Autonomous Robots (2026) 50:6
https://doi.org/10.1007/s10514-025-10221-8

Estimating map completeness in robot exploration

Matteo Luperto[1] · Marco Maria Ferrara[2] · Matteo Princisgh[1] · Giacomo Boracchi[2] · Francesco Amigoni[2]

Received: 25 April 2025 / Accepted: 4 October 2025
© The Author(s) 2025

Abstract

We present a novel method that, given a grid map of a partially explored indoor environment, estimates the amount of the explored area in the map and whether it is worth continuing to explore the uncovered part of the environment. Our method is based on the idea that modern deep learning models can successfully solve this task by leveraging visual clues in the map. Thus, we train a deep convolutional neural network on images depicting grid maps from partially explored environments, with annotations derived from the knowledge of the entire map, which is not available when the network is used for inference. We show that our network can be used to define a stopping criterion to successfully terminate the exploration process when this is expected to no longer add relevant details about the environment to the map, saving more than 35% of the total exploration time compared to covering the whole environment area.

Keywords Robot exploration · Stopping criterion for robot exploration · Map completeness · 2D grid maps

1 Introduction

When exploring an initially unknown indoor environment, an autonomous mobile robot builds a representation, or *map*, for the environment by iteratively performing a sequence of steps (Thrun, 2002). First, the robot identifies a set of reachable candidate locations within the known portion of the environment represented by the current map. Usually, these candidate locations are at the map boundaries, called *frontiers*, between known and unknown parts of the environment. The robot then selects the most promising location to visit, according to an *exploration strategy*. Finally, while reaching the selected location, the robot updates its map with newly acquired sensor data. Then, the cycle repeats.

In some operational contexts, such as Urban Search and Rescue (Liu & Nejat, 2013), the exploration process stops after a pre-defined (usually not-so-long) amount of time since the application requires to quickly provide (possibly incomplete) information about the environment to the human rescuers. In other applications, the exploration is stopped when the process is no longer acquiring relevant information, for example, when no frontiers exceeding a minimum length are still to be explored (Placed & Castellanos, 2022). A method that decides when an exploration process should stop is called a *stopping criterion*. Although the need for stopping criteria is well recognised, identifying robust stopping criteria is a critical challenge and research in the field is still rather limited (Cadena et al., 2016; Placed & Castellanos, 2022; Placed et al., 2023).

Good stopping criteria are of paramount importance. Indeed, the later stages of a typical exploration process often provide little contribution to the completion of the final map, at a high cost, as shown in Fig. 1: while the robot moves back and forth to map small portions of the environment, mapping inconsistencies and unrecoverable states become likely and can degrade and even invalidate the entire mapping process (Placed & Castellanos, 2022).

Often, robots are not aware of the current status of their exploration missions, that is, if the exploration run has just started or if it is close to conclusion. This knowledge could enable specific exploration strategies for single- and multi-robot systems, where robots can be assigned different roles as explorers and relays (Amigoni et al., 2019). A proxy to estimate the status of an exploration mission is the percentage of the total area of the environment that is covered by the current map. This percentage can be calculated if we

✉ Matteo Luperto
matteo.luperto@unimi.it

Francesco Amigoni
francesco.amigoni@polimi.it

[1] Universitá degli Studi di Milano, Milan, Italy

[2] Politecnico di Milano, Milan, Italy

Published online: 26 December 2025



(a) $t = 90\,\text{min}$.



(b) $T = 150\,\text{min}$.

Fig. 1 The map (**a**) after $t = 90\,\text{min}$ of exploration already represents almost all of the environment (the percentage of the mapped area is $A_t = 0.98$). The portions of the environment that are still to be explored, highlighted in blue in (**a**), are uninteresting corners and portions of rooms scattered around the whole environment. The only unexplored area that may be of interest is the room on the left, which is only partially explored; however, from the rest of the map, the shape of such room can be easily estimated. The exploration run is ended at $T = 150\,\text{min}$ (**b**) when all the frontiers have been visited. During the time interval $T - t$, the robot moves back and forth to explore the remaining frontiers, possibly jeopardizing the mapping process

assume to know the total area (Placed & Castellanos, 2022). However, to the best of our knowledge, there are no established methods to reliably predict the percentage of explored area when the complete map of the environment or the total area is unknown.

In this paper, we present a method that evaluates the partial map of an environment and estimates the advancement of the exploration process. Our intuition is that the exploration status can be estimated using visual clues in the map, therefore we transform the grid map obtained during exploration in an image, and train a Convolutional Neural Network (CNN) to predict whether the map can be considered explored or not. We exploit the fact that this is a well-suited task for CNNs, since maps of indoor environments exhibit very peculiar patterns that allow humans to correctly assess – in most cases – whether there is still some potentially large area to explore. For example, looking at the partial map of Fig. 1a, it is intuitive to say that this represents an environment that is almost completely explored. Our network, which is trained on an extensive dataset of maps obtained during the exploration of indoor environments, is used to:

- Defining a stopping criterion that determines whether a partial map under scrutiny can be considered as sufficiently explored or not,
- Estimating the status of the exploration process as the percentage of the total area explored by the robot (without requiring any knowledge about the entire environment).

We also provide an automated training set preparation procedure that we use to train our network. Experimental results show that our method can significantly reduce the exploration time while ensuring a good level of completeness of the

map. As our method is based on 2D grid maps, it is agnostic of the exploration strategy employed to build the map, and can be used to increase the awareness of any exploration strategy. At the same time, we do not require any previous knowledge or priors on the environment that is explored.

2 Related work

In the context of robot exploration of initially unknown (static) environments, the main (and most studied) issue a robot faces is to decide which is the next location to reach to progress the exploration (Lluvia et al., 2021; Stachniss & Burgard, 2005). Over the years, different exploration strategies have been proposed (Stachniss, 2009). A common one is to identify a few candidate locations at the boundaries between the explored and unexplored portions of the environment, and to select the most promising location to reach, according to some exploration strategy. These boundaries are called frontiers, and their ranking is commonly based on their expected *information gain*, i.e., an estimate of the amount of knowledge to be acquired by the robot when visiting the locations, and on their *distance* from the current location of the robot (Yamauchi, 1997; Lluvia et al., 2021; Amigoni et al., 2017).

While the general goal of an autonomous mobile robot exploring initially unknown environments is to obtain their complete maps, there are often additional desiderata to address, such as maximizing the explored area within a time budget (Amigoni et al., 2013) or maximizing the quality of the reconstructed map (Placed et al., 2023). Differently from the many studies on exploration strategies, some works highlight that deciding when to stop exploration is a particularly relevant, yet under-investigated, problem (Cadena et al., 2016; Placed & Castellanos, 2022; Placed et al., 2023). In Placed and Castellanos (2022), an overview of the stopping criteria that are commonly used to end exploration is presented, as well as a discussion of their limitations. A straightforward criterion is to stop the exploration process when a time budget is reached (Leung et al., 2008); however, the time budget should be tuned for each environment and exploration strategy, and there is no guarantee that the map obtained by the robot when the time budget expires is complete.

Other works end the exploration when no candidate location is left (Yamauchi et al., 1999). A positive aspect of this stopping criterion is that it guarantees that the map is complete; on the negative side, this completeness is obtained at the expense of a long exploration time, where most of the time is spent reaching far–away uninteresting locations that have been left behind in the initial part of exploration (Ericson et al., 2021; Luperto et al., 2020) (see also the example of Fig. 1a). This is a well known problem that motivated exploration strategies that predict the shape of the unobserved parts of the environment and compute their information gain (Luperto et al., 2021; Katsumata et al., 2022; Shrestha et al., 2019; Luperto et al., 2020). As pointed out in Ericson et al. (2021), these map-prediction exploration strategies allow a robot to greedily map most of the environment in a short time. However, the robot can spend up to 71% of the total time in exploring marginal details of the environment that have been left behind, but that constitute about 10% of the area of the environment.

Some works assess the percentage of the total explored area to stop the exploration (Amigoni et al., 2013). However, this criterion requires either the a priori knowledge of the size of the map or a way to infer it. Moreover, the fact that most of the area is already explored does not guarantee that all the areas of interest in the environment have been mapped. In Bircher et al. (2018), the authors apply a strategy based on Rapidly-exploring Random Trees (RRTs) to exploration, using the size of the RRT as a stopping criterion. An alternative approach compares maps acquired at different times (e.g., after 10 min) and stops exploration when the maps are nearly identical, indicating that no new areas were observed (Amigoni et al., 2018). However, this method may fail to ensure mission completion, especially when the robot is travelling between distant frontiers.

Other works, like (Stachniss & Burgard, 2003), leverage Information Theory and adopt the map entropy as a stopping criterion: the exploration is stopped when the map entropy is below a pre-defined threshold, or when the map entropy increases without adding new areas. A similar approach is applied in Ghaffari Jadidi et al. (2019), where entropy saturation in the current map is considered as a stopping criterion. In Stachniss and Burgard (2005), these methods stop the exploration and mapping process in dynamic environments. However, these techniques do not guarantee that the exploration process is fully completed, often require additional knowledge about the environment, like the footprint or the size of the map, and might become difficult to employ when some locations (cells in grid maps) are unreachable or unobservable (Placed & Castellanos, 2022; Stachniss & Burgard, 2003). The work of Placed and Castellanos (2022) proposes to use the Theory of Optimal Experimental Design (TOED) – which estimates the utility of a variable by a statistical analysis of its variance – to overcome some of the above limitations, defining a stopping criterion that considers the completion of the task assigned to the robot. In particular, Placed and Castellanos (2022) use a D-optimality criterion, a statistical measure of the map utility, to stop the exploration when no new knowledge is added to the map. However, such a stopping criterion is intertwined with a specific exploration framework that uses TOED, making the

approach difficult to generalize to other exploration strategies, such as the frontier-based ones.

Compared with the above approaches, our solution does not need any previous knowledge of the environment being explored, it is task–independent, and it is agnostic of the exploration strategy employed to build the maps.

As mentioned earlier, several recent methods (Luperto et al., 2021; Katsumata et al., 2022; Shrestha et al., 2019; Ericson & Jensfelt, 2024; Tao & Iceland, 2024; Ho et al., 2025; Song et al., 2025) leverage inpainting deep neural networks to *predict*, from a partial 2D map of the environment, a plausible (yet possibly inaccurate) status of the occupancy of unobserved parts of the environment. Then, the predicted map is used to inform the exploration process by estimating the explorable area (Baek et al., 2025; Ho et al., 2025; Georgakis et al., 2022). The maps predicted using these methods are often inaccurate (Ho et al., 2025), overestimating the area directly observable from each frontier and also by hallucinating the existence of explorable areas in unreachable and not existing places (e.g., behind a wall). While these maps can be effectively used to estimate the information gain of a frontier, that is to identify the most promising frontier to explore, it is not straightforward how to use them to develop a stopping criterion for exploration, due to the high uncertainty of the predicted map. We further discuss this issue in Sect. 5.1. Note that, similarly to these methods, we also use deep learning models and we leverage the visual representation of the current explored map, but we solve a significantly different (and simpler) problem: we directly use the currently available map to estimate the status of the exploration mission, without predicting the occupancy of the unobserved parts of the environment.

3 The problem

In this paper, we consider an autonomous mobile robot in an initially unknown environment that explores it using a frontier-based exploration strategy (Yamauchi, 1997), and we propose a method to evaluate the current status of the exploration process. Our method simply relies on an image depicting the current grid map built by the robot, and addresses two key tasks:

(*a*) Classifying the current map to satisfactorily represent (or not) the whole environment. The classification output defines the stopping criterion.

(*b*) Estimating the percentage of the area of the environment that is represented in the current partial map, thus estimating the progress of the exploration task.

Our solution can significantly improve the awareness of the robot about the map during exploration. While (*a*) can be used to stop the exploration earlier than conventional methods, thus saving time (Sect. 4.5), (*b*) can be used by a high-level decision-making process of single- and multi-robot systems, to plan the robot activity. Note that the percentage of the explored area is (dually) linked to the amount of information that can be still obtained by continuing exploration.

We train a deep CNN for each of the two tasks (*a*) and (*b*). The stopping criterion (*a*) is modeled as a visual recognition problem where the classification task is to distinguish between maps that are `not-explored` yet from those that are sufficiently `explored`. The first class contains maps that a human (who knows the full map of the environment) would consider incomplete, because they miss areas of interest, like entire rooms that have not been mapped. The second class contains maps that a human would consider complete enough to represent the full environment. In this second case, some small parts of the environments (like corners) may not be entirely mapped yet, but such parts are not considered relevant. An example of an `explored` map can be seen in Fig. 1a. The second task (*b*), namely estimating the percentage of the explored area, is modeled as a regression problem using the same CNN backbone that solves task (*a*).

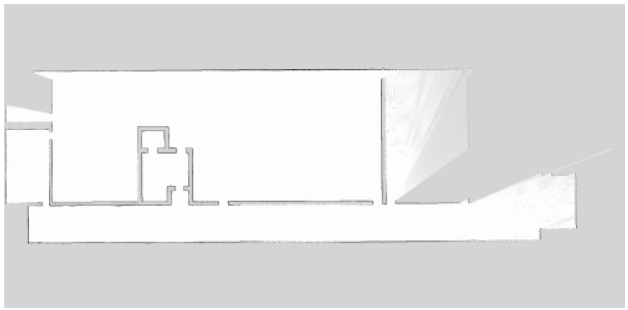
The input for our method is a partial grid map M_t built up to time t, where each cell can contain three possible values: free (white), obstacle (black), or unknown (grey). Therefore, our method can be applied to any grid map M_t regardless of how it was obtained and our method can be used in combination with any exploration strategy.

Formally, given an autonomous mobile robot that explores an initially unknown environment, we indicate with $A_t \in [0,1]$ the actual percentage of the area covered in the current map M_t at time t, and with $\hat{A}_t \in [0,1]$ the percentage of the explored area estimated by our method. To formulate our problem, consider an exploration process that takes T time steps to obtain a complete map $M_T = M$ (namely, M_T is a map for which $A_T = 1$). A perfect (ideal) stopping criterion with full knowledge of the complete map M should stop the exploration at time $\bar{t} \leq T$ when $M_{\bar{t}} \approx M$, namely when the free and obstacle cells in $M_{\bar{t}}$ match free and obstacle cells in M and the portion of the environment left unexplored, i.e., the free and obstacle cells in M that are labeled as unknown in $M_{\bar{t}}$, represent corners and small gaps that are not worth exploring (according to human judgement, see above).

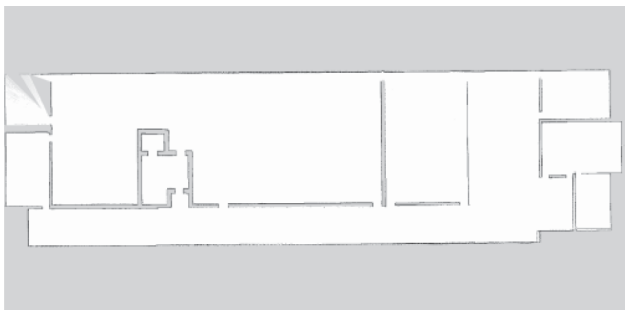
Our method, without knowing the complete map M, defines a stopping criterion that estimates the timestamp $\hat{t} \leq T$ when the exploration can be interrupted and for which $M_{\hat{t}} \approx M$ and the unexplored parts of the environment are not relevant. Note that different partial maps M_t with the same portion of the explored area A_t can be considered

as sufficiently representing the whole environment or not, depending on how the unexplored portion of the area is distributed across the space (e.g., spread in many corners or concentrated in an unexplored room).

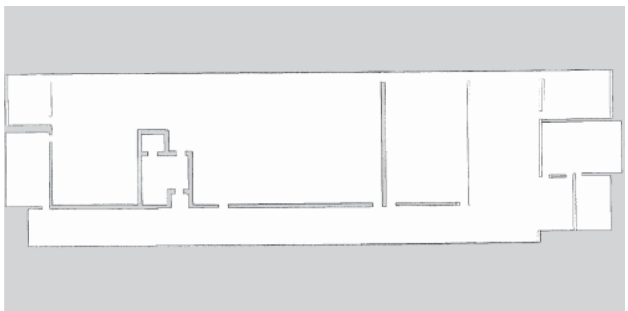
To solve the task (*a*), relative to the stopping criterion, our method identifies the first map classified as `explored`, thus minimizing the normalized time error $err_t = |\bar{t} - \hat{t}|/T$. Note that our goal is to avoid cases where the stopping criterion is triggered too early, namely when $\hat{t} < \bar{t}$, in order to avoid missing some interesting parts of the map. Triggering the stopping criterion at a time $\hat{t} > \bar{t}$ is less critical, as it is only expected to reduce the amount of time saved without affecting relevant knowledge of the environment. To solve the second task (*b*), namely estimating the percentage of the explored area, our method attempts to minimize the area estimation error $err_A = |A_t - \hat{A}_t|$.



(a) Partial map M_t labeled as `not-explored` $A_t = 0.77$.



(b) Partial map M_t labeled as `explored` $A_t = 0.99$.



(c) Complete map M.

Fig. 2 Examples of partial maps and labels from our dataset

4 The proposed solution

4.1 Input definition and CNN models

Our main intuition consists in casting the assessment of the exploration status in a visual recognition problem. Given a map M_t, it is centred, scaled, and resized into a 500×500 pixels grayscale image I_t. Thus, the scaling factor is not fixed and depends on the size of the original map M_t. Each pixel of this image can take only three values: 0 (black) for obstacles, 1 (white) for free areas, and 0.5 (grey) for unknown areas (see Fig. 2).

To solve the classification (*a*) and percentage of explored area estimation (*b*) tasks previously described, we train two CNNs sharing the vast majority of their layers and differing only for their output layers, as they address different tasks (classification and regression). In particular, the binary classification network $\mathcal{C}$ has two output neurons (corresponding to the `explored` and `not-explored` classes) with a softmax activation function.[1] The network $\mathcal{C}$ is trained by minimizing the binary cross-entropy as the loss function. The regression network $\mathcal{R}$ has a single output neuron with a ReLU activation function and is trained to minimize the mean squared error. The feature extraction component shared in both networks is from the EfficientNet B1 (Tan & Le, 2019), which is pre-trained on ImageNet (Deng et al., 2009) followed by a convolutional layer, average pooling, and dropout. When training both $\mathcal{C}$ and $\mathcal{R}$, we perform transfer learning by freezing all the parameters of the first 200 layers, using Kingma and Ba (2014).

We selected the EfficientNet B1 as a good tradeoff between accuracy and computational complexity (enabling timely inference even on a CPU). However, our method allows different backbones to be used instead of EfficientNet as feature extraction components. In Sect. 5.7, we evaluate alternative backbones: a ResNet (He et al., 2016) and two transformer-based architectures, namely a Vision Transformer (ViT) (Dosovitskiy et al., 2021) and a hierarchical shifted-window vision transformer (Swin Transformer) (Liu et al., 2022).

The full code and training details are available in our repository: https://aislabunimi.github.io/explore-stop/.

4.2 Dataset

To train and test our models, we use a set of partial maps $\{M_t\}$ collected at different exploration stages, each one paired with the corresponding label (`explored` or `not-explored`) and the percentage of explored area A_t, as illustrated in the next section. These partial maps are acquired

[1] During inference, we set a threshold θ to define the minimum probability estimate returned by the network to claim the map is explored.

incrementally during the robot exploration runs performed for a previous work (Amigoni et al., 2018; Luperto et al., 2021), where we collected over 20k partial maps at different stages of exploration from 100 different large–scale indoor environments. All maps are free of clutter, like objects or furniture.

Every environment is autonomously explored multiple times by a simulated robot using the method in Yamauchi (1997); we equally weigh the expected information gain (IG) of a frontier (i.e., the footprint of a simulated LiDAR over the unknown area that is beyond the frontier) and the distance of the robot to the frontier. In each run, due to simulated noise in perception, the robot makes different choices during exploration, follows different trajectories, and ultimately builds different maps. During exploration, partial maps of the environments are saved every 10 min. The number of runs for each environment depends on the complexity of the environment; on average, we collected 36 runs for each environment. See Amigoni et al. (2018); Luperto et al. (2021) for further details.

4.3 Partial maps labelling

We automatically label each partial map M_t as `explored` or `not-explored` using a rule-based approach. First, we compute the portion A_t of the explored area for each partial map M_t by comparing it with the complete map M. More precisely, we count the number of cells (pixels) that differ between them when both M_t annd M are converted to 500×500 pixels grayscale images. This count is then converted into square meters using the map's scaling factor. A_t is then calculated as the total area of differing cells between M_t and M, divided by the total area of M. When the portion of explored map A_t is smaller than a threshold α, the map is `not-explored`. The remaining partial maps can be either labeled as `explored` or not depending on the spatial distribution of the unexplored cells. More precisely, if the unexplored area is scattered in multiple small locations across the environment, as in the example of Fig. 1a, the map is labeled as `explored`. Instead, if there is a large unexplored area, like a full unexplored room, the map is labeled as `not-explored`. We identify all unexplored regions by clustering the unknown cells of the partial map M_t that correspond to free and obstacle cells in M, using DBSCAN (Ester et al., 1996). Then, if the largest unexplored region, expressed as a percentage of the area to M, is greater than a threshold β, the map is labeled as `not-explored`; otherwise, we consider the map as `explored`. The procedure is sketched in Algorithm 1. Examples of maps belonging to the two classes are shown in Fig. 2 while for implementation details please refer to our online repository.

We set different α and β for small, medium, and large environments: 12, 8, and 6 for α; 1.35, 0.8, and 0.5 for β, respectively. These values and the splitting of the maps in the three size categories (small, medium, large) are empirically defined by visually validating the resulting automatically assigned labels. Different sets of values for α and β can be used to obtain different labeling schemas, thus training the network to adopt different termination criteria. As an example, with the values we use, we consider the room at the top-left of the map in Fig. 2as explored. Informally, with our values of α and β, we classify rooms as explored when the majority of their floor area has been mapped, with only minor unmapped sections such as corners remaining. Additionally, a room is considered as explored when the unmapped areas clearly cannot contain doorways or passages that would connect to other rooms. If a single unexplored room is present, we consider the map as `not-explored`. However, if a specific application requires a more (less) extensive coverage of a room before triggering the stopping criterion, more (less) conservative values for α and β can be used instead.

Require: partial map M_t, full map M, thresholds α and β
compare M_t with M and find percentage of explored area A_t
if $A_t < \alpha$ **then**
    return `not-explored`
else
    $\mathcal{C} = \{C_1, C_2, \ldots\} \leftarrow$ cluster unknown cells
    $C_{\max} \leftarrow$ largest cluster in $\mathcal{C}$
    $A_{C_{\max}} \leftarrow$ percentage of area of M covered by $C_{\max}$
    if $A_{C_{\max}} > \beta$ **then**
        return `not-explored`
    else
        return `explored`
    end if
end if

Algorithm 1 Automated labeling of partial maps

4.4 Model training

The environments in the labeled dataset are split into the training, validation, and test sets in a 70 : 15 : 15 ratio. These splits are based on the environments from which the maps are obtained; therefore, environments whose maps are in the training set are not providing maps to validation and test sets, and vice versa. The training set, called D_{IG}, contains about 15k partial maps. To artificially increase the size of our training and validation sets and lower the risk of overfitting, we use image augmentation by changing the orientation and zoom of the original images. We train the model for 24 epochs with a batch size of 8, a learning rate of 10^{-4}, and the Adam optimizer.

4.5 Stopping criterion

Our stopping criterion is triggered when a partial map obtained during the exploration process is labeled as `explored` by our model $\mathcal{C}$. A more conservative approach can be developed by considering additional factors, for example when multiple consecutive partial maps are labeled as `explored` or when the estimated $\hat{A}_t$ returned by model $\mathcal{R}$ exceeds a fixed threshold. In this sense, the estimated percentage of the explored area can be exploited to inform the exploration strategy as discussed in Sects. 1 and 3.

When used online by a robot, the stopping criterion can be evaluated following different events. For instance, the stopping criterion can be evaluated every time a new map is produced by the SLAM pipeline, or when the robot has to select the next location to reach following its exploration strategy, or after a time budget. Thanks to the relatively low computational requirements of our method – which could process a map in less than 0.25 s on a low-end CPU (see next section) – all three options are viable on a standard robot.

A further benefit of our method is that we can leverage saliency maps on the trained CNN classifier to design new exploration strategies. Saliency maps, like Grad-CAM (Selvaraju et al., 2017), are explainability techniques that provide, for each input image, a heatmap that highlights which regions have primarily influenced the network to predict a specific class. Therefore, saliency maps associated with the class `not-explored` can suggest which regions to prioritize in the exploration, as shown in Fig. 3. However, the investigation of novel exploration strategies based on the estimated percentage of explored area and saliency maps is left for future work.

5 Experimental evaluation

In this section, after introducing our baseline stopping criterion (Sect. 5.1), we first assess the performance of our CNN $\mathcal{C}$ in classifying maps as `explored` or `not-explored` (Sect. 5.2), as well as the corresponding exploration time saved by adopting our stopping criterion (Sect. 5.3). We then assess the performance of our CNN $\mathcal{R}$ in estimating the portion of area explored (Sect. 5.4). Moreover, we show the generalizability of our method on a different test set (Sect. 5.5) and some qualitative results that give insights into how our method works (Sect. 5.6). In Sect. 5.7, we investigate how our proposed approach works when different deep neural network architectures are used as backbones. Finally, we apply our method to maps obtained from explorations of real-world environments (Sect. 5.8).

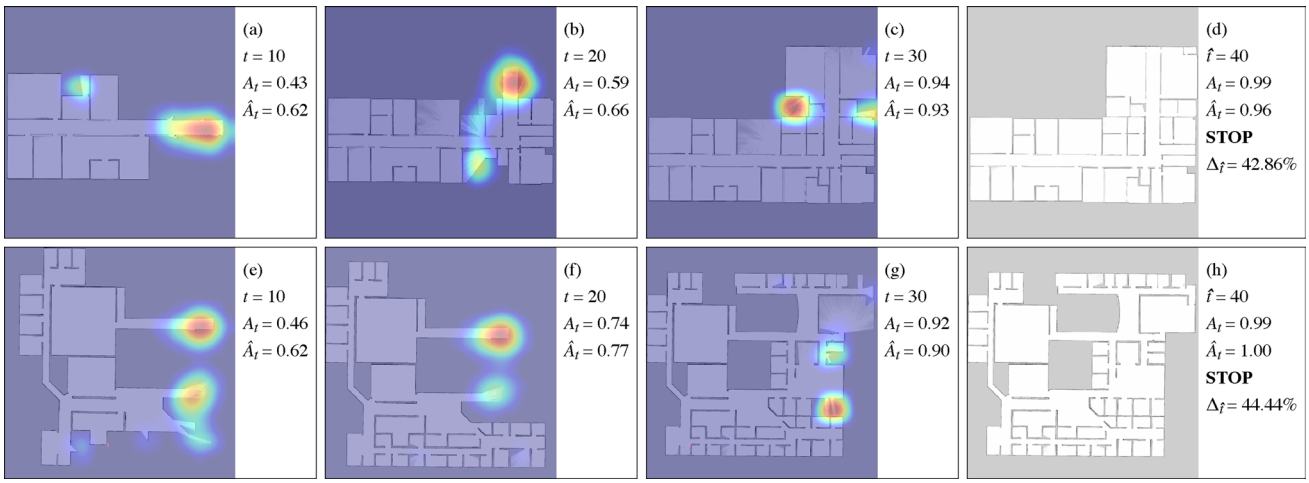


Fig. 3 Incremental examples of our method applied to partial maps obtained after 10, 20, 30, and 40 min of exploration of two environments, from batch-wise assessment. In the environment in the first row (**a**–**d**), the exploration with the baseline stopping criterion ends after $T = 70\,\mathrm{min}$; we reduce such time by 30 min (**d**). In the environment in the second row (**e**–**h**), the exploration with the baseline stopping criterion is concluded after $T = 90\,\mathrm{min}$; we reduce such time by 50 min (**h**). The areas highlighted in the partial maps are the locations that our CCN $\mathcal{C}$ identifies as relevant to make its prediction (red and blue denotes the most and the least relevant regions, respectively)

Our experiments are performed using a test set of maps obtained by incremental exploration of indoor environments, as explained in Sect. 4.2. The environments considered for evaluation have not been used for training and are of different sizes and shapes, varying from small-scale environments with less than 10 rooms and smaller than 100 m^2 to large-scale environments with more than 100 rooms that cover more than 10000 m^2. To assess the benefits of our approach, we run both a batch-wise and an online assessment.

The *batch-wise* (B) assessment allows us to understand the performance of the proposed solution when processing maps that are visually different from each other, thus ensuring the generalisation capabilities of our solution. The batch-wise evaluation is performed on a test set containing 3009 maps obtained from 15 environments. In total, we simulated 613 robot exploration runs; during each run, a snapshot of the current map of the environment was saved every 10 min. The relatively long time interval between two snapshots ensures that two consecutive maps obtained by the robot are significantly different, as in the examples in Fig. 3, where maps obtained in two different robot runs are shown.

The *online* (O) assessment is meant to understand the benefits of using our method during exploration when the robot frequently decides whether to continue exploring, based on its current map. To replicate this setting, we consider maps collected during an exploration run at a higher frequency, namely every 30 s. We consider 10 environments and we perform 5 exploration runs in each environment, for a total of 50 runs, obtaining 2255 maps.

Note that the online assessment cannot be used for a fair evaluation of the map classification performance, since consecutive maps are very similar, due to the higher sampling frequency. Our classifier $\mathcal{C}$ can easily distinguish that all maps obtained during the early (late) stages of explorations are `not-explored` (`explored`), resulting in an optimistic performance assessment. At the same time, the batch-wise assessment cannot provide a precise estimate of the stopping criterion due to the long interval of time between the acquisition of two maps.

We also analyse the impact of the threshold θ for classifying a map as explored or not (Sect. 4.1). We then set $\theta = 0.5$ as a reference and present also the results with a more conservative value $\theta = 0.8$, to reduce the number of partial maps wrongly considered as `explored`.

All the CNNs are trained on a RTX 2070 GPU. The method can be used online, as evaluating the status of the exploration process, given a partial map, takes between 0.1 s and 0.25 s on a GTX 1060 GPU/i7-7700k CPU. More experimental results and details are available in our code repository.

5.1 Baseline stopping criterion

All exploration runs in our experiments continue until a *baseline* stopping criterion is met, which consists of ending the exploration at time T when two partial maps, obtained at times T minus 10 min and T, are similar enough. We measure map similarity as the pixel-wise difference between two images, as in Amigoni et al. (2018). We manually inspect all exploration runs to ensure that the baseline stopping criterion is activated only when the environment is fully explored; if the baseline stopping criterion is triggered too early, the run is removed, and another run is performed.

Our baseline is particularly effective as it does not suffer from the major limitations of many exploration strategies. For instance, some approaches stop exploration when no frontiers are available (Yamauchi et al., 1999), whereas our baseline does not require visiting all frontiers and often terminates earlier, when several (not relevant) frontiers are still present at time T. Stopping criteria based on time-budget fail to ensure full mapping unless the budget is guaranteed to be larger than T (Leung et al., 2008), information-theory-based ones (Ghaffari Jadidi et al., 2019) are difficult to employ in the presence of unreachable or unobservable areas, while others depend on specific strategies (Placed & Castellanos, 2022).

We therefore compare the time $\hat{t}$ when our approach stops the exploration against both the ideal stopping criterion $\bar{t}$ (approximated as the earliest time when our automated labelling, acting as a proxy for an human expert who knows the complete map, marks a partial map as `explored`) and against the challenging baseline T.

In principle, an interesting baseline could be based on map-predictive methods, such as those in Shrestha et al. (2019); Ericson and Jensfelt (2024); Ho et al. (2025), that, given a partial 2D map of the environment M_t, predict a plausible (yet possibly inaccurate) status for the occupancy of the unobserved parts of the environment $\widehat{M}_t$. However, comparing the current partial map M_t with the estimate of the full map $\widehat{M}_t$ does not directly lead to a stopping criterion, as it is not straightforward to define when differences are relevant enough to stop the exploration. While it is clear that the exploration should be stopped when the currently explored area is almost equal to the predicted one, this decision is more difficult to make when the difference between the predicted map $\widehat{M}_t$ and the current map M_t is small but not null. As we have shown previously in the paper, the correct decision on stopping or not the exploration process depends on whether the unknown areas are spread in small portions over the entire map or clustered (namely, on whether small corners or entire rooms are missing in the map). Moreover, determining the relevance of the discrepancy between a predicted map and the current partial map

Table 1 Map classification as `(not)-explored` during batch (B) or online (O) evaluation

	θ	TE (%)	FE (%)	TN (%)	FN (%)	A	P	R
B	0.5	1472 (48.91)	53 (1.76)	1324 (44.00)	160 (5.31)	0.92	0.97	0.91
	0.8	1376 (45.73)	21 (0.70)	1352 (44.93)	260 (8.60)	0.91	0.99	0.84
O	0.5	970 (43.00)	48 (2.12)	1116 (49.49)	121 (5.37)	0.93	0.95	0.89
	0.8	926 (41.06)	28 (1.24)	1136 (50.38)	165 (7.32)	0.91	0.97	0.85

involves considering whether differences between predicted and observed cells should be weighted differently according to their locations. Accounting for the accuracy of a given 2D map prediction method adds further complexity, as such methods often do not report an estimate of the accuracy of their predictions, as discussed in Ho et al. (2025). Several works (Ho et al., 2025; Baek et al., 2025; Georgakis et al., 2022) propose to combine different 2D map predictions (obtained starting from the same map) to estimate the accuracy of the prediction. A stopping criterion that leverages on 2D map prediction should thus also take into account such an estimate. Additionally, defining an appropriate threshold for stopping exploration based on the difference between the predicted and actual map presents another challenge. These factors make the use of the map-predictive methods for defining stopping criteria an interesting but complex problem, extending far beyond the scope of this paper. We will investigate these insights as future work.

5.2 Map classification performance

We indicate as True Explored (True Not-Explored), or TE (TN), the percentage of test maps correctly classified as `explored` (`not-explored`), and as False Explored (False Not-Explored), or FE (FN), the classification errors. We report accuracy (A), precision (P), and recall (R) in Table 1 for both batch and online evaluation.

Table 1 shows that in our batch-wise assessment (B) with $\theta = 0.5$, we achieve an accuracy A = 92.9%, with only 1.76% of false positives FE. Minimising FE is particularly important as a map wrongly considered as `explored` (FEs) can result in loss of relevant portions of the map. In our experiments, a false positive FE has a minimal impact on the performance metric; a map incorrectly labelled as `explored` covers, on average, 2.51% less area than the first truly `explored` map obtained in the same run. FNs are less harmful, as considering not finished the exploration of an actually explored map causes only time delays.

Increasing θ to 0.8 further reduces the FE errors, at the cost of slightly reducing the value of TE. Manual inspection of the 21 FEs with $\theta = 0.8$ reveals that they occur in 15 out of 613 runs, across 7 of the 15 environments. These FEs are due to the environments' topology combined with the robot exploration strategy: a single room is missing from M_t, with no LiDAR beams entering through its doorway, while the other room connected to such a doorway is fully explored. An example of such error is shown in Fig. 5. While FEs cannot be entirely avoided, they could potentially be mitigated using LiDAR-based (e.g., Akai (2023)) or vision-based (e.g., Antonazzi et al. (2024)) door detection methods.

Table 2 Saved time with our stopping criterion and error in predicting the percentage of explored area for each environment of the batch (B, left) and online (O, right) evaluation

envs	# runs	$\Delta\hat{t}$ [%]	err_A [%]
b_1	41	43.54	4.57
b_2	36	19.57	4.40
b_3	43	41.23	5.04
b_4	44	29.09	3.70
b_5	39	48.71	3.60
b_6	30	24.16	5.90
b_7	40	24.40	4.40
b_8	47	31.48	4.50
b_9	39	50.00	5.30
b_{10}	46	14.40	7.03
b_{11}	52	21.20	7.80
b_{12}	36	27.00	2.20
b_{13}	34	23.77	5.40
b_{14}	56	48.21	2.20
b_{15}	30	27.63	4.10
All	613	31.62	4.68

envs	# runs	$\Delta\hat{t}$)[%]
o_1	5	29.8
o_2	5	59
o_3	5	40.69
o_4	5	31.3
o_5	5	37
o_6	5	34
o_7	5	36
o_8	5	26.88
o_9	5	36.83
o_{10}	5	41.17
all	50	37.37

5.3 Stopping criterion evaluation

To assess the effectiveness of our stopping criterion (with $\theta = 0.5$), which stops the exploration when a partial map M_t is labeled as `explored` by the network $\mathcal{C}$, we compute the percentage of the time saved compared to an exploration ending with the baseline stopping criterion, $\Delta\hat{t} = |T - \hat{t}|/T$,

as well as the average of $\Delta\hat{t}$ across all runs in all environments. Table 2 (left) shows the results of the batch evaluation, averaged across all runs for each test environment. Our stopping criterion can reduce the total exploration time (against the baseline) by 31.62%, on average. Note how this value is close to that of an ideal stopping criterion (defined in Sect. 3), which for our batch evaluation test environments has an average time saved of $\Delta\bar{t} = |T - \bar{t}|/T = 33.46\%$. As a consequence, err_t of Sect. 3 is small, under 2%.

While already promising, we emphasize that the batch evaluation reported in Table 2 (left) provides a rough estimation of the actual performance of our stopping criterion, as the test set for `B` contains maps that are saved every 10 min during an exploration run. Indeed, during a run, it is likely that the stopping criterion could ideally be triggered at a time that lies within the 10-minute interval.

Results on the online settings, where maps are saved every 30 s during each run, provide a more realistic evaluation of the speed up enabled by our stopping criterion, which saves on average 37.37% of the exploration time, as reported in Table 2 (right). This significant decrease in the exploration time comes at the small cost of stopping the exploration when less than 1% of the total area of the environment is still to be mapped. Remarkably, our stopping criterion is triggered in all runs at the same time as the ideal stopping criterion $\bar{t}$, thus having a $err_t = |\bar{t} - \hat{t}|/T = 0$. A more conservative $\theta = 0.8$ result in $\Delta\hat{t} = 35\%$ and $err_t = 1.92\%$.

In general, we could say that our stopping criterion terminates the exploration earlier than the baseline, when almost all parts of the environment have been mapped. Put it differently, our approach is able to estimate when a map represents an environment almost completely. Thus, when the exploration process is stopped by our method, the map is not complete and one could explore more, but the areas that could be further explored have little relevance. In this sense, our stopping criterion should be employed in applications where a trade-off between time and map completeness is important.

5.4 Map completion percentage prediction

The rightmost column of Table 2 (left) reports the average error $err_A = |A_t - \hat{A}_t|$ in estimating the percentage of explored area in the map M_t. Overall, we observe an average value of $err_A = 4.68\%$, which is low and stable across all the 15 test environments. To enable a fair comparison, we have assessed the map completion percentage prediction only on the data collected with the batch evaluation as, during the late stages of the online evaluation, we typically have multiple almost-complete maps whose percentage of explored area is almost perfectly estimated by our method, thus overestimating its performance.

Fig. 4 shows how the average err_A progresses during an exploration run. While at the beginning of the exploration the error is higher (as the map known to the robot is too small to make an accurate estimation), err_A becomes stable and under 10% when half or more of the environment is

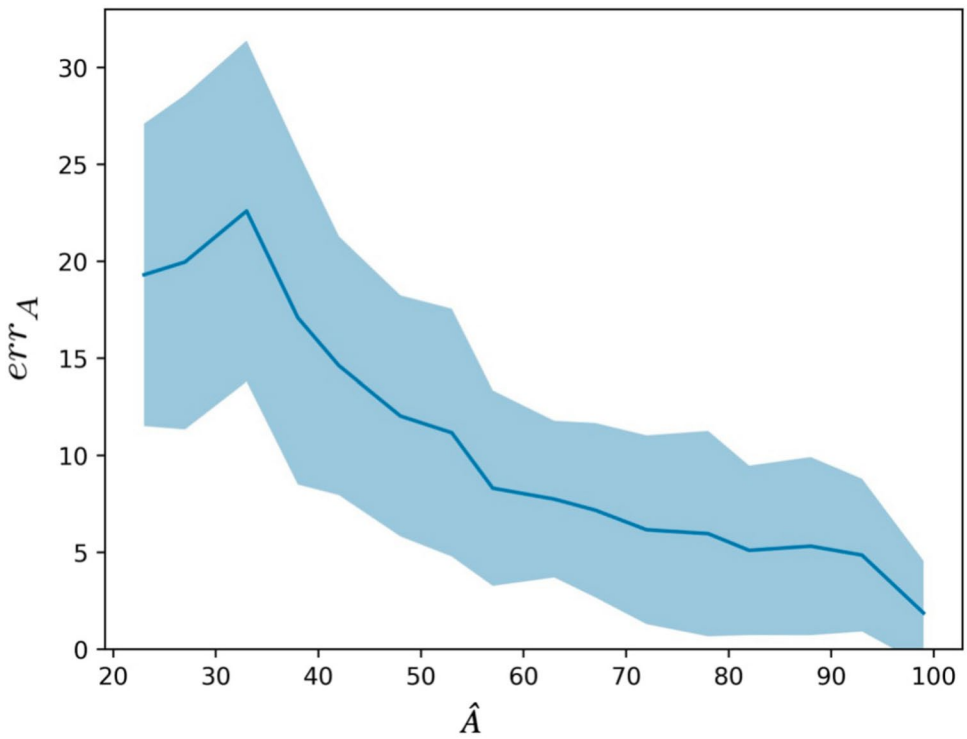
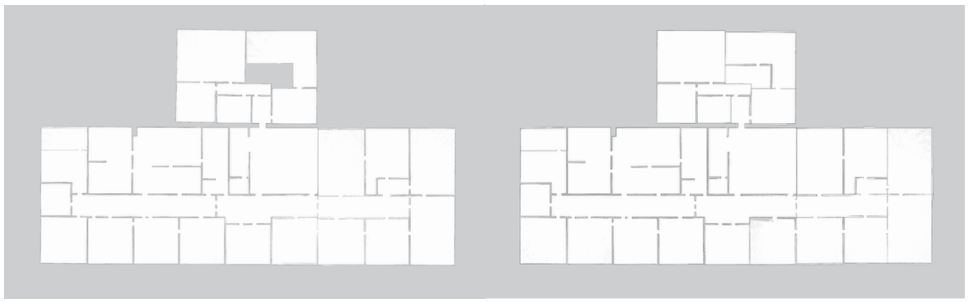


Fig. 4 The error in predicting the percentage of the explored area (err_A) as exploration progresses and the percentage of the area increases. The standard deviation is in light blue



(a) FE map. (b) TE map.

Fig. 5 An example of a map wrongly labeled as `explored` by our method (**a**), even if a full room is missing, and the complete map of the same environment from another exploration run (**b**)

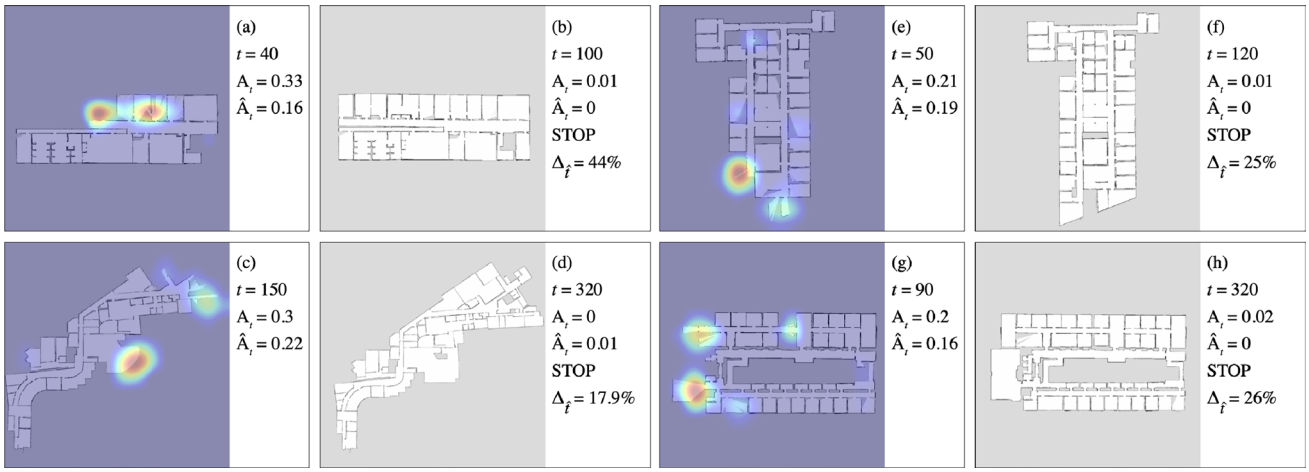


Fig. 6 Examples of maps classified as `not-explored` and `explored` (those denoted with STOP) from the experiments described in Sect. 5.5

explored. Overall, our method can provide a reliable estimate of the percentage of the explored area represented in a map.

5.5 Maps obtained with a different exploration strategy

As the exploration strategy used by the robot can influence the visual content of partial maps, in this section we evaluate the generalization abilities of our method on maps obtained with different exploration strategies. Maps obtained with an exploration strategy that selects the closest frontier (Placed & Castellanos, 2022) often have *few* small gaps, since the robot fully explores its surroundings before moving on to unexplored areas. Differently, an exploration strategy that uses map-prediction methods to estimate information gain quickly explores large portions of the environments, leaving behind several gaps (Ericson et al., 2021, 2025). The maps we used so far are obtained using an exploration strategy that balances information gain and distance (see Sect. 4.2).

We repeat our experiments on a new dataset D_{CF} of 7701 maps (4702/2099 `explored/not-explored`) of 267 large-size environments from MIT and KTH. These maps are obtained with a method similar to that of Sect. 4.2 and saved as in the batch-wise evaluation, but with two significant differences. First, we use a closest-frontier (CF) exploration strategy. Second, we employ a more conservative baseline stopping criterion during data collection: we end the exploration at a time T either when there are no frontiers left or when all frontiers are unreachable by the robot. A frontier is unreachable when there are no available paths to it. We then test the maps of D_{CF} with the model trained on D_{IG}. No environment whose maps are used for training in D_{IG} are used in D_{CF}, so there is not any dataset leakage.

Despite the visual differences in maps of these two datasets, our method is able to generalize to these examples well. We obtain, using a threshold $\theta = 0.5$, a minor drop in performance: $\mathrm{A} = 0.85$, $\mathrm{P} = 0.88$, and $\mathrm{R} = 0.72$. The lower recall is a good compromise, and we match our desiderata of having a robust stopping criterion to false positives (293 FE over 4702 examples). With $\theta = 0.8$, $\mathrm{A} = 0.81$ and $\mathrm{R} = 0.56$ decrease, but $\mathrm{P} = 0.92$ increases and the FEs are halved to 153 (less than 2%). Our stopping criterion can reduce the exploration time of $\Delta\hat{t} = 36.22\%$, a value close to the one of the ideal stopping criterion $\bar{t}$, $\Delta\bar{t} = 42.33\%$. Despite the domain shift, the error in predicting the percentage of explored area is also good, with an average $err_A = 11.3\%$. Some examples of maps obtained in such a way and processed by our method is shown in Fig. 6. Full results and examples are available in our repository.

5.6 Qualitative evaluation

Figure 3 shows our method applied to maps that are incrementally obtained during exploration runs from the batch evaluation `B`. Fig. 3a–c and e–g show how the saliency maps, computed using Grad-CAM from $\mathcal{C}$, can successfully identify relevant areas to be explored. Fig. 3d and h show the maps when the stopping criterion is triggered. The maps are correctly considered complete 30 and 50 min before the actual end of the exploration runs when using the baseline stopping criterion, thus saving $\approx 40\%$ of the time. Also, our method can robustly predict the percentage of the explored area, increasing the accuracy of the estimate as the exploration progresses. These results are confirmed by the examples in Fig. 6, which shows 8 maps, from 4 different environments processed as described in Sect. 5.5. Also in this case, even if the exploration strategy used to explore the environments is different from that used for acquiring the data on which our model is trained, our method can reliably estimate the status of the maps as well as the percentage of explored area.

5.7 Evaluation with different feature extraction backbones

In this section, we evaluate different backbones as feature extractors, and we compare their results with those obtained with our reference model, EfficientNet B1 (Tan & Le, 2019). We select three different architectures, a ResNet (ResNet18) (He et al., 2016) and two vision transformers, ViT (Dosovitskiy et al., 2021) and Swin Transformer (SwinV2-T) (Liu et al., 2022). For ViT we also test two configurations – tiny (ViT-T) and base (Vit-B) – to evaluate the performance difference across models of different sizes. All the considered models consist of a pre-trained backbone and a head MLP, with a small number of fully connected layers (i.e., 1 or 2 layers). Different hyperparameters are used for different models and are tuned using the Tree-Structured Parzen Estimator from the Optuna suite (Akiba et al., 2019).

For this comparison, we use the same dataset used for the batch evaluation `B`. Each experiment is conducted on the same training-validation-test split to ensure comparable results. The data augmentation transformations are kept identical across different architectures. To fine-tune the backbones on our downstream tasks, we unfreeze during training both the backbone and the head with different learning rates, using AdamW (Loshchilov & Hutter, 2017). The implementation of the backbones, as well as the results, are available in our repository.

Table 3 reports the results when different backbones are used both for the task (*a*), map classification as (`not`)-`explored`, and for the task (*b*), map completion percentage prediction. We also report the size in terms of parameters for each model, and we use different threshold values θ as done in Sect. 5.2. For the map classification tasks, the EfficientNet model combines both a low number of parameters and a high performance. Transformer-based methods have slightly superior performance for the task of map completion percentage prediction, while also having higher complexity. Overall, the performance of all models is similar. We thus conclude that the choice of EfficientNet B1 as a backbone is a good compromise between computational requirements and performance.

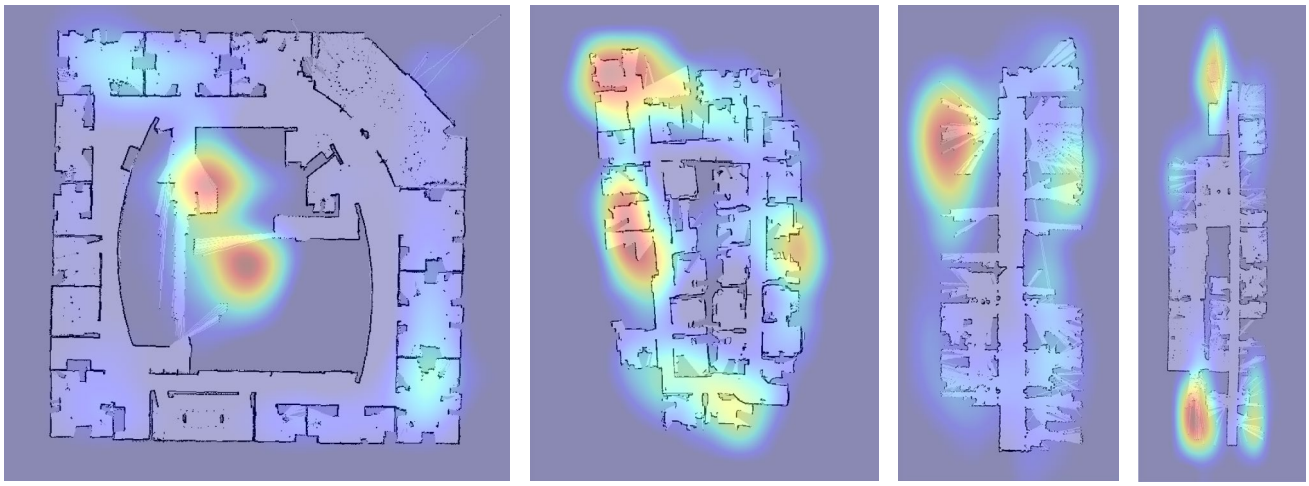
5.8 Real-world maps

As our method uses simulated and empty maps for training, we run the following test to assess how it generalizes to real-world maps acquired in cluttered environments. More precisely, we test our approach on 64 maps of real-world environments. 54 maps are obtained during the incremental mapping process of 3 real-world environments, as recorded in the publicly-available dataset of Howard and Roy (2003). The remaining 10 maps are obtained during the frontier-based exploration process of a large-scale

Table 3 Results in both tasks (*a*) and (*b*) using different network architectures for the maps in batch evaluation `B`

Method	Params	θ	A	P	R	err_A [%]
EfficientNet B1 (Tan , Le, 2019)	7.8M	0.5	**0.92**	**0.97**	**0.91**	4.68
		0.8	**0.91**	**0.99**	**0.84**	
ResNet18 (He et al., 2016)	11.2M	0.5	0.91	**0.97**	0.87	3.72
		0.8	0.87	**0.99**	0.77	
ViT-T (Dosovitskiy et al., 2021)	5.6M	0.5	**0.92**	0.96	0.88	4.32
		0.8	0.85	0.98	0.76	
ViT-B (Dosovitskiy et al., 2021)	86.4M	0.5	0.86	0.94	0.80	3.82
		0.8	0.85	0.97	0.75	
SwinV2-T (Liu et al., 2022)	27.6M	0.5	**0.92**	0.94	0.90	**3.41**
		0.8	0.88	0.97	0.81	

Params is the number of parameters of each model, θ is defined as in Sect. 4.1, while A, P, and R are accuracy, precision, and recall, respectively, as in Sect. 5.2. Finally, err_A is defined as in Sect. 5.4. We highlight in bold the best performance for each metric



(a) $\hat{A} = 0.22$ (b) $\hat{A} = 0.27$ (c) $\hat{A} = 0.48$ (d) $\hat{A} = 0.40$

Fig. 7 Examples of real-world maps, all labeled as `not-explored`. Our method can highlight the parts relevant to explore as well as predict the explored area $\hat{A}$

real-world environment from Chang et al. (2017). We label all the 64 test maps as `not-explored` by a visual inspection, because in all these maps some relevant portions of the environments are not yet mapped by the robot.

We remark that these maps are significantly different from those used to train our model, which are obtained in empty environments without clutter. Despite the domain shift, our model shows robust performance by correctly classifying all the maps as `not-explored`. Fig. 7 shows Grad-CAMs for four examples, highlighting how in all cases our model can accurately identify what is still left to explore, while also estimating the percentage of explored area $\hat{A}$. Since we do not know the actual area of these environments, we cannot compute err_A.

6 Conclusion

In this work, we have presented an approach to estimate the completeness of partial maps that increases the awareness of an autonomous mobile robot during the process of exploration for map building of unknown environments. Given the partial grid map of an environment, our method reliably estimates the percentage of explored area with an error lower than 5% and predicts if the exploration run should be stopped because the map contains all the relevant knowledge about the environment (or not), reducing the exploration time up to 35%.

Future work involves using the output of our CNNs to inform exploration strategies, improving the quality of the resulting map in tasks like Active SLAM, and using the estimate $\hat{A}$ to support the exploration process, e.g., by switching between different strategies (Amigoni et al., 2017). In general, our CNNs can be used to define stopping criteria that embed different desiderata of the exploration process.

Acknowledgements The authors would like to thank Valerii Stakanov for his contribution in the technical implementation of the experimental evaluation of our method.

Author contributions M.L. and F.A. wrote the main manuscript, M.L, G.B, and F.A. reviewed the manuscript text, all authors contributed to the conceptualisation of the work, M.M.F and M.L. implemented the method and ran the experimental evaluation, as well as prepared the figures. M.P. collected the dataset used for the second set of experiments, validated the data, and performed and evaluated new experiments testing different backbones, and revised the paper.

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare no Conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Akai, N. (2023). Slamer: Simultaneous localization and map-assisted environment recognition. In *Proc. ICRA* (pp. 6203–6209).

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proc. KDD* (pp. 2623–2631)

Amigoni, F., Banfi, J., Longoni, A., & Luperto, M. (2017). Online switch of communication modalities for efficient multirobot exploration. In *Proc. ECMR* (pp. 1–6).

Amigoni, F., Castelli, V., & Luperto, M. (2018). Improving repeatability of experiments by automatic evaluation of slam algorithms. In *Proc. IROS* (pp. 7237–7243).

Amigoni, F., Fusi, D., Luperto, M., & Borghese, A. (2019). Robot exploration using knowledge of inaccurate floor plans. In *Proc. ECMR*.

Amigoni, F., Quattrini Li, A., & Holz, D. (2013). Evaluating the impact of perception and decision timing on autonomous robotic exploration. In *Proc. ECMR* (pp. 68–73).

Amigoni, F., Banfi, J., & Basilico, N. (2017). Multirobot exploration of communication-restricted environments: A survey. *IEEE Intelligent Systems, 32*(6), 48–57.

Antonazzi, M., Luperto, M., Borghese, N.A., & Basilico, N. (2024). Development and adaptation of robotic vision in the real-world: the challenge of door detection. arXiv preprint arXiv:2401.17996.

Baek, S., Moon, B., Kim, S., Cao, M., Ho, C., & Scherer, S., et al. (2025). Pipe planner: Pathwise information gain with map predictions for indoor robot exploration. arXiv preprint arXiv:2503.07504.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., & Siegwart, R. (2018). Receding horizon path planning for 3d exploration and surface inspection. *Autonomous Robots, 42*, 291–306.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., & Leonard, J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics, 32*(6), 1309–1332.

Chang, A., Dai, A., Funkhouser, T., et al. (2017). Matterport3D: Learning from RGB-D data in indoor environments. arXiv preprint arXiv:1709.06158.

Chang, A., Dai, A., Funkhouser, T., et al. (2017). Matterport3D: Learning from RGB-D data in indoor environments. arXiv preprint arXiv:1709.06158.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. ICLR*.

Ericson, L., & Jensfelt, P.(2024). Beyond the frontier: Predicting unseen walls from occupancy grids by learning from floor plans. In *IEEE RAL*.

Ericson, L., Duberg, D., & Jensfelt, P. (2021). Understanding greediness in map-predictive exploration planning. In *Proc. ECMR* (pp. 1–7).

Ericson, L., Pedro, J., & Jensfelt, P. (2025). Information gain is not all you need. arXiv preprint arXiv:2504.01980.

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. KDD* (pp. 226–231).

Georgakis, G., Bucher, B., Arapin, A., Schmeckpeper, K., Matni, N., & Daniilidis, K. (2022). Uncertainty-driven planner for exploration and navigation. In *Proc. ICRA* (pp. 11295–11302).

Ghaffari Jadidi, M., Valls Miro, J., & Dissanayake, G. (2019). Sampling-based incremental information gathering with applications to robotic exploration and environmental monitoring. *International Journal of Robotics Research, 38*(6), 658–685.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proc. CVPR* (pp. 770–778).

Ho, C., Kim, S., & Moon, B., et al. (2025). Mapex: Indoor structure exploration with probabilistic information gain from global map predictions. In *Proc. ICRA*.

Howard, A., & Roy, N. (2003). The Robotics Data Set Repository (Radish) http://radish.sourceforge.net/.

Katsumata, Y., Kanechika, A., Taniguchi, A., El Hafi, L., Hagiwara, Y., & Taniguchi, T. (2022). Map completion from partial observation using the global structure of multiple environmental maps. *Advanced Robotics, 36*(5–6), 279–290.

Kingma, D.P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Leung, C., Huang, S., & Dissanayake, G. (2008) Active slam in structured environments. In *Proc ICRA* (pp. 1898–1903).

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., Wei, F., & Guo, B. (2022). Swin transformer v2: Scaling up capacity and resolution. In *Proc. CVPR* (pp. 12009–12019).

Liu, Y., & Nejat, G. (2013). Robotic urban search and rescue: A survey from the control perspective. *Journal of Intelligent and Robotic Systems, 72*(2), 147–165.

Lluvia, I., Lazkano, E., & Ansuategi, A. (2021). Active mapping and robot exploration: A survey. *Sensors, 21*(7), 2445.

Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101.

Luperto, M., Antonazzi, M., Amigoni, F., & Borghese, N. A. (2020). Robot exploration of indoor environments using incomplete and inaccurate prior knowledge. *Robotics and Autonomous Systems, 133*, Article 103662.

Luperto, M., Castelli, V., & Amigoni, F. (2021). Predicting performance of slam algorithms. arXiv preprint arXiv:2109.02329.

Luperto, M., Fochetta, L., & Amigoni, F. (2021). Exploration of indoor environments through predicting the layout of partially observed rooms. In *Proc. AAMAS* (pp. 836–843).

Placed, J., & Castellanos, J. (2022). Enough is enough: Towards autonomous uncertainty-driven stopping criteria. *Proc. IAV* pp. 126–132.

Placed, J., Strader, J., Carrillo, H., Atanasov, N., Indelman, V., Carlone, L., & Castellanos, J. A. (2023). A survey on active simultaneous localization and mapping: State of the art and new frontiers. *IEEE Transactions on Robotics, 39*(3), 1686–1705.

Selvaraju, R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proc. ICCV* (pp. 618–626).

Shrestha, R., Tian, F.-P., Feng, W., Tan, P., & Vaughan, R. (2019). Learned map prediction for enhanced mobile robot exploration. In *Proc. ICRA* (pp. 1197–1204).

Song, K., Chen, G., Tomizuka, M., Zhan, W., Xiong, Z., & Ding, M. (2025). P2 explore: Efficient exploration in unknown cluttered environment with floor plan prediction. In *Proc. IROS*.

Stachniss, C., & Burgard, W. (2003). Exploring unknown environments with mobile robots using coverage maps. In *Proc. IJCAI* (pp. 1127–1134).

Stachniss, C., & Burgard, W. (2005) Mobile robot mapping and localization in non-static environments. In *Proc. AAAI* (pp. 1324–1329).

Stachniss, C. (2009). *Robotic Mapping and Exploration* (Vol. 55). Berlin, Heidelberg: Springer.

Tan, M., & Le, Q. (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proc. ICML* (pp. 6105–6114).

Tao, Y., & Iceland, E. et al. (2024). Learning to explore indoor environments using autonomous micro aerial vehicles. In *Proc. ICRA IEEE*.

Thrun, S. (2002). Robotic mapping: A survey. In *Exploring Artificial Intelligence in the New Millenium* (pp. 1–35).
Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *Proc. CIRA* (pp. 146–151).
Yamauchi, B., Schultz, A., & Adams, W. (1999). Integrating exploration and localization for mobile robots. *Adaptive Behavior, 7*(2), 217–229.

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



Matteo Luperto received the Ph.D. degree in information technology from the Politecnico di Milano, Milano, Italy, in 2017. He is currently an Assistant Professor with the Dipartimento di Informatica Giovanni Degli Antoni, Università degli Studi di Milano, Milano. His main research interests lie within the field of semantic mapping for autonomous mobile robots in indoor environments, with a particular attention on the analysis of the structural properties of buildings, and long-term autonomy for social assistive mobile robots.



Marco Maria Ferrara earned his Master's degree in Computer Science from Politecnico di Milano, Italy, in 2022. His research interests include artificial intelligence and robotics.



Matteo Princisgh received his B.Sc. degree in Computer Science from the Università degli Studi di Milano, Italy, in 2023. He is currently pursuing his M.Sc. in Computer Science at the same institution. His research interests include robotic perception and autonomous navigation in indoor environments.



Giacomo Boracchi is Associate Professor of Computer Engineering at Dipartimento di Elettronica, Informazione e Bioingegneria of the Politecnico di Milano (DEIB). He received a Ph.D. in information technology (DEIB, 2008) and an MSc in Mathematics (Universitá Statale di Milano, 2004). His primary research interests concern image processing and machine learning, particularly image restoration, image analysis, change/anomaly detection, domain adaptation, and learning in nonstationary environments. He leads multiple research projects applying imaging algorithms to design new industrial solutions. He authored over 120 papers in international conferences and journals and served as Associate Editor for the IEEE Transactions on Image Processing. In 2015, he received the IBM Faculty Award; in 2016, the IEEE Transactions on Neural Networks and Learning Systems Outstanding Paper Award; in 2017, the Nokia Visiting Professor Scholarship; in 2021, an Nvidia Applied Research Grant.



Francesco Amigoni was born in Soresina (CR), Italy, on December 2, 1971. He got the Laurea degree in Computer Engineering from the Politecnico di Milano in 1996 and the PhD degree in Computer Engineering and Automatica from the Politecnico di Milano in 2000. From December 1999 to September 2000, he was a visiting scholar at the Computer Science Department of Stanford University (USA). He has been an assistant professor (2002-2007) and an associate professor (2007-2021) and, since 2021, a full professor at the Dipartimento di Elettronica e Informazione of the Politecnico di Milano. His main research interests include agents and multiagent systems, autonomous mobile robotics, and the philosophical aspects of artificial intelligence.