corridor presents three or more outlets (two laterals: left and right, and another frontal one).
- Non node: It is considered as a special case of node and it allows to incorporate additional context into the exploration. It refers to the zones of transition between nodes while the agent is moving along the corridors.

The LiDAR signature feeds the input of our classifier whose output determines the type of node. Because of the efficiency of Support Vector Machines (SVMs) (Cortes & Vapnik, 1995), we used this technique for our classification module. Fig. 8 shows examples of LiDAR signatures for the five node types considered. The left images represent the point cloud based on a raw LiDAR scan, where the red dot and line indicate the position and orientation of the robot. In addition, right-hand figures depict in blue the signature extracted by our system after processing raw scans as a function of the angle. Here, green and red boxes show explorable (free space) and non-explorable directions, respectively, for navigation considering a distance threshold of 5 m. We have established the parameter N_b as the number of bins into which to divide the angular range of measurements. Note that we limit the angular range of laser scans from −90° to 90° because, depending on the mobile platform structure, the backside may intercept the beam.

An inherent challenge of SVM-based classifiers lies in selecting the appropriate kernel and its specific parameters. It requires a search for the optimum settings for the particular problem. Optimal values of parameters in our model were determined in our early work (Lafuente-Arroyo et al., 2022) as a trade-off between accuracy and model complexity. Specifically, optimal values were $N_b = 50$, $C = 8$ and $\gamma = 0.25$, where C is the regularization parameter and γ is the parameter of influence of the kernel RBF.

3.3. Topological localization

The purpose of this module is to provide an automatic localization based on a comparison of the perceptual observations captured by the agent on its movement and the topological map used as a reference. In essence, we aim to address the question: where are we in the graph? Specifically, this module is activated after completing each edge to search for the best matching route. The matching of each edge is computed by an evaluation function that encompasses its approximate length, its topological distribution of objects and the categories of the two nodes that define it. This function assigns a weight to each edge of the reference map as a measure of similarity and, consequently, the highest weight determines the edge that best fits the observations. Recursively, the weight of a trajectory can be calculated as the product of the weights of the edges that constitute it.

In this approach, we define a trajectory as a set $Y = \{y_i\}_{i=1:P}$ of P node indices and $P-1$ edges as connections of each pair of consecutive nodes. Additionally, $M = \{M_i, d_i\}_{i=1:P-1}$ represents the

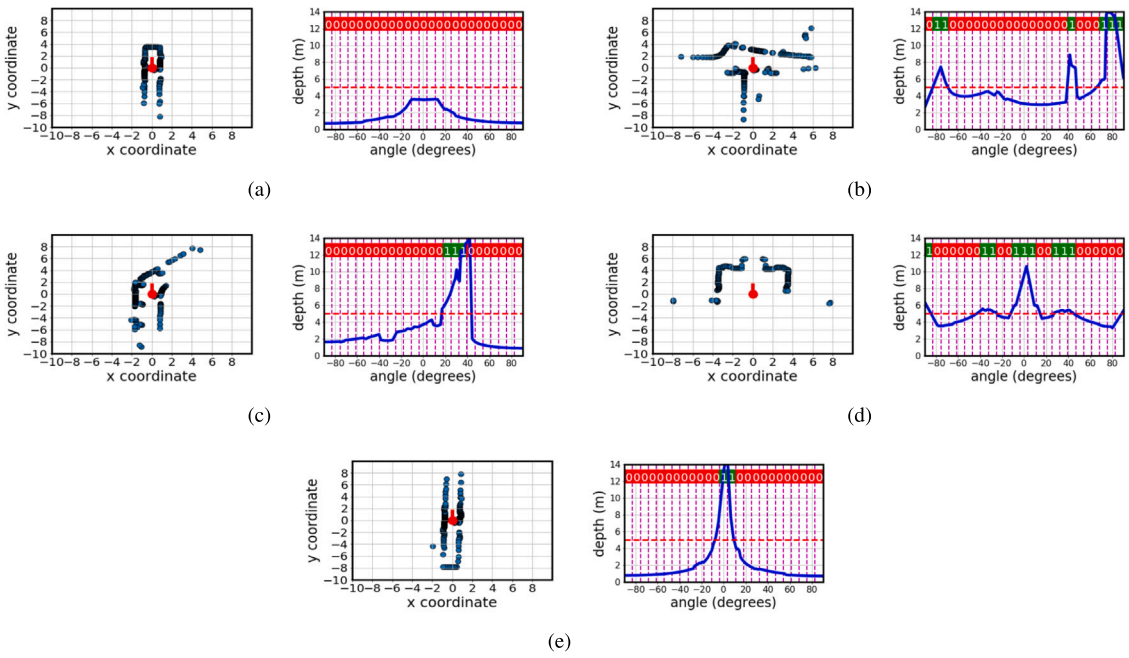


Fig. 8. Examples of depth signatures with 2D LiDAR scanning in different indoor contexts (Left image: LiDAR scan point cloud with robot position and orientation in red, and right image: signature extracted by our system represented in blue considering 25 bins). (a) End node. (b) 'T'-type node. (c) Node 'L'. (d) Cross node. (e) Non node.

observations/measurements of sensors through edges. Thus, for a specific edge $e_i = (y_i, y_{i+1})$, the parameter d_i denotes the edge length provided by encoders and M_i is a tuple that defines the detected objects and their positions concerning the origin node of the edge. Taking into account sensory information, we determine a weight associated to each possible trajectory of the same length (number of edges) in the reference map (GT) defined by the sequence of nodes $Y' = \{y'_i\}_{i=1:P}$ and sensory information $M' = \{M'_i, d'_i\}_{i=1:P-1}$ annotated in a previous process.

The analytic model we propose to estimate the weight $w_{Y'}$ of the hypothetical trajectory Y' can be broken down for each ith edge into two contributions: $w_{i,\text{nodes}}$ and $w_{i,\text{objects}}$, corresponding to the nodes and distribution of objects, respectively. The model is given by the following equation:

$$w_{Y'} = \frac{1}{2^{P-1}} \prod_{i=1}^{P-1} \left(w_{i,\text{nodes}} + w_{i,\text{objects}} \right), \tag{1}$$

where both the weights of $w_{i,\text{nodes}}$ and $w_{i,\text{objects}}$ as well as the weight of $w_{Y'}$ will lie within the interval [0,1] due to the normalization factor 2^{P-1}. Here $w_{i,\text{nodes}}$ quantifies the similarity between node classes and the edge length between detection and annotation, whereas $w_{i,\text{objects}}$ considers the similarity in distributions of objects.

In more detail, Eq. (1) can be rewritten as follows:

$$w_{Y'} = \frac{1}{2^{P-1}} \prod_{i=1}^{P-1} \left(f_i e^{-\gamma_1 \Delta d_{n_i}} + \prod_{j=1}^{O_i} g_j e^{-\left(\gamma_2 \Delta d_{l,o_j} + \gamma_3 \Delta d_{t,o_j}\right)} \right), \tag{2}$$

where for each ith edge the term $\Delta d_{n_i} = \left| d_i - d'_i \right|$ represents the difference of length in absolute value between odometry readings and annotation, and O_i is the number of objects. The terms $\Delta d_{l,o_j}$ and $\Delta d_{t,o_j}$ correspond to the difference in positioning: longitudinal and transverse, respectively, between detected objects and reference positions. Here, it is important to point out that we only consider the matching of objects of the same category. Thus, we have established a correspondence based on the search for the detected object closest to the GT objects, ensuring that the Euclidean distance between matched objects does not exceed a certain threshold. In addition, the constant γ_1 weights the distance differences between nodes, whereas γ_2 and γ_3 weight, respectively, the differences between longitudinal and transverse distances of objects. In the particular case of objects without correspondence the distances $\Delta d_{l,o_j}$ and $\Delta d_{t,o_j}$ are considered zero and only the parameter g_j, which will be described below, comes into play.

Finally, the meaning of the parameters f_i and g_j can be described as follows. The penalization factor f_i quantifies the discrepancy between the detected categories: $c(y_i)$ and $c(y_{i+1})$, of the two terminal nodes of the ith edge in the sequence Y and the categories: $c(y'_i)$ and $c(y'_{i+1})$, of the homologous ith edge in the reference sequence Y' as:

$$f_i = \begin{cases} 1 & \text{if } \left(c(y_i) = c(y'_i) \quad \text{and} \quad c(y_{i+1}) = c(y'_{i+1})\right) \\ K_1 & \text{if } \left(c(y_i) \neq c(y'_i) \quad \text{and} \quad c(y_{i+1}) = c(y'_{i+1})\right) \\ & \quad \text{or } \left(c(y_i) = c(y'_i) \quad \text{and} \quad c(y_{i+1}) \neq c(y'_{i+1})\right) \\ K_2 & \text{others.} \end{cases} \tag{3}$$

Thus, when the node type matches in the detection and GT, then $f_i = 1$. In addition, the constants K_1 and K_2 penalize the impact of having discrepancy in only one type of node or both of them, respectively. We can also see that the parameter g_j quantifies the effect of correspondence between the set of detected objects $O = \{o_j\}_{j=1:D}$ and the set of reference objects $O' = \{o'_j\}_{j=1:R}$ for the ith edge as:

$$g_j = \begin{cases} 1 & \text{if } o_j \in O_D \longrightarrow o'_j \in O' \\ K_3 & \text{other.} \end{cases} \tag{4}$$

Thus, when there is correspondence between a reference object in the map and a detected object, then $g_j = 1$. If, on the other hand, there is no correspondence, we set $g_j = K_3$. In our model, the constants K_1, K_2 and $K_3 \in \mathbb{R}^+$ with $K_1 < 1, K_2 < 1$ and $K_3 < 1$, being their optimal values experimentally determined.

In a scenario defined by a graph with a set of L nodes $N = \{n_i\}_{i=1:L}$ and with a route of P nodes and $(P-1)$ edges, the algorithm requires the computation of a multidimensional weight matrix or tensor W of shape L^P. Here, each entry represents the weight of each hypothetical route regarding measurements. We shall refer to the order of the matrix as the number of considered edges in the evaluation. However, we can reduce the computational burden by considering only the history of the last edges in each iteration. Thus, we define a first-order $P \times P$

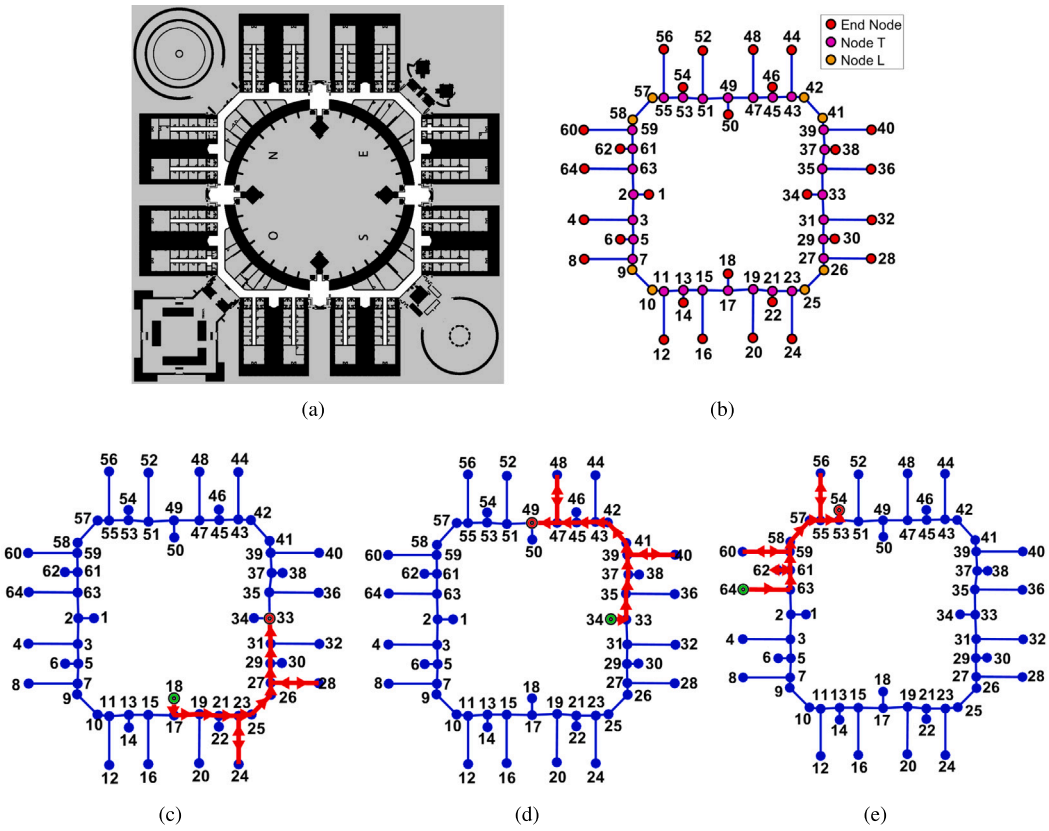


Fig. 9. Map of the first scenario used in our experiments. (a) Occupancy map of the building of the Polytechnic School of the University of Alcalá (white, gray and black pixels represent free space, non visitable space and occupied space, respectively). (b) Topological map with node indices. (c) Validation route. (d) Test route 1. (e) Test route 2. The direction of travel on the edge is represented by the arrow and the double arrow indicates the corridors that are traveled in both directions. The source and destination nodes of each route are shown with green and red circles, respectively.

matrix W_1 only if the last traversed edge is considered, wherein each entry represents the weight of a possible route defined by an edge as a transition between two nodes. In the case of a second order $P \times P \times P$ matrix W_{12}, we consider only the two last edges and each entry represents a possible route defined by two edges as a transition between three nodes, and so on. For instance, the entry $w_{i,j,k}$ of the second order matrix W_{12} indicates the weight of the route defined by the set of nodes: $\{y_i, y_j, y_k\}$. It is worth noting that to reduce the number of entries to be computed, we establish the following criterion: only entries of the matrix that define a route consistent with the adjacency matrix, which indicate whether pairs of nodes are adjacent in the reference map, and with the movements provided by encoders are computed and have a nonzero value.

4. Experiments and results

We evaluated the proposed localization method using two public real-world environments. Specifically, the first localization experiments were conducted on the building of the Polytechnic School of the University of Alcalá, which is distributed over four floors of approximately 10,000 m^2. Fig. 9(a) shows the detailed occupancy map of one of these floors and Fig. 9(b) depicts the topological map generated for the experiments. As we can observe, the structure includes a total of 64 nodes referenced by their corresponding indices and 128 bidirectional connecting edges considering that the topological distribution of objects is different for the two directions. Because the object-viewing structure is not symmetrical, each of the two possible directions of an edge is treated differently. Complementary experiments were conducted at the Faculty of Nursing and Physiotherapy of the University of Alcalá. It is a medium-sized building and the objective of these last experiments, which are described in Section 4.5, was to test the generalization capacity of the system in another kind of environment.

4.1. Experimental setup

Captures have been taken in both environments with our own low-cost assistive robotic platform (see Fig. 1(a)). LOLA is a low-cost mobile assistive robotic platform described in López-Sastre et al. (2021) and the entire mechanical and electrical design has been conceived by our research team. It is equipped with two motors and their corresponding encoders, which are all controlled with an open-source Arduino board. The internal structure is constructed of wood and metal. Additionally, the outer shell, imitating a person wearing a tuxedo, was made entirely by 3D printing. The platform is powered by two batteries, and it includes an electronic driver interface to allow easy interconnection of the different parts of the system and all the power management. The sensing part of LOLA is composed of an Intel RealSense D435 camera and an RPLIDAR A1 sensor. To integrate into the mobile robot all the high-level processing that cannot be embedded into the Arduino, the platform has two possibilities: a Jetson TX2 board from NVIDIA and an Intel NUC as Mini PC for the following functionalities: (a) visual perception, (b) online action detection, and (c) navigation.

We collected three datasets in the first environment (Polytechnic School): a reference dataset for generating the topological map, a validation dataset for optimization of the hyperparameters of the model and, finally, a test dataset to evaluate the proposed method. To obtain multisensory acquisition we teleoperated the robot to collect time series data. Captures were made with the mobile platform having fixed linear displacements of 0.5 m in the sections of edges.

The description of the three datasets is as follows:

1. Reference dataset: a topological map of the scenario was generated by exploring the entire floor of the building. Based on

this dataset, the defined two-level structure was generated automatically for the considered scenario using the two first steps of the system: 2D Position Estimation and Node Detection. This structure, which includes the distribution of objects and node categories, was used as a reference in validation and test processes. In total, the topological map includes 28 'T'-type nodes, 28 'End'- nodes and 8 'L'-type nodes. The map also includes 64 edges, which being considered different for each one of the two possible directions, generate a total of 128 bidirectional edges. The minimum length per edge in the reference dataset is 2.01 m, the maximum length is 19.65 m, and the mean length is 10.13 m.

2. Validation dataset: it includes the captures of a route of 14 edges defined by the following sequence of nodes (see Fig. 9(c)): $VS = \{18, 17, 19, 21, 23, 24, 23, 25, 26, 27, 28, 27, 29, 31, 33\}$, which presents 10 'T'-type nodes, 3 'End'-nodes and 2 'L'-type nodes.
3. Test Set: it includes the captures of two different routes with 14 edges each one (see Figs. 9(d) and 9(e)): $TS_1 = \{34, 33, 35, 37, 39, 40, 39, 41, 42, 43, 45, 47, 48, 47, 49\}$ and $TS_2 = \{64, 63, 61, 62, 61, 59, 60, 59, 58, 57, 55, 56, 55, 53, 54\}$. The distribution by categories of the nodes in test sequences is the following: 'T'-type nodes (10 and 8 for the first and second sequences, respectively), 'End' nodes (3 and 5 for the first and second sequences, respectively) and 'L'-type nodes (2 for both sequences).

To evaluate the performance of the model, we used as metric the normalized weight of the real edge w_{real_norm} defined as the ratio between the weight evaluated at the real traveled edge (w_{real}) and the maximum weight among all possible edges of the map excluding the real traveled edge ($w_{max-real}$). Consequently, three possible cases can arise:

$$w_{real\_norm} = \frac{w_{real}}{w_{max-real}} = \begin{cases} > 1 & \text{if } w_{real} > w_{max-real} \\ = 1 & \text{if } w_{real} = w_{max-real} \\ < 1 & \text{if } w_{real} < w_{max-real}. \end{cases} \quad (5)$$

Values above 1 indicate that the actual edge is discriminating, whereas a value equal to 1 indicates ambiguity in the sense that there exist similar edges to the real one. However, this latter case does not imply a problem because it is very likely that the concatenation of edges in a route is sufficient to achieve localization. Finally, values below 1 introduce location errors since there exist one or more edges with higher weight than the real traveled one. This is a critical case because of the greater difficulty in retrieving the true location. Since we considered the unit value as the threshold of the normalized weight defined by Eq. , we established the following criterion to quantify the location error in a sequence of edges: we analyzed the summation of deviations respect to the threshold value equal to 1, considering only those edges that introduce error locations (i.e. $w_{i,real\_norm} < 1$). Thus, we compute the accumulative error location (AEL) factor given by:

$$AEL = \sum_i (1 - w_{i,real\_norm})_{|w_{i,real\_norm}<1} \quad (6)$$

4.2. Hyperparametric optimization

Due to the computational complexity of optimizing our multi-parametric localization model, which owns six degrees of freedom, a hyperparametric analysis has been performed using the validation set. For this purpose, each one of the parameters was varied while the rest were kept fixed. The range of initial values was chosen after having carried out previous tests beforehand, thus ensuring that the optimum value of the parameter will be in that range. Specifically, the range of variation of γ_1, γ_2 and γ_3 was [0.00–0.36] with steps of 0.0125, whereas the range of variation of K_1, K_2 and K_3 was [0.00–0.95] with steps of 0.05. In this analysis of hyperparameters, each edge of the validation set is evaluated independently and not as a route of edges. We determine the optimal value of each of the six above-mentioned parameters (γ_1, γ_2, γ_3, K_1, K_2 and K_3) using the minimization of the AEL factor. Fig. 10 shows the graphs of the AEL factor as a function of the range of values analyzed for each hyperparameter. To evaluate the impact of the parameters K_1 and K_2, which by definition penalize errors in the classification of nodes, we manipulated node categories of three of 14 total edges of the validation set.

Table 1 shows the optimal values and their corresponding values of the discrimination factor. Analyzing the results, it should be noted that the optimal value of $\gamma_1^* = 0.00$ warns of the irrelevance of analyzing the difference in lengths in scenarios like ours with corridors of similar lengths for similar distributions of objects. Additionally, we can verify that the discrepancy factors between node categories present high values ($K_1^* = 0.70$ and $K_2^* = 0.80$) close to 1. Note that the value 1 was fixed in Eq. for cases in which the classification of nodes is correct. These results suggest that metric information of the edge as well as categorization of nodes modeled by the first term of Eq. (1) is much less discriminatory than object distribution corresponding to the second term in the same equation modeled by the parameters γ_2 and γ_3 in exponentials. In conclusion, experiments have demonstrated that the distribution of characteristic objects is the most relevant information for localization in corridor environments to the detriment of node types and length of corridors.

4.3. Matrix order analysis: Performance and processing time

To evaluate the influence of the order of the weight matrix W in the performance we conducted experiments with the first sequence of the test set. Unlike in the previous sub-section, the evaluation is made by integrating the time series of the route and not independently for each edge. Figs. 11(a)–11(c) illustrate the results corresponding to first (W_1), second (W_{12}) and third order (W_{123}) matrices, where w_{real} represents the computed weight of the real traveled route, w_{max} the maximum weight of the matrix and $w_{max-real}$ the maximum weight of the matrix excluding the real traveled route for each iteration. To correct possible discontinuities in consecutive iterations of the process we propose to consider a variation of the algorithm that takes into account spatial and temporal consistency in estimation of the route. Hence, all entries of the matrix are multiplied in each iteration by a penalization factor $\alpha < 1$ except those entries that have continuity with the location estimated in the previous iteration. Specifically, Figs. 11(d)–11(f) show the output values (W_1', W_{12}' and W_{123}') after applying a penalty factor $\alpha = 0.9$. From the inspection of the results, it can be seen that initially, the algorithm is able to self-locate correctly ($w_{i,max} = w_{i,real}$) in some specific intervals: from iteration 6 to 9 for W_{12} and from iteration 7 to 10 for W_{123} and from 13 to the end in both cases. However, considering the consistency of the trajectory in estimations denoted by hyperindex prime, we observe greater stability and the algorithm estimates the real location correctly from the first iteration in which it is able to self-locate (from iterations 6 and 7, respectively, for W_{12}' and W_{123}', to the end).

From inspection of Fig. 11, it is worth noting that as we increase the order of the matrix, the performance improves and discriminating power also increases. This is also demonstrated by the values of the AEL factor for the different orders: 0.90, 0.37 and 0.32 for W_1', W_{12}' and W_{123}', respectively.

The dual trade-off between performance and computational burden led us to analyze, in addition, computation time as a function of the order of the weight matrix. The localization function is executed once for each edge detected and includes two tasks: matrix computation and determination of the maximum weight. Fig. 12(a) represents on a logarithmic scale the box plot as a function of the matrix order for test sequence 1 considering that the experiments have been conducted on a PC with 3.40 GHz Intel® Core™ i7-6700 CPU, 16 GB memory, NVIDIA GeForce GTX TITAN X GPU and 64-bit Ubuntu 20.04 operating system. Here it can be seen that the computation time exhibits a clear exponential variation as the matrix order increases, with average times of 45.45 ms, 30.90 ms, 252.50 ms and 133.76 s for W_1, W_{12}, W_{123} and

Table 1
Optimal values of the hyperparameters of the proposed localization model and their corresponding AEL factor.

$\gamma_1^* = 0.00$	$\gamma_2^* = 0.025$	$\gamma_3^* = 0.05$	$K_1^* = 0.70$	$K_2^* = 0.80$	$K_3^* = 0.80$
$AEL = 0.38$	$AEL = 0.13$	$AEL = 0.13$	$AEL = 1.55$	$AEL = 0.15$	$AEL = 0.15$

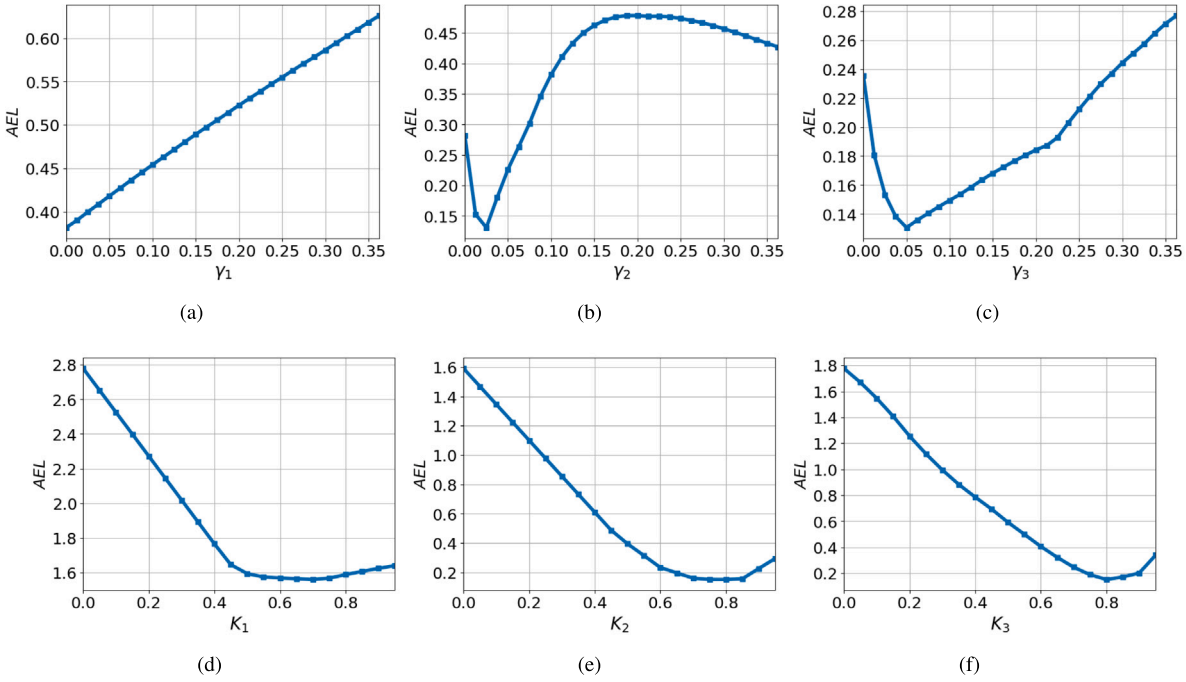


Fig. 10. Hyperparametric optimization of the model. (a) Parameter γ_1. (b) Parameter γ_2. (c). Parameter γ_3. (d) Parameter K_1. (e) Parameter K_2. (f) Parameter K_3.

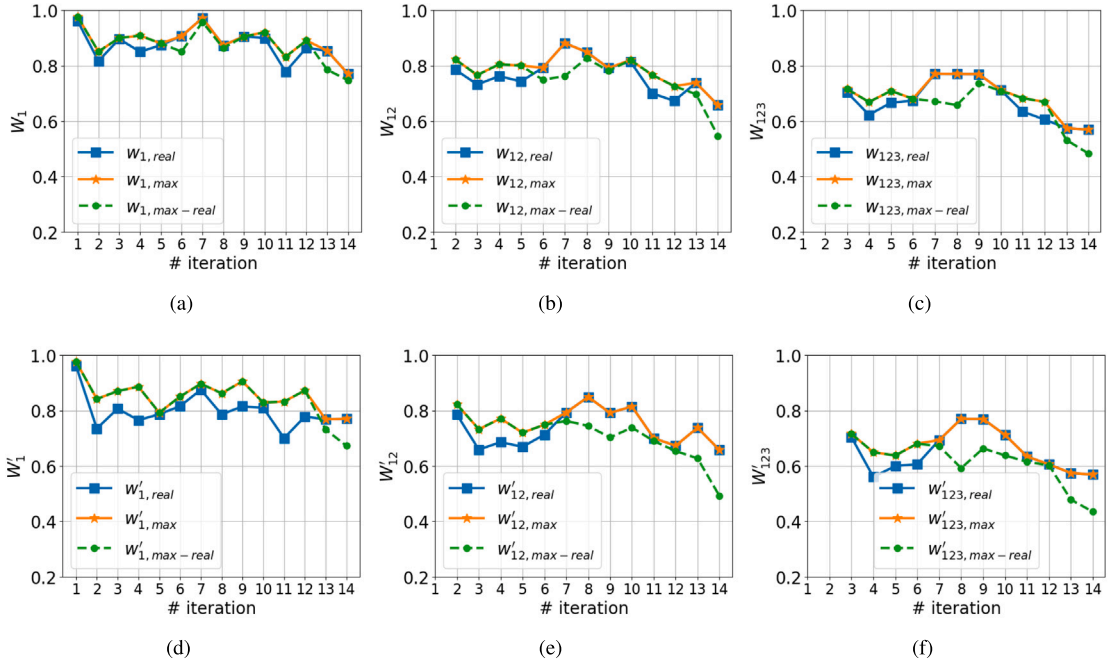


Fig. 11. Localization performance of the proposed algorithm on test sequence 1. (a, b and c). Weights of matrices W_1, W_{12} and W_{123}, respectively. (d, e and f). Weights of matrices W_1', W_{12}' and W_{123}' that include the trajectory continuity modification.

W_{1234}, respectively. As a trade-off between accuracy and computational time, the results lead us to consider the matrix of order 3 as the best option of those analyzed, showing the fourth order a prohibitive value. It is important to note that the computation time for the first-order matrix is slightly longer than for the second-order matrix. This is because, as is explained below, the system only considers the routes consistent with rotation angles provided by the encoders and in the case of working with a first-order matrix it does not make sense to apply turn compatibility. The breakdown of processing time of the two tasks involved in the localization process is shown in Fig. 12(b). As is to be

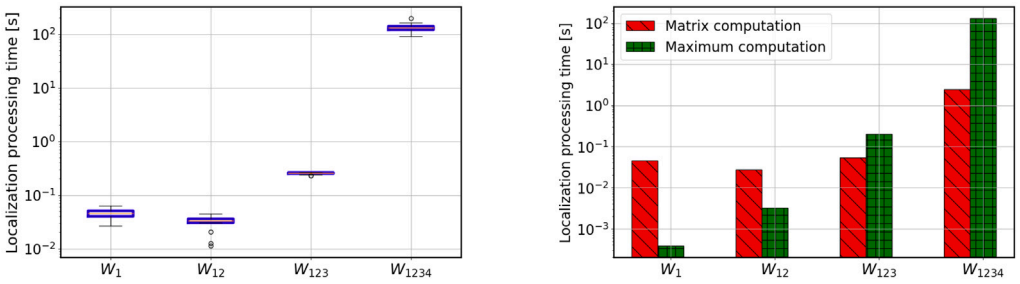


Fig. 12. Localization processing time of test sequence 1. (a) Box plot for different matrix orders. (b) Average time of the two sub-tasks: matrix computation and determination of maximum weight.

expected, the computational burden for the computation of matrices increases with the order of the matrix, but the biggest bottleneck as the order increases is in the determination of the maximum weight.

Here, we need to differentiate between the time required to self-locate at the end of each edge and the time spent to generate the topological structure of each edge. Considering the 14 edges of test sequence 1, we have obtained an average processing time per capture of 99.91 ms, which involves object detection, tracking and node classification and brings us closer to a real-time solution.

4.4. Localization: Analysis of results

This experiment is intended to evaluate the system under different conditions after adjusting the optimal hyperparameter and setting the third-order matrix with trajectory consistency. Table 2 summarizes localization results using the test sequences 1 and 2 by breaking them down into sections of three edges (third-order sections). We include the real sections in each iteration, the estimated section and the total number of candidate sections. By inspection, we can observe that the system is capable of estimating the true location from iteration 7 in test sequence 1 and from iteration 3 in test sequence 2. The better estimation of sequence 2 is because the route includes a larger number of corridors with discriminative objects (fire extinguishers, light boxes, hoses, etc.) than sequence 1, which includes a larger number of central ring sections in which the objects (doors, columns and windows) are not able to significantly reduce ambiguity. A demo video is provided on the site,[1] where a small pause has been added when the agent reaches the end of each edge to show the clustering result and the new location estimate.

It is important to point out that from the total number of entries ($64^4 = 16,777,216$) of the third order matrix, we impose two restrictions: (1) we only take into account those candidate routes of three edges that are consistent with the adjacency matrix (728 possible entries according to our scenery), and (2) the system filters out the possible routes of three edges consistent with the rotation angles at nodes provided by the encoders. As it can be observed in Table 2, both restrictions suppose a great reduction in the number of possible routes to discriminate. In more detail, it should be noted that most common trajectories in structured indoor environments involve routes under the following characteristics of turning: (1) movement without turn when the robot passes through aligned edges (0° turn), (2) movement out of or into a corridor (±90° turn), and (3) movement back from the end of a corridor (±180° turn). Therefore, certain movement patterns with a lower probability of occurrence are highly discriminatory. In the case of our test scenario, most discriminating movements correspond to passing through 'L-nodes' with turning angles of approximately ±45°. Thus, we can see in Table 2 that the number of possible routes is reduced to 4 in these cases. As an example, Fig. 13 shows with red color the four candidate sections by turning compatibility for the two following third order trajectories: 40→39→41→42 from test sequence 1 and 60→59→58→57 from test sequence 2. Here, we represent with green and red circles the first and the fourth nodes of the section, respectively.

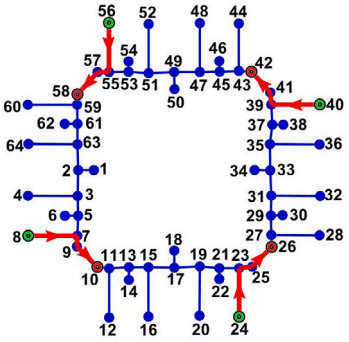


Fig. 13. Compatible paths (red color trajectories) with the third order sequence sections of test 1 (40→39→41→42) and test 2 (60→59→58→57). Source and destination nodes of the routes are shown with green and red circles respectively.

Finally, to analyze the contributions of nodes and objects to the weight of Eq. (2), we consider, as an example, the contributions when the platform runs along the edge 58→57. In this case, the system is able to perform the localization correctly being the two most weighted edges: 58→57 and 26→25 with values 0.867 and 0.797, in first and second place, respectively. Fig. 14 illustrates the object distribution map generated in the test process after traversing the edge 58→57 and the maps used as reference of the two most weighted edges, which were generated previously from the reference dataset. In addition, Table 3 summarizes the contributions of weights when comparing outputs of node and object detection with reference information. Here, we include the categories of both detected terminal nodes: $c(y_1)$ and $c(y_2)$, and those of the nodes annotated on the reference edge: $c(y'_i)$ and $c(y'_{i+1})$, as well as the difference in length Δd_n between each annotated edge and odometry. On the other hand, Table 3 includes else for each category of objects with correspondences the discrepancy vectors of longitudinal ($\boldsymbol{\Delta}\mathbf{d}_{l,o}$) and transverse ($\boldsymbol{\Delta}\mathbf{d}_{t,o}$) distances. As we can observe, the discrepancy between the detection process and references of edges 58→57 and 26→25 includes one and two unmatched objects, respectively. In this example, the discrimination between the two edges is given by the number of unmatched objects, which serves to attenuate the weight of the edge 26→25 concerning the edge 58→57.

The baseline methods used for comparison are based on two classic image retrieval approaches for image localization. Content-Based Image Retrieval (CBIR) allows to establish an image feature vector description from low-level visual features. Hence, CBIR algorithms find the image of the dataset with the closest similarity to the query image. Specifically, the two methods chosen as baseline for the evaluation are the following:

[1] https://youtu.be/sEp-kSXtRZY

Table 2
Localization results for test sequences 1 and 2, including the number of iteration, the third order real traveled sections, the estimations based on the use of the third order matrix and the total number of candidate routes. Incorrect estimations are shown in red and correct estimations in bold.

Test sequence 1 (TS_1)				Test sequence 2 (TS_2)		
it	Real route	Estimation	# routes	Real route	Estimation	# routes
1	34→33	–	–	64→63	–	–
2	34→ 33 →35	–	–	64→ 63 →61	–	–
3	34→ 33 → 35 →37	50→ 49 → 51 →53	24	64→ 63 → 61 →62	**64 → 63 → 61 → 62**	16
4	33→ 35 → 37 →39	49→ 51 → 53 →55	20	63→ 61 → 62 →61	**63 → 61 → 62 → 61**	16
5	35→ 37 → 39 →40	3→ 5 → 7 →8	20	61→ 62 → 61 →59	**61 → 62 → 61 → 59**	28
6	37→ 39 → 40 →39	5→ 7 → 8 →7	24	62→ 61 → 59 →60	**62 → 61 → 59 → 60**	16
7	39→ 40 → 39 →41	**39 → 40 → 39 → 41**	28	61→ 59 → 60 →59	**61 → 59 → 60 → 59**	16
8	40→ 39 → 41 →42	**40 → 39 → 41 → 42**	4	59→ 60 → 59 →58	**59 → 60 → 59 → 58**	28
9	39→ 41 → 42 →43	**39 → 41 → 42 → 43**	4	60→ 59 → 58 →57	**60 → 59 → 58 → 57**	4
10	41→ 42 → 43 →45	**41 → 42 → 43 → 45**	4	59→ 58 → 57 →55	**59 → 58 → 57 → 55**	4
11	42→ 43 → 45 →47	**42 → 43 → 45 → 47**	4	58→ 57 → 55 →56	**58 → 57 → 55 → 56**	4
12	43→ 45 → 47 →48	**43 → 45 → 47 → 48**	20	57→ 55 → 56 →55	**57 → 55 → 56 → 55**	4
13	45→ 47 → 48 →47	**45 → 47 → 48 → 47**	24	55→ 56 → 55 →53	**55 → 56 → 55 → 53**	28
14	47→ 48 → 47 →49	**47 → 48 → 47 → 49**	28	56→ 55 → 53 →54	**56 → 55 → 53 → 54**	16

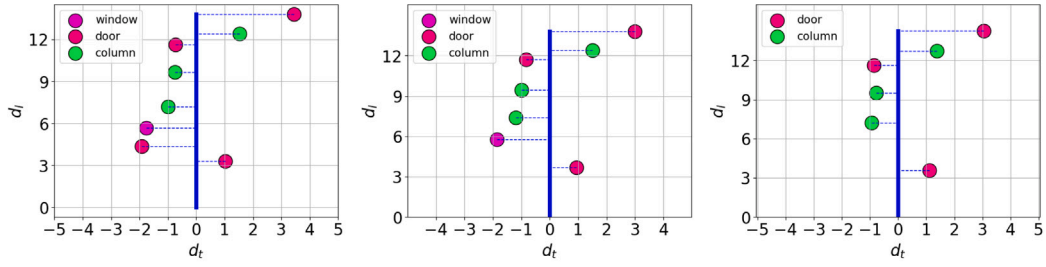


Fig. 14. Examples of topological maps for evaluation of weights. (a) Map generated in test process for the edge 58→57. (b) Map generated in annotation process as reference for the edge 58→57. (c) Map generated in annotation process as reference for the edge 26→25.

Table 3
Example of analysis of contributions in the weight evaluation for the edges represented in Fig. 14. Here, $c(y_1)$ and $c(y_2)$ represent the categories of both detected terminal nodes, $c(y'_i)$ and $c(y'_{i+1})$ the categories of reference edges and Δd_n the difference distance in meters between sensory information and references. It also includes for each category of objects with correspondences: discrepancy vectors of longitudinal ($\mathbf{\Delta d}_{l,o}$) and transverse distances ($\mathbf{\Delta d}_{t,o}$) in meters, and finally, the number of unmatched objects (NC).

Detection vs. edge 58→57		Detection vs. edge 26→25	
Node contrib.	Object contributions	Node contrib.	Object contributions
$c(y_1)$ = 'L'	'Door' ($w_{door} = 0.956$)	$c(y_1)$ = 'L'	'Door' ($w_{door} = 0.952$)
$c(y_2)$ = 'L'	$\mathbf{\Delta d}_{l,o} = \{0.090, 0.378, 0.020\}$	$c(y_2)$ = 'L'	$\mathbf{\Delta d}_{l,o} = \{0.001, 0.268, 0.490\}$
$c(y'_{58})$ = 'L'	$\mathbf{\Delta d}_{t,o} = \{0.101, 0.088, 0.460\}$	$c(y'_{26})$ = 'L'	$\mathbf{\Delta d}_{t,o} = \{0.121, 0.083, 0.400\}$
$c(y'_{57})$ = 'L'	'Window' ($w_{window} = 0.992$)	$c(y'_{25})$ = 'L'	'Column' ($w_{column} = 0.975$)
Δd_n=0.480	$\mathbf{\Delta d}_{l,o} = \{0.115\}$, $\mathbf{\Delta d}_{t,o} = \{0.108\}$	Δd_n=0.310	$\mathbf{\Delta d}_{l,o} = \{0.037, 0.148, 0.310\}$
	'Column' ($w_{column} = 0.967$)		$\mathbf{\Delta d}_{t,o} = \{0.061, 0.046, 0.153\}$
	$\mathbf{\Delta d}_{l,o} = \{0.010, 0.197, 0.188\}$)		$NC = 2$ ($w_{NC} = 0.640$)
	$\mathbf{\Delta d}_{t,o} = \{0.022, 0.199, 0.246\}$)		
	$NC = 1$ ($w_{NC} = 0.800$)		
$w_{nodes} = 1.0$	$w_{objects} = 0.734$	$w_{nodes} = 1.0$	$w_{objects} = 0.594$
$w_{58\to57} = \frac{1}{2}(w_{\text{nodes}} + w_{\text{objects}}) = 0.867$		$w_{26\to25} = \frac{1}{2}(w_{\text{nodes}} + w_{\text{objects}}) = 0.797$	

- **Pixel MSE** (Wang & Bovik, 2009): In our adaptation, we determine the localization as the edge with the smallest accumulative pixel-wise mean squared error (MSE) based on reference images.
- **SSIM** (Wang et al., 2004): Structural similarity index measure (SSIM) is used to measure the similarity between two images considering the distribution of pixel values, contrast and structure. In our adaptation, the edge with the highest accumulative similarity to the observations is localized to the current edge.

Baseline methods have been integrated into our system replacing the 2D object detection layer. Here, MSE and SSIM weights replace, respectively, the term corresponding to the distribution of objects $w_{i,\text{objects}}$ in the expression (1) for our approach. Fig. 15 represents the values of the first and third-order matrices (W'_1 and W'_{123}) for our method, as well as for MSE and SSIM on test sequence 1. From the results of W'_1 we can observe that MSE and SSIM achieve better values for the AEL factor than our method (0.11 and 0.13, respectively, for MSE and SSIM compared to 0.90 for our method) in estimations of first order. In fact, they estimate correctly 8 real edges considering sections of first order. However, the high visual similarity of our building translates into poor discrimination capacity of MSE and SSIM. Thus, they exhibit a small margin of variation between the weight of the real edge and the one with the highest score in case of failure. This is the reason that justifies the degradation of MSE and SSIM as the order of temporal and spatial integration increases. Unlike our method, which was able to locate correctly since iteration 7 on test sequence 1 (see

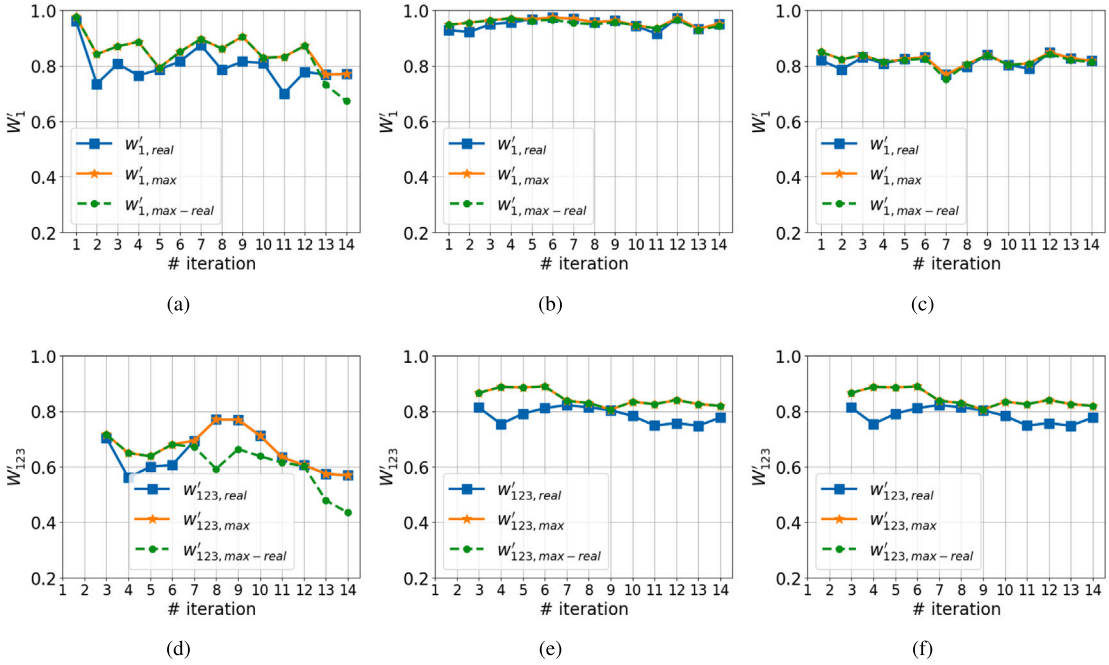


Fig. 15. Performance comparison of localization with baseline methods on test sequence 1. (a, b and c). Weights of matrix W'_1 for our method ($AEL = 0.90$), MSE ($AEL = 0.11$) and SSIM ($AEL = 0.13$), respectively. (d, e and f). Weights of matrix W'_{123} for our method ($AEL = 0.32$), MSE ($AEL = 0.85$) and SSIM ($AEL = 0.93$), respectively including the trajectory continuity modification.

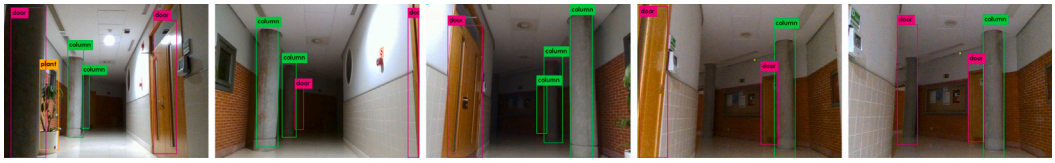


Fig. 16. Examples of visual detections in test sequences 1 and 2 under night-time lighting conditions.

Fig. 15(d)) based on the third order matrix, MSE and SSIM approaches were not capable of locating themselves with stability by using third order matrices and consistency of continuity in route (see Figs. 15(e) and 15(f)). Moreover, our proposed method also outperformed baseline methods on test sequence 2, where the system localized the self-position accurately since the third iteration. In this case, the MSE method did not localize itself until the last iteration (iteration 14) while SSIM was not able to achieve it.

4.5. Generalization capability

To test the generalization capability of the proposed method, it has been evaluated under different conditions: lighting changes and presence of people. Furthermore, the system has been tested in a new scenario different from that of the previous experiments. Firstly, since variations in lighting can have an impact on visual object detection, test sequences 1 and 2 were captured again in nighttime lighting (see Fig. 16). In this instance, location estimations were correct from iteration 8 and iteration 3 in sequences 1 and 2, respectively. The findings, which showed resilience to controlled illumination changes in indoor environments, were consistent with those observed in daylight.

Secondly, we intentionally took captures of a second order route (17→19→21) with the presence of people to show the impact of interaction with humans on the performance. Fig. 17 shows some result images at the output detector as examples. Here, we can observe some of the situations considered: people in front of the platform, crossing from one side to another, or in static groups. It is interesting to verify that few objects were completely occluded from the agent's point of view and not detected during the entire process (for example, a pair of doors in the first edge and a door and a column in the second one). Figs. 17(f) to 17(i) include the topological maps generated under presence of people versus the reference maps. Despite discrepancies between the distribution of objects due to occlusions, the system exhibited robustness and was able to self-locate correctly since the second iteration.

Finally, we tested our algorithm in another different real-world environment to check its reliability and generalizability. Specifically, these additional experiments were conducted on the ground floor of the Faculty of Nursing and Physiotherapy of the University of Alcalá. It is a medium-sized building with a square floor of 30 × 34 m. The building has a structure based on corridors but in this case, there is not as high a degree of symmetry as in the first test environment. Fig. 18(a) depicts the topological map generated from the reference dataset. The structure includes a total of 12 nodes (5 'T'-type nodes, 5 'End'- nodes and 2 'L'-type nodes) and 24 bidirectional connecting edges for the two directions. In this case, we did not collect a validation set, since the goal was to test if the algorithm was able to localize itself maintaining the same hyperparameter values that were previously adjusted in the previous scenario and without needing a new optimization of the model. Despite the presence of other relevant objects in this environment, such as corkscrews or sofas, the same nine object classes (window, door, fire extinguisher, …) that had been used in the previous experiments were also used here as beacons without adding others. Figs. 18(b) to 18(f) show examples of some outputs of the visual object detector in this environment. Because the number of edges is relatively

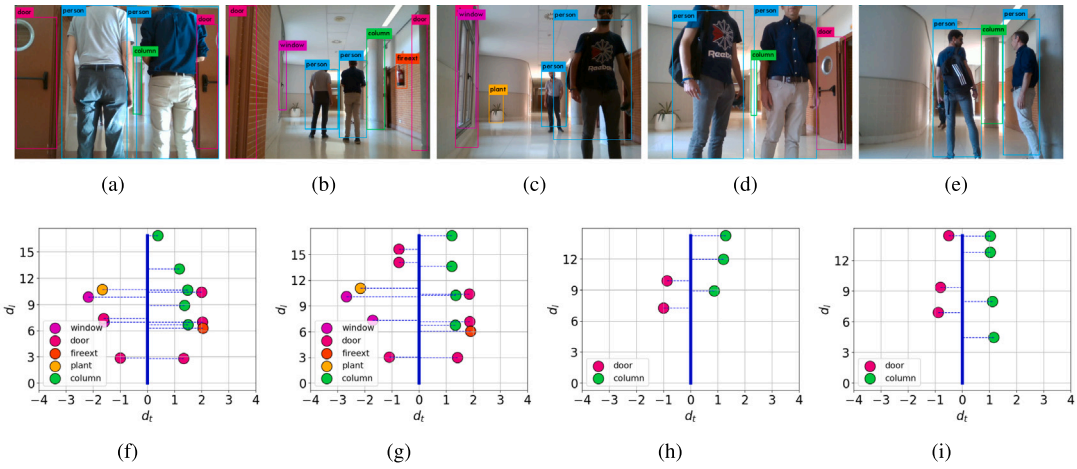


Fig. 17. Example of performance in presence of humans. (a,b,c,d,e) Visual detections in human interactive environment. Topological map (f) generated in presence of humans vs. the reference map (g) for the edge 17→19. (h,i) Idem for topological maps of the edge 19→21.

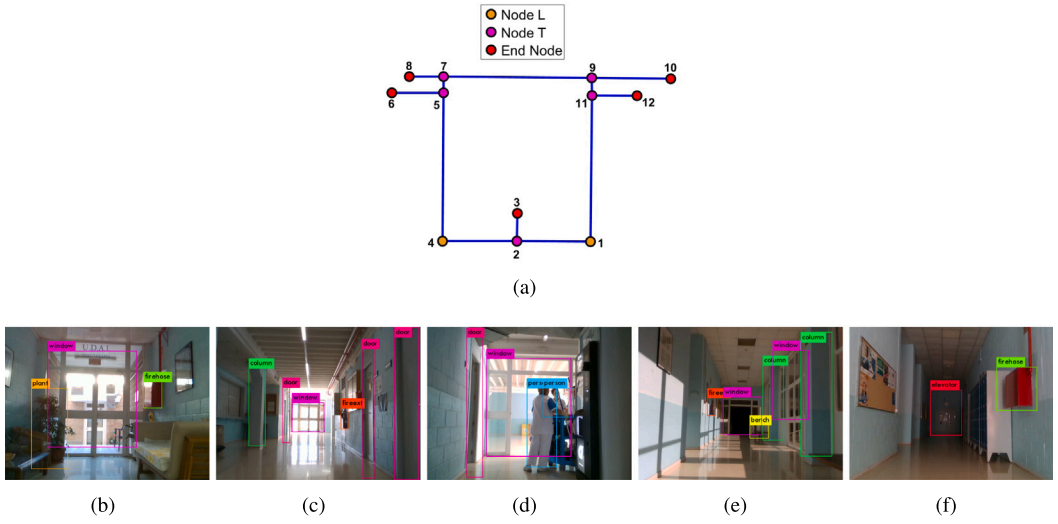


Fig. 18. Map of the second environment used in our experiments (Faculty of Nursing and Physiotherapy of the University of Alcalá). (a) Topological map. (b,c,d,e,f). Examples of visual detections in this building.

low in this building, we have defined two new test trajectories of only three sections: TS_3= 4→2→3→2 and TS_4= 2→1→11→9. In this case, the routes estimated by the system coincided exactly with the two real routes in each of the iterations of first, second and third order. In detail, the weights achieved by the third order matrix are W'_{123} = 0.8097 and W'_{123} = 0.5747, respectively for TS_3 and TS_4, while the weights of the second candidates are: 0.5651 and 0.345, respectively. The results demonstrate that moderate visual discrimination between edges in this building is sufficient for the system to localize correctly and that the estimate remains stable from the first iteration. We can conclude that results support the validity of our approach and its generalization capacity to other environments without the need for any new hyperparametric optimization of the model.

5. Conclusions

In this paper, we have presented a novel indoor localization model based on multisensory integration to lessen high levels of ambiguity in complex symmetrical environments. This method is inspired by a two-level topological structure defined by characteristic objects and geometric information. Based on assessing temporal and spatial consistency, an analytic model has been implemented that compares the topological structure generated in the exploration process with a topological map of reference.

Experimental results have demonstrated the system is able to locate itself in two real environments and localization ambiguity is reduced as the mobile platform moves on the route. The following conclusions were drawn from the analysis of results:

- The incorporation of graph structure and natural beacons into the topological map enables accurate estimation, even in scenarios with multiple corridors or visually and geometrically similar sections.
- The experiments highlight that the distribution of characteristic objects plays a crucial role in localization within corridor environments, surpassing the significance of node types and corridor lengths.
- The double trade-off between performance and computational load has allowed us to determine the optimal order of the weight matrix to integrate the historical record of locations and obtain consistency over time without discontinuities in the estimated trajectory.
- The system demonstrates robustness to controlled lighting changes typical of indoor environments, basically due to the high accuracy of visual detectors in the scientific community. It is achieved

through data augmentation techniques introducing variations in color, lighting conditions, and contrast.

As a limitation, the algorithm has been designed for environments consisting of corridors or narrow areas, with a maximum width of around 4 m in our experiments. However, the approach is not constrained by the width of corridors. The estimation of objects on each edge is only limited by the maximum distance range (approximately 10 m) of our RGB-D camera model and its field of view. In future work, we plan to extend our approach to large indoor spaces away from a layout based on sections or corridors. For larger spaces, we will introduce the concept of 'Open-Space node', which will require a different treatment for localization. Additionally, even with people present, we have assumed that the world is static in this paper; extending to non-stationary environments still presents a significant challenge. As a result, opening and closing doors will, respectively, create and remove routes, which will cause typology changes in the nodes of the topological map.

CRediT authorship contribution statement

Sergio Lafuente-Arroyo: Conceptualization, Methodology, Software, Writing – original draft. **Saturnino Maldonado-Bascón:** Conceptualization, Methodology, Supervision. **Diego Delgado-Mena:** Software, Data curation, Validation. **Carlos Gutiérrez-Álvarez:** Data curation, Validation. **Francisco Javier Acevedo-Rodríguez:** Conceptualization, Writing – review & editing.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the project AIRPLANE, with reference PID2019-104323RBC31, of Spain's Ministry of Science and Innovation.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2023.122561.

References

Arandjelovic, R., Gronát, P., Torii, A., Pajdla, T., & Sivic, J. (2015). NetVLAD: CNN architecture for weakly supervised place recognition. In *Computer vision and pattern recognition* (pp. 5297–5307). arXiv:1511.07247.

Ayyalasomayajula, R., Vasisht, D., & Bharadia, D. (2018). BLoc: CSI-based accurate localization for BLE tags. In *Proceedings of the 14th international conference on emerging networking experiments and technologies* (pp. 126—138). Association for Computing Machinery.

Chang, M., Gupta, A., & Gupta, S. (2020). Semantic visual navigation by watching youtube videos. In *NeurIPS*.

Chaplot, D., Salakhutdinov, R., Gupta, A., & Gupta, S. (2020). *Neural topological SLAM for visual navigation* (pp. 12872–12881).

Chen, C.-H., Chen, P.-W., Chen, P.-J., & Liu, T.-H. (2021). Indoor positioning using magnetic fingerprint map captured by magnetic sensor array. *Sensors*, *21*(17).

Chen, K., de Vicente, J. P., Sepulveda, G., Xia, F., Soto, A., Vázquez, M., & Savarese, S. (2019). A behavioral approach to visual navigation with graph localization networks. CoRR abs/1903.00445.

Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning*, *20*, 273–297.

Dellaert, F., Fox, D., Burgard, W., & Thrun, S. (1999). Monte Carlo localization for mobile robots. In *Proceedings 1999 IEEE international conference on robotics and automation, vol. 2* (pp. 1322–1328).

Diallo, A., Lu, Z., & Zhao, X. (2019). Wireless indoor localization using passive RFID tags. *Procedia Computer Science*, *155*, 210–217.

Ge, G., Zhang, Y., Wang, W., Jiang, Q., Hu, L., & Wang, Y. (2022). Text-MCL: Autonomous mobile robot localization in similar environment using text-level semantic information. *Machines*, *10*(3: 169).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hernández, N., Parra, I., Corrales, H., Izquierdo, R., Ballardini, A. L., Salinas, C., & García, I. (2021). WiFiNet: WiFi-based indoor localisation using CNNs. *Expert Systems with Applications*, *177*, Article 114906.

Kendall, A., & Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning. CoRR abs/1704.00390.

Kendall, A., Grimes, M., & Cipolla, R. (2015). Convolutional networks for real-time 6-DOF camera relocalization. CoRR abs/1505.07427,

Kuipers, B., & Byun, Y.-T. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, *8*(1), 47–63, Special Issue Toward Learning Robots.

Lafuente-Arroyo, S., Maldonado-Bascón, S., López-Sastre, R. J., Molina-Cantero, A. J., & Martín-Martín, P. (2022). LIDAR signature based node detection and classification in graph topological maps for indoor navigation. In *Lecture notes in computer science: vol. 13256, Pattern recognition and image analysis - 10th Iberian Conference, IbPRIA 2022, Aveiro, Portugal, May 4-6, 2022, proceedings* (pp. 390–401). Springer.

Li, L., Yang, M., Guo, L., Wang, C., & Wang, B. (2017). Precise and reliable localization of intelligent vehicles for safe driving. *531*, (pp. 1103–1115).

López-Sastre, R. J., Baptista-Ríos, M., Acevedo-Rodríguez, F. J., Pacheco-da Costa, S., Maldonado-Bascón, S., & Lafuente-Arroyo, S. (2021). A low-cost assistive robot for children with neurodevelopmental disorders to aid in daily living activities. *International Journal of Environmental Research and Public Health*, *18*(8).

Luo, C., Cheng, L., Chan, M. C., Gu, Y., Li, J., & Ming, Z. (2017). Pallas: Self-bootstrapping fine-grained passive indoor localization using WiFi monitors. *IEEE Transactions on Mobile Computing*, *16*(2), 466–481.

Mannay, K., Ureña, J., Hernández, Á., Machhout, M., & Aguili, T. (2020). Characterization of an ultrasonic local positioning system for 3D measurements. *Sensors*, *20*(10).

Maruyama, Y., Kato, S., & Azumi, T. (2016). Exploring the performance of ROS2. In *Proceedings of the 13th ACM SIGBED international conference on embedded software* (pp. 1–10).

Niwa, T., Taguchi, S., & Hirose, N. (2022). Spatio-temporal graph localization networks for image-based navigation. In *IEEE/RSJ international conference on intelligent robots and systems*.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., & Ng, A. (2009). ROS: an open-source Robot Operating System. In *Proc. of the IEEE Intl. Conf. on robotics and automation (ICRA) workshop on open source robotics*.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv: 1804.02767.

Ridolfi, M., Velde, S., Steendam, H., & De Poorter, E. (2016). WiFi ad-hoc mesh network and MAC protocol solution for UWB indoor localization systems. (pp. 1–6).

Sattler, T., Leibe, B., & Kobbelt, L. (2017). Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(9), 1744–1756.

Savinov, N., Dosovitskiy, A., & Koltun, V. (2018). Semi-parametric topological memory for navigation. CoRRabs/1803.00653,

Shah, D., Eysenbach, B., Kahn, G., Rhinehart, N., & Levine, S. (2021). ViNG: Learning open-world navigation with visual goals. In *IEEE international conference on robotics and automation*.

Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., ... Batra, D. (2021). Habitat 2.0: Training home assistants to rearrange their habitat. In *NeurIPS*.

Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., & Torii, A. (2018). InLoc: Indoor visual localization with dense matching and view synthesis. CoRR abs/1803.10368,

Talwar, D., & Jung, S. (2019). Particle Filter-based Localization of a Mobile Robot by Using a Single Lidar Sensor under SLAM in ROS Environment. In *2019 19th international conference on control, automation and systems* (pp. 1112–1115).

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. MIT Press.

Tomatis, N., Nourbakhsh, I., & Siegwart, R. (2001). Combining topological and metric: A natural integration for simultaneous localization and map building. In *Proceedings of the fourth European workshop on advanced mobile robots*.

Valada, A., Radwan, N., & Burgard, W. (2018). Deep auxiliary learning for visual localization and odometry. CoRR abs/1803.03642,

Valencia, R., Saarinen, J., Andreasson, H., Vallvé, J., Andrade-Cetto, J., & Lilienthal, A. J. (2014). Localization in highly dynamic environments using dual-timescale NDT-MCL. In *2014 IEEE international conference on robotics and automation* (pp. 3956–3962).

Walch, F., Hazirbas, C., Leal-Taixé, L., Sattler, T., Hilsenbeck, S., & Cremers, D. (2016). Image-based localization with spatial LSTMs. CoRR abs/1611.07890.

Wang, Z., & Bovik, A. C. (2009). Mean squared error: Love it or leave it? A new look at Signal Fidelity Measures. *IEEE Signal Processing Magazine*, *26*(1), 98–117.

Wang, Z., Bovik, A., Sheikh, H., & Simoncelli, E. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612.

Wiyatno, R. R., Xu, A., & Paull, L. (2021). Lifelong topological visual navigation. CoRR abs/2110.08488,

Wolf, J., Burgard, W., & Burkhardt, H. (2005). Robust vision-based localization by combining an image-retrieval system with Monte Carlo localization. *IEEE Transactions on Robotics*, *21*(2), 208–216.

Ye, X., Lin, Z., Li, H., Zheng, S., & Yang, Y. (2018). Active object perceiver: Recognition-guided policy learning for object searching on mobile robots. CoRR abs/1807.11174,

Zhang, Q., Wang, P., & Chen, Z. (2019). An improved particle filter for mobile robot localization based on particle swarm optimization. *Expert Systems with Applications*, *135*, 181–193.