This paper has been accepted for publication at the International Symposium on Robotics Research (ISRR), Hanoi, 2019.

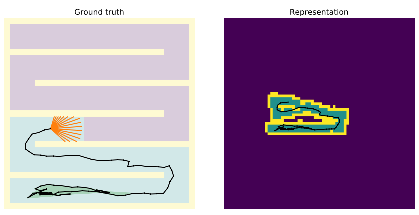
# Exploration Without Global Consistency Using Local Volume Consolidation

Titus Cieslewski, Andreas Ziegler, and Davide Scaramuzza

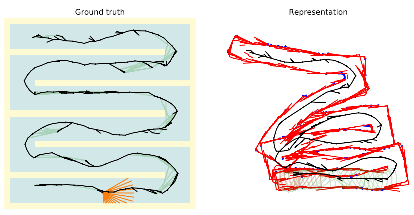
Robotics and Perception Group, Depts. of Informatics and Neuroinformatics, University of Zurich and ETH Zurich
`http://rpg.ifi.uzh.ch`

**Abstract.** In exploration, the goal is to build a map of an unknown environment. Most state-of-the-art approaches use map representations that require drift-free state estimates to function properly. Real-world state estimators, however, exhibit drift. In this paper, we present a 2D map representation for exploration that is robust to drift. Rather than a global map, it uses local metric volumes connected by relative pose estimates. This pose-graph does not need to be globally consistent. Overlaps between the volumes are resolved locally, rather than on the faulty estimate of space. We demonstrate our representation with a frontier-based exploration approach, evaluate it under different conditions and compare it with a commonly-used grid-based representation. We show that, at the cost of longer exploration time, using the proposed representation allows full coverage of space even for very large drift in the state estimate, contrary to the grid-based representation. The system is validated in a real world experiment and we discuss its extension to 3D.

**Video:** A video is available at `https://youtu.be/s4Xnet_h4ss`



(a) Global, grid-based



(b) Ours

Fig. 1: In a maze-like environment where loop closures cannot be established to account for pose estimate drift, grid-based representations build an inconsistent map, which can lead to premature termination of exploration (a). The purple region in the ground truth of explored space, in the leftmost image, indicates unexplored space. Our method, in contrast, only resolves overlaps between volumes locally, which allows to fully explore the environment (b). See Fig. 2 for a detailed legend.

## 1 Introduction

Exploration is a fundamental task in autonomous robotics. The goal is to map an unknown environment [30], for example to establish how a robot can navigate through it. Exploration can be used in various scenarios from search and rescue, oil and gas exploration, 3D reconstruction to inspection tasks. Different applications have different requirements. In search and rescue, for instance, it is important to find survivors rapidly. Other requirements could be that the generated map has a high accuracy or that the path travelled by the robot is as short as possible. In exploration, a map representation is used to describe the space that the robot perceived so far. Based on this description, a planning algorithm decides where to move next. In this work, we focus on the map representation.

Most map representations currently used for exploration assume perfect state estimates. However, on-board state estimators, such as visual-inertial odometry, are prone to drift. Since these are the estimators most likely to be used in unknown environments in the real world, these map representations can in practice lead to incorrect maps, as illustrated in Fig. 1.

To some extent, drift can be removed when a robot revisits a place and a correction of the trajectory estimate can be performed with a loop closure. However, as Fig. 1 illustrates, not all drift error can be removed in this way. Furthermore, such a correction causes heavy computational load in typical grid-based maps, as every depth measurement ray needs to be re-cast with the new trajectory estimate [29]. Sub-mapping approaches can mitigate this computational cost [4], [25], [20].

### 1.1 Contributions

In this paper, we propose a map representation for exploration that is robust to state estimate drift. We build our approach using ideas from [16], where free space is represented with local polygons connected by relative poses. Our contributions are as follows:

- We apply the ideas from [16] to exploration by adding semantics to the local polygons, which incorporate information from other nearby polygons. Nearby polygons are determined based on vicinity in a pose graph, as opposed to vicinity in an estimate of space that is faulty due to drift.
- We show that with this, global map consistency is not required for the robot to know when exploration is complete. With this, global map optimization is not needed, which is particularly interesting for multi-robot exploration.
- We show how to apply our representation to an exploration algorithm that previously used a grid-based representation. Local navigation is performed in local polygons, while navigation between remote locations in the map is performed using teach-and-repeat [8].
- Finally, we simulate our proposed map representation under different conditions, compare it to grid-based representations, and validate it in the real world. As we show, our method comes at a cost of longer exploration times,

but given reliable place recognition, is able to deal with large drift in cases where grid-based representations fail.

## 2 Related Work

While a map representation is used to represent the space that the robot perceived so far, a planning algorithm has to plan where the robot should move next. In the following, we review these two aspects of exploration separately.

### 2.1 Map Representations

Exploration can be formalized as the problem of discovering all free space within a bounded volume. Thus, the map representation needs to represent the spatial information necessary to achieve this. Map representations are commonly divided into three categories [28]: metric representations, topological representations and hybrid, topometric representations.

**Metric Representations** Metric representations express locations as coordinates in a global reference frame. The predominant type of metric representations used in exploration are grid-based representations. In a grid-based representation, the volume is quantized into voxels, arranged in a grid. The voxels are then assigned appropriate labels, such as “free”, “unknown” or “obstacle” [30]. Voxels typically carry further information, such as occupancy probabilities [21, 29] or a signed distance to the closest obstacle [17, 22]. In two dimensions, an alternative way of representing known free space is to represent it with a polygon [5,12]. Here, the inside of the polygon represents free space, while its boundaries can either represent obstacles or the interface to unknown space. Because the location of all entities in fully metric representations are expressed in a global reference frame, they are highly reliant on accurate pose estimates: wrong pose estimates would lead to depth measurements being inconsistent with the true scene geometry, and with each other. As a consequence, the robot could wrongly believe occupied space to be free or inversely, free space to be occupied, which could have disastrous consequences for navigation.

**Topological Representations** Unlike metric representations, topological representations do not express locations as coordinates in a reference frame. Instead, locations are represented as vertices in a graph. Relationships between locations, such as adjacency, are expressed as edges. In order to be navigable, these edges should carry enough information to allow a robot to move between adjacent vertices. Vision-based navigation in such maps has been demonstrated using Visual Teach and Repeat [8,10]. While topological maps can also be derived from volumetric maps [3, 28], we are not aware of any work that uses purely topological maps to directly represent volumetric information. Pure topological maps are thus rarely used for exploration. A notable exception to this are [1,4]. Rather than explicitly keeping track of free space, their robots build a pose graph similar to [8]. At every vertex of this graph, the robot adds to the graph a finite amount of adjacent candidate locations where the robot could move to.

Subsequently, these candidate locations are all visited and the process is repeated recursively unless a candidate turns out to correspond to a previously visited location. However, candidate robot locations are only a very crude representation of free space, which makes it hard to estimate the actual coverage.

**Hybrid Representations** Hybrid, or *topometric*, representations are a combination of metric and topological representations, aimed at combining their advantages. Strictly speaking, the aforementioned pose graphs were already topometric, since they contained metric positions and transformations. Nonetheless, in this section we focus on representations which augment topological graphs with volumetric information. An early topometric representation has been proposed in [16]. This representation consists of a set of polygons, each representing the free space measured around a specific pose of the robot. The locations of these polygons are expressed relative to each other. However, [16] does not consolidate the information of neighbouring polygons in a way that would be helpful for exploration. As a consequence, in [15], where [16] is used in an exploration system, the authors fall back on local occupancy grids for exploration. In contrast, we extend [16] with a consolidation procedure that allows its direct application to exploration. Furthermore, while [15] ends up building a globally consistent map, we show that global consistency is not required for exploration. The use of local occupancy grid *submaps* is also proposed in [20, 25], where the authors aim at producing a globally consistent map of the environment just like [15]. Achieving global consistency requires map optimization, which in turn leads to costly re-calculation of the occupancy grid submaps, even if it can sometimes be avoided for small loop closure corrections [25]. We show that all of this can be avoided since global consistency is not necessary for exploration.

## 2.2 Path Planning for Exploration

In order to understand how to build a good representation, we need to understand how it is used. In path planning for exploration, after every observation the robot has to decide where to move next. In order to achieve fast exploration, the robot should perceive unknown space as quickly as possible. There are two main planning approaches in the literature: Next-best-view (NBV) planning and frontier-based planning.

**Next-best-view (NBV) Planning** NBV planners try to choose the next position in such a way that the perception is optimal according to a *utility* function. This problem has also been studied in the computer vision community [7]. Most NBV planners determine the next best view by sampling candidate views, predicting their utility [12,23], and picking the view with the highest utility. The advantage is that the utility function can be adapted to contain arbitrary terms and constraints, such as the motion model of the robot, or terms describing map accuracy. The disadvantage is that finding the view with optimal utility is usually intractable, hence a sampling method has to be applied. Furthermore, and somewhat as a consequence, NBV planning commonly requires heavy computational load.

**Frontier-Based Exploration Planning** Frontier-based exploration planners follow a simple and efficient scheme. The strategy is to navigate to the closest boundary between known free and unknown space. These boundaries are referred to as *frontiers*. Frontier-based exploration methods assume that navigating to a frontier will result in the exploration of new space. The original idea was introduced in [30]. Unlike NBV planners, frontier-based planners do not natively support arbitrary terms or constraints. However, it was shown that they cover the space faster than their NBV counterparts [6, 14]. In this work, we will evaluate our representation using a modified version of the frontier-based exploration method presented in [6]. Frontiers are explicitly expressed in our method, so it is natural to adopt frontier-based exploration.

## 3 Problem Statement

The goal of exploration is to build a map of an unknown environment. Consider a bounded region of space $\mathcal{V} \in \mathbb{R}^2$ that represents the unknown environment that we would like to explore. $\mathcal{V}$ consists of free and occupied space $\mathcal{V} = \mathcal{V}_{\text{free}} \cup \mathcal{V}_{\text{occ}}$. With a robot that can measure the free space around itself, we can explore $\mathcal{V}_{\text{free}}$. We denote a robot pose in the world frame $W$ as $\mathtt{T}_{W,R}$ with orientation $\mathtt{R}_{W,R}$ and translation $t_{W,R}$. The Field of View (FOV) of the robot at pose $\mathtt{T}_{W,R}$ is denoted as $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R}) \subset \mathcal{V}_{\text{free}}$. While a robot is moving on a trajectory $\mathtt{T}_{W,R}(t)$, it will measure $\mathcal{V}_{\text{fov}}$ at consecutive sampling times $t_0, t_1, \ldots$. We denote the corresponding poses as $\mathtt{T}_{W,R_0}, \mathtt{T}_{W,R_1}, \ldots$. The space explored up to time $t_k$ (*known free space*) is the union of all the FOVs measured so far:

$$\bar{\mathcal{V}}_{\text{free}}(t_k) = \bigcup_k \mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R_k}). \tag{1}$$

We consider two goals for our representation: 1) represent the robots' estimate of $\bar{\mathcal{V}}_{\text{free}}$. 2) Allow to determine when exploration is complete, that is, when the whole free space is covered:

$$\bar{\mathcal{V}}_{\text{free}} = \mathcal{V}_{\text{free}}. \tag{2}$$

Without knowing the ground truth $\mathcal{V}_{\text{free}}$, this condition can be established given sufficient knowledge about the boundary of $\bar{\mathcal{V}}_{\text{free}}$, $\partial\bar{\mathcal{V}}_{\text{free}}$. Generally, $\partial\bar{\mathcal{V}}_{\text{free}}$ either consists of interfaces to occupied space or interfaces to unknown free space. As can be shown by contradiction, (2) holds if and only if $\partial\bar{\mathcal{V}}_{\text{free}}$ consists only of interfaces to occupied space[1] ("no frontiers left"). Hence, the second goal can be reached by correctly representing $\partial\bar{\mathcal{V}}_{\text{free}}$.

## 4 Proposed Representation

As in [16], the proposed map representation is based on polygons. More specifically, local polygons individually represent the FOV of every pose, $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R})$.

[1] Trivial exceptions, such as disconnected free space, are omitted for brevity.

The inside of a polygon represents known free space $\bar{\mathcal{V}}_{\text{free}}$ while its edges represent the boundary $\partial\bar{\mathcal{V}}_{\text{free}}$. Unlike [16], $\partial\bar{\mathcal{V}}_{\text{free}}$ is labeled, which allows us to reach the second goal of the problem statement. The labels are *obstacle* for parts of the boundary that are common with occupied space, *frontier* for parts adjacent to unknown space, and *free* for parts that are considered to lie within another polygon. See Fig. 2 for a visualization of our representation.

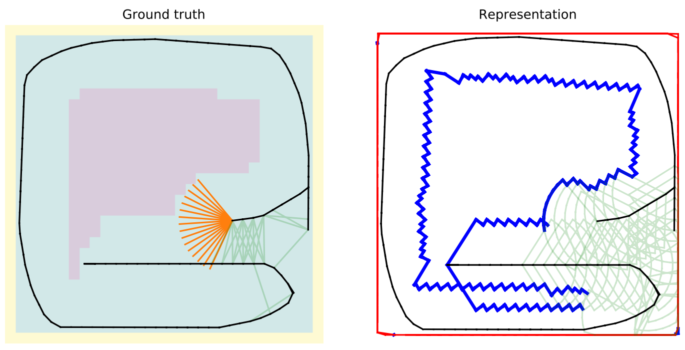


Fig. 2: Our representation, visualized for a simulation without drift in the pose estimate. Left image (ground truth): black line: robot trajectory, green lines: place recognitions, orange lines: depth sensor rays at current pose; in the background, yellow squares indicate true occupied space, green squares true known free space and blue squares true unknown free space. Right image (our representation): black line: pose graph without loop closures, red lines: known obstacles, blue lines: frontiers, green lines: local volumes in consolidation scope (see Section 4.3). Note how frontiers may overlap, as they are not resolved globally but only within the consolidation scope.

## 4.1 Local Volumes from Depth Measurements

We assume a depth sensor as source to build the polygon of the robot's Field of View (FOV). Each measurement consists of a set of simultaneously acquired depth samples. In our simulations, these samples are modelled as equally distributed within the FOV, as shown in Fig. 3. We build the polygon from these samples by taking the position of adjacent samples and the robot's position as vertices and connecting them with edges. Some samples correspond to measured depths while others correspond to samples where the closest obstacle is beyond the sensor range. If two adjacent depth samples are not beyond range, and have values with a difference below a threshold $\delta$, the edge connecting them is considered an obstacle edge. In all other cases, the edge is considered a frontier edge. The threshold $\delta$ is intended to account for occlusions (see Fig. 3). Sometimes, these cannot be distinguished from obstacle surfaces that are nearly parallel to the measurement rays [12]. These incorrectly labeled frontiers can be resolved by approaching that obstacle from another angle. Furthermore, the two edges adjacent to the robot position are also considered frontiers.

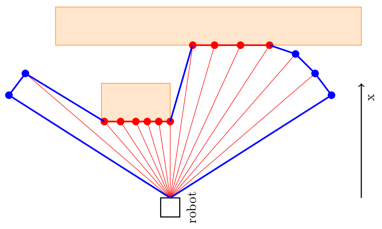


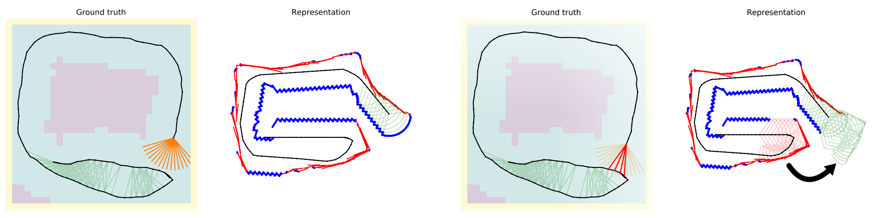
Fig. 3: A local volume obtained from one measurement. The red lines starting at the robot position represent the depth measurement rays. Blue points represent samples at maximum sensor range and red points represent samples that hit an obstacle. Blue edges are the frontier edges and red edges obstacle edges.

## 4.2 Pose Graph Representation

In our representation, we store separate local volumes for each set of depth measurement samples, which are built as described in the previous section. These polygons are referenced in the vertices of a *pose graph*. Each vertex of this graph represents a robot pose at which a measurement was taken. The vertices are connected with edges that carry the relative transformation $\mathtt{T}_{R_{k-1},R_k}$ between the two poses. The edges are either established between subsequent measurements, in which case $\mathtt{T}_{R_{k-1},R_k}$ is estimated using odometry, or they are established due to place recognition, in which case $\mathtt{T}_{R_{k-x},R_k}$ comes from a relative pose estimation algorithm. The pose graph and our representation can be constructed incrementally and on-line. The relative transformation between non-adjacent poses can be estimated by integrating relative transformations along a path connecting the two corresponding vertices. Considering odometry drift, this estimate becomes less accurate as the path length increases. Without global consistency, estimates integrated along different paths can differ [4, 8]. Thus, for our purposes, we use the estimate resulting from integrating along the shortest path.

## 4.3 Frontier Consolidation

When adding new depth measurements, we need to update the local polygons. New measurements can turn unknown space into known free space, and the corresponding frontier boundaries need to be converted into free boundaries. Similarly, frontiers might need to be re-assessed after place recognition. This process, which we call *frontier consolidation*, is illustrated in Fig. 4. For every new depth measurement $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R_k})$, a *consolidation scope* comprising the local volumes of nearby poses is established using Dijkstra's algorithm. All poses within a distance of $\mathcal{R}$ are added to that scope, where the norm of the estimated translation $t_{A,B}$ between two adjacent poses is used as the weight of the edge between them. Consequently, $\mathcal{R}$ reflects the distance over which the pose estimation (odometry and relative pose estimation) has small drift and can be used to consolidate local volumes. Inside the consolidation scope, all local volumes are then consolidated pairwise among all possible pairs. To that end, all



(a) Before loop closure (b) After loop closure

Fig. 4: Frontier consolidation prompted by place recognition. See Fig. 2 for a detailed legend. (a) Before place recognition, the consolidation scope contains most recent poses. (b) After place recognition (red lines in ground truth), local volumes across the place recognition edges (light red) are added to the consolidation scope, temporarily transformed into the local frame of the current pose (arrow, light green volumes). This affects frontiers (blue): They frontiers above the red volumes and the "range arc" of the current pose are both resolved.

local volumes are temporarily expressed relative to $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R_k})$ by using the pose resulting from pose integration along the shortest path from $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R_k})$, see Fig. 4. Given two local volumes transformed in this way, any frontier edge of one volume that lies inside the other volume is re-labeled as free edge, and vice versa. If a frontier edge is only partially inside, it is subdivided accordingly.

Consolidation is triggered for every new depth measurement. In our experiments, we assume that previously visited places are recognized at the same time as a vertex is inserted into the polygon. Otherwise, polygon consolidations would also need to be triggered as new edges are inserted into the pose graph due to place recognition.

So while our approach does not rely on odometry accuracy, it relies on good place recognition performance. An increasing rate of false negatives could potentially be dealt with by methods that expand existing matches, or by increasing the volume in which polygons are consolidated. An increasing rate of false positives (perceptual aliasing) would be harder to deal with. However, false positives have also been shown to be catastrophic for optimization-based approaches [26] – remedies to false positives are equally applicable to both approaches.

### 4.4 Extension to 3D

While for simplicity, we focus on two dimensions in this paper, we believe that the presented approach is readily applicable in 3D. The main challenge here is the 3D representation of local volumes, and the intersection of those local volumes, where volume boundaries become surfaces, rather than edges. One could use local occupancy grids, or local coarse meshes [13, 27] for memory efficiency.

## 5 Application to Frontier-Based Exploration

To demonstrate how our map representation can be fitted to exploration approaches that previously used metric representations, we adapt the state-of-the-art method proposed in [6] to use our solution. Exploration in [6] is performed using a state machine with the following three states:

1. **Reactive:** While frontiers are present in the current FOV, navigate towards the frontier that incurs the least change to the current velocity.
2. **Deliberate:** If no frontiers are left in the FOV, but frontiers are left globally, plan a path to the closest frontier.
3. If no frontiers are left overall, exploration is considered **complete**.

Since the **reactive** state is based on the current field of view, it is trivial to adapt to our representation. In the original implementation, information was extracted from OctoMap [29] updates to determine frontier voxels in the current FOV. With our representation, frontier candidates are simply selected from the edges of the current local polygon. The **deliberate** state cannot be adapted directly. In the original implementation, a path to the closest frontier was found using a Dijkstra search across adjacent free space voxels. With our representation, adjacency of free space is only meaningful among polygons that are also close to each other in the pose graph. Hence, instead of looking for the closest frontier in 2D space, we instead look for the closest vertex in the pose graph that has a local volume with unconsolidated frontiers. We then let the robot navigate to that vertex using the teach and repeat navigation method proposed in [8], as it does not require global consistency of the map. Once the robot arrives at that vertex, it switches back to the reactive state. **Completion** in the original method is assumed once there are no frontier voxels left. In our approach, this corresponds to none of the polygons having frontier edges left.

## 6 Experiments

Using [6] as exploration algorithm, we compare our representation to a metric grid-based representation, with and without loop closure capability, in simulated 2D environments under varying conditions. The representations are compared in different environments, with simulated odometry drift of varying intensity and with simulated place recognition of varying recall.

### 6.1 Experimental Setup

We simulate a robot with a forward-looking depth sensor operating in either of the $30m \times 30m$ environments shown in Fig. 5. The depth sensor has a FOV of $115°$ in horizontal direction and a range $d_{\text{FOV}}$ of $5m$, similar to the sensor used in [6]. The simulation is performed in time steps $k \in \mathbb{N}$, for each of which the robot's true pose $\mathtt{T}_{W,R_k}$ and pose estimate $\bar{\mathtt{T}}_{W,R_k}$ are updated as follows:

$$\mathtt{T}_{W,R_k} = \mathtt{T}_{W,R_{k-1}} \bar{\mathtt{T}}_{R_{k-1},R_k} \eta_{\mathtt{T}_{R_{k-1},R_k}}, \tag{3}$$

$$\bar{\mathtt{T}}_{W,R_k} = \bar{\mathtt{T}}_{W,R_{k-1}} \bar{\mathtt{T}}_{R_{k-1},R_k}, \tag{4}$$

where $\bar{\mathtt{T}}_{R_{k-1},R_k}$ is the relative pose command output provided by the exploration planner. In the true pose update, a pose increment $\eta_{\mathtt{T}_{R_{k-1},R_k}}$ is applied

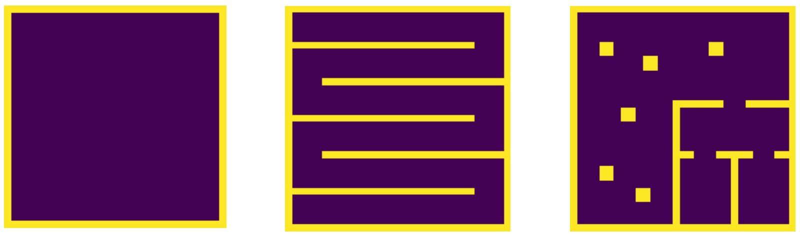


Fig. 5: The simulated environments: “Open”, “Maze” and “Forest house”. All have a size of 30 × 30 meters.

to $\bar{\mathtt{T}}_{R_{k-1},R_k}$. As it is the robot that executes the desired pose increment, it believes it has executed $\bar{\mathtt{T}}_{R_{k-1},R_k}$, and so noise needs to be applied to the true pose update, rather than the estimate. The translation of $\eta_{\mathtt{T}_{R_{k-1},R_k}}$ has coefficients sampled from a zero-mean Gaussian $\mathcal{N}(0, f(\sigma_{\text{pos}}, \bar{\mathtt{T}}_{R_{k-1},R_k}))$ and its rotation angle is sampled from $\mathcal{N}(0, f(\sigma_{\text{rot}}, \bar{\mathtt{T}}_{R_{k-1},R_k}))$. Here, $f$ is a function that ensures that the variance of the noise is proportional with the distance travelled:

$$f(\sigma, \bar{\mathtt{T}}_{R_{k-1},R_k}) = \left\|\bar{\mathbf{t}}_{R_{k-1},R_k}\right\| \cdot \sigma^2. \tag{5}$$

We furthermore simulate place recognition. Whenever the current pose $\mathtt{T}_{W,R_k}$ falls within a threshold distance $d_{\text{pr}}$ to a previous pose $\mathtt{T}_{W,R_l}$, the identifier of that pose, $l$, as well as $\mathtt{T}_{R_l,R_k}$ are provided to the simulated robot. To prevent self-matches, all poses within $1.5 \cdot d_{\text{pr}}$ of pose graph traversal are excluded from place recognition. Also, place recognition is only provided if $\mathtt{T}_{W,R_l}$ is in line of sight from $\mathtt{T}_{W,R_k}$, to prevent place recognition across occlusions.

## 6.2 Baseline Implementation

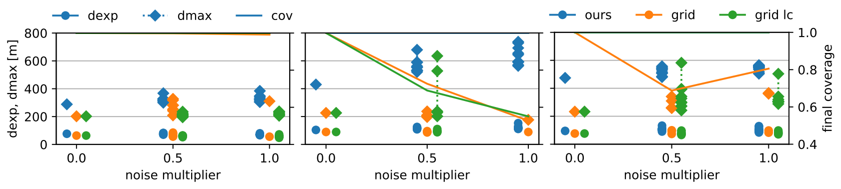
As a baseline, we implement a simplified version of the grid representation [29] using a regular grid instead of octrees, as we do not take computational performance into account. According to [21, 29], the occupation probability $P(n|z_{1:k})$ of each cell is updated using the most recent measurement $z_k$ with

$$L(n|z_{1:k}) = L(n|z_{1:k-1}) + L(n|z_k), \quad L(n) = \lim_{x \to P(n)} \log(\frac{x}{1-x}). \tag{6}$$

We assume a precise depth sensor and let $P(n|z_k) = 1$ if a ray hits an obstacle inside $n$ at time $k$, $P(n|z_k) = 0$ if cell $n$ is fully contained inside the polygon spanned by the sensor rays as defined in Section 4.1, and $P(n|z_k) = 0.5$ otherwise. Consequently, $P(n)$ can only assume values $\in \{0, 0.5, 1\}$, which we accordingly label as *free*, *unknown* and *occupied*. For cases where (6) is undefined, later measurements override earlier measurements. This baseline is implemented in two ways: a naive one (“*grid*”), which uses the state estimate $\bar{\mathtt{T}}_{W,R_k}$ as-is, and one with loop closure capability (“*grid-lc*”). The latter exploits place recognition events to optimize its pose-graph according to all relative pose estimates present in the pose graph using g2o [18]. Every place recognition triggers an optimization after which all updates according to (6) are executed anew with the optimized pose estimates $\bar{\mathtt{T}}^{\star}_{W,R_k}$. We use a voxel length of 1m for all maps.

### 6.3 Evaluation Metrics

To quantify the performance of exploration, we measure the distance traveled until the robot *believes* full coverage has been achieved, $d_{\max}$, and the expected distance to be traveled to discover an arbitrary location in free space, $d_{\exp}$. $d_{\exp}$ reflects the general speed of exploration. It is not severely affected if the robot takes a very long time at the end to navigate to the last couple of frontiers. Depending on the application, one metric is more important than the other. For more insight into the progress of exploration, we track the ratio between the volume of known free space $\bar{\mathcal{V}}_{\text{free}}$ and the full free space, $\mathcal{V}_{\text{free}}$, $|\bar{\mathcal{V}}_{\text{free}}|/|\mathcal{V}_{\text{free}}|$, which we call *coverage ratio*. It is initially 0 and reaches 1 when full coverage is achieved. Note that $\bar{\mathcal{V}}_{\text{free}}$ refers to the ground truth volume covered by the robot, not the estimate of that space by the robot. $\mathcal{V}_{\text{free}}$ is represented using a grid – in our simulations we use this grid to define the environment in the first place, see Fig. 5. Unlike in the baseline grid representation, occupied cells are known beforehand, and only free cells $n \in \mathcal{V}_{\text{free}}$ are labeled as *known free* or *unknown free*. Known free space $\bar{\mathcal{V}}_{\text{free}}$ is approximated as the set of cells labeled *known free*. An unknown free cell is relabeled *known free* as soon as any sensor ray intersects it and remains *known free* until the end of the simulation. Given this approximation of $\bar{\mathcal{V}}_{\text{free}}$, the coverage ratio is calculated by dividing the count of cells in $\bar{\mathcal{V}}_{\text{free}}$ by the count of cells in $\mathcal{V}_{\text{free}}$. Since full coverage will not be reached in all experiments, even if the robot believes this to be the case, we also report the final coverage ratio. In these cases, $d_{\exp}$ is undefined.



(a) Open (b) Maze (c) Forest house

Fig. 6: Expected distance until discovery $d_{\exp}$ and distance until termination $d_{\max}$ versus final coverage for the evaluated approaches, with different noise intensity, on different maps. For fair comparison, $d_{\exp}$ and $d_{\max}$ are omitted for samples where the final coverage is below 1. Ten samples are collected for every setting.

### 6.4 Experiments

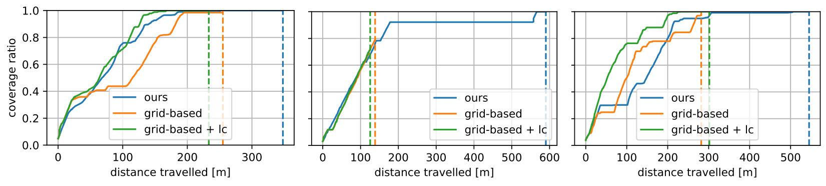
We perform experiments with the following parameter combinations: simulation in either of the environments shown in Fig. 5; pose estimate noise simulated with

$$\sigma_{\text{pos}} = \alpha \cdot 0.1m, \quad \sigma_{\text{rot}} = \alpha \cdot 5^\circ, \tag{7}$$

with the noise multiplier $\alpha \in \{0, 0.5, 1\}$; and place recognition radius $d_{\text{pr}}$ of $0.5, 1, 1.5$ or 2 times the sensor range $d_{\text{FOV}}$. For every parameter setting, ten different runs have been performed (due to the stochastic nature of the simulated noise). The place recognition radius is varied because a larger radius should result in the robot having to travel less in order to consolidate frontiers. It is limited to two times the sensor range as beyond that, polygon intersection between $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R_k})$ and $\mathcal{V}_{\text{fov}}(\mathtt{T}_{W,R_l})$ is impossible.

## 7 Results

Fig. 6 shows the performance of the evaluated methods as pose estimation noise is increased. As we can see, only the proposed representation always reaches full coverage in all environments. The other approaches perform particularly poorly in the maze environment, where the robot has to travel long distances in close proximity without the ability to close loops, see Fig. 1. We can also see that this comes at a cost of longer distances covered until the robot believes coverage is achieved. $d_{\text{max}}$ is between 1.5 and 3 times larger for our approach, $d_{\text{exp}}$, however, only up to 1.5 larger. This happens because the proposed approach needs to consolidate frontiers of all spatially close polygons through place recognition, while the grid-based representation is able to consolidate frontiers from measurements even if the relative pose established between them comes from an estimate integrated over a very long path. A typical evolution of the coverage ratios for one sample of each method and noise multiplier $\alpha = 1$ is shown in Fig. 7.



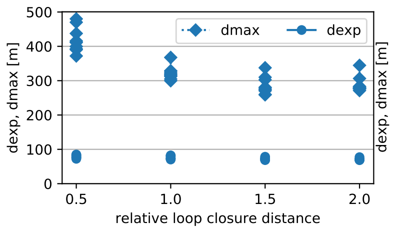
(a) Open (b) Maze (c) Forest house

Fig. 7: True coverage as a function of distance travelled for individual runs at noise multiplier $\alpha = 1$. Vertical dashed lines indicate the distance at which the robot believes coverage to be complete according to its own representation.

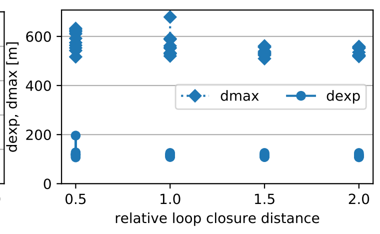
Fig. 8 shows how our method is affected by the loop closure distance $d_{\text{pr}}$. As can be seen, a larger distance at which places can be recognized leads to faster exploration times in the open environment. This makes sense, as a larger loop closure distance leads to generally larger frontier consolidation scopes, allowing frontiers to be removed faster. Exploration speed in the maze environment, however, is not significantly affected, as there are generally not many place recognition events happening in that environment.

## 8 Validation in the Real World

We validate that our approach works in the real world with the experimental setup shown in Fig. 9. The platform is a ClearPath *Jackal* equipped with a

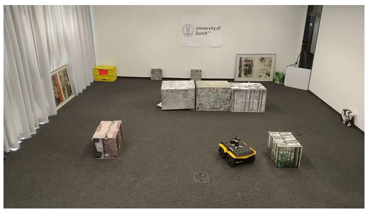


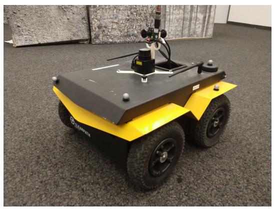
(a) Open



(b) Maze

Fig. 8: $d_{\mathrm{exp}}$ and $d_{\mathrm{max}}$ when using our representation in the *open* and *maze* environments with noise multiplier $\alpha = 0.5$, for different relative loop closure distances $\beta$. Place recognition is provided by the simulator for any two poses when the distance between them is below $\beta$ times the depth sensor range.





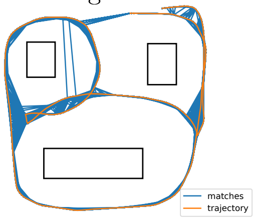


Fig. 9: (a) Real environment in which we have validated our approach. (b) Close-up of our experimental platform. (c) NetVLAD matches on a trajectory circumnavigating all obstacles. Obstacle locations are approximate.

Hokuyo laser scanner as depth sensor and two fisheye cameras that provide a 360° view of the environment. Ground truth is obtained with a motion tracking system using reflective IR markers. For the state estimate, we use the wheel odometry provided by the Jackal robot. We intentionally do not use the best available state estimate, to demonstrate robustness to drift.

The most important assumption that we have made in our simulations is that sufficient place recognition and relative pose estimation can be provided. To this end, we use visual place recognition from a panoramic image stitched from the two fisheye cameras. The vertical field of view is restricted to prevent place recognition from structure that is visible from everywhere in the room. Using a 360° view allows place recognition independent of the orientation of the robot between the two matched places. First, the CNN full image desciptor NetVLAD [2] is used to quickly determine the visually most similar previously recorded image. Fig. 9 (c) shows NetVLAD matches in a trajectory where the robot drives once around each obstacle. As can be seen, all place recognitions occur locally, with only two matches across a longer distance, but still within line of sight. Given a NetVLAD match, we match ORB features [24] and use P3P [11] and RANSAC [9] for geometric verification and relative pose estimation. For P3P, the 3D locations of features in each frame are triangulated using KLT tracks [19] of those features in subsequent frames. The only component that we have not implemented, as it is out of the scope of this project, is teach-and-repeat [8]. To simulate teach-and-repeat, we instead use the motion tracking system to let the robot backtrack its trajectory. This, and ground truth for evaluation, are the only things for which the motion tracking system is used.

Fig. 10 (a) shows the final state of exploration in our experiment. As we can

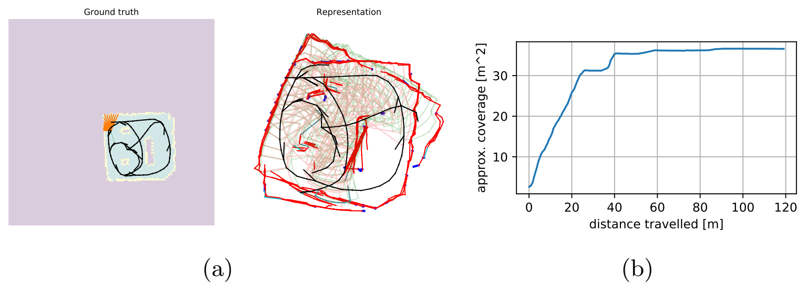


Fig. 10: (a) Ground truth and representation once exploration is complete. (b) Corresponding coverage over time, obtained from the ground truth grid.

see, there is drift in the estimated trajectory, yet our approach still manages to fully cover the environment. Note that there are parts of the environment where trajectories overlap. These are locations where our visual relative pose estimator fails to obtain enough feature matches. However, as long as relative pose estimations are obtained within the distance of the consolidation scope (see polygons shown in Fig. 10 (a)), this does not pose a significant problem. Fig. 10 (b) shows the corresponding coverage over the travelled distance. The behaviour is consistent with the results obtained in simulation, wherein the robot first quickly covers most of the map, after which it does a lot of backtracking to seek out frontiers that remain in the map.

## 9 Conclusion

In this paper, we present a map representation that allows exploration in spite of large drift in the pose estimate. We show that global consistency in the pose graph is not required to determine if exploration is complete. This alleviates the need for map optimization, which is particularly interesting for multi-robot exploration. In addition, the proposed method can be adapted to algorithms that currently use different representations. Using a state-of-the-art exploration algorithm, we compare our representation to a grid-based representation. In contrast to the latter, and at a cost of longer exploration time, all of the free space can be fully covered with our representation, even with large drift in the state estimate.

## Acknowledgments

This work was supported by the National Centre of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation and the SNSF-ERC Starting Grant.

## References

1. Akdeniz, B.C., Bozma, H.I.: Exploration and topological map building in unknown environments. In: IEEE Int. Conf. Robot. Autom. (ICRA), pp. 1079–1084 (2015). DOI 10.1109/icra.2015.7139310
2. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 5297–5307 (2016). DOI 10.1109/CVPR.2016.572
3. Blöchliger, F., Fehr, M., Dymczyk, M., Schneider, T., Siegwart, R.: Topomap: Topological mapping and navigation based on visual SLAM maps. In: IEEE Int. Conf. Robot. Autom. (ICRA), pp. 1–9 (2018). DOI 10.1109/ICRA.2018.8460641
4. Bosse, M., Newman, P., Leonard, J., Soika, M., Feiten, W., Teller, S.: An Atlas framework for scalable mapping. In: IEEE Int. Conf. Robot. Autom. (ICRA), vol. 2, pp. 1899–1906 (2003). DOI 10.1109/robot.2003.1241872
5. Caccavale, A., Schwager, M.: Wireframe mapping for resource-constrained robots. In: IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), pp. 1–9 (2018). DOI 10.1109/IROS.2018.8594057
6. Cieslewski, T., Kaufmann, E., Scaramuzza, D.: Rapid exploration with multi-rotors: A frontier selection method for high speed flight. In: IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), pp. 2135–2142 (2017). DOI 10.1109/IROS.2017.8206030
7. Connolly, C., et al.: The determination of next best views. In: IEEE Int. Conf. Robot. Autom. (ICRA), vol. 2, pp. 432–435 (1985)
8. van Es, S.K., Barfoot, T.D.: Being in two places at once: Smooth visual path following on globally inconsistent pose graphs. In: Conf. Comput. Robot Vis. (CRV) (2015). DOI 10.1109/crv.2015.17
9. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981). DOI 10.1145/358669.358692
10. Furgale, P., Barfoot, T.D.: Visual teach and repeat for long-range rover autonomy. J. Field Robot. **27**(5), 534–560 (2010). DOI 10.1002/rob.20342
11. Gao, X.S., Hou, X.R., Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. IEEE Trans. Pattern Anal. Mach. Intell. **25**(8), 930–943 (2003). DOI 10.1109/TPAMI.2003.1217599
12. González-Baños, H.H., Latombe, J.C.: Navigation strategies for exploring indoor environments. Int. J. Robot. Research **21**(10–11), 829–848 (2002). DOI 10.1177/027836490202101083
13. Greene, W.N., Roy, N.: FLaME: Fast lightweight mesh estimation using variational smoothing on delaunay graphs. In: Int. Conf. Comput. Vis. (ICCV), pp. 4696–4704 (2017). DOI 10.1109/ICCV.2017.502
14. Holz, D., Basilico, N., Amigoni, F., Behnke, S.: Evaluating the efficiency of frontier-based exploration strategies. In: Int. Symp. Robotics (ISR), pp. 1–8 (2010)
15. Howard, A., Parker, L.E., Sukhatme, G.S.: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. Int. J. Robot. Research **25**(5–6), 431–447 (2006). DOI 10.1177/0278364906065378
16. Howard, A., Sukhatme, G.S., Matarić, M.J.: Multi-robot mapping using manifold representations. Proc. IEEE **94**(9), 1360–1369 (2006)
17. Izadi, S., Newcombe, R.A., Kim, D., Hilliges, O., Molyneaux, D., Hodges, S., Kohli, P., Shotton, J., Davison, A., Fitzgibbon, A.: KinectFusion: Real-time dynamic 3D surface reconstruction and interaction. In: SIGGRAPH, p. 23 (2011)

18. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g2o: A general framework for graph optimization. In: IEEE Int. Conf. Robot. Autom. (ICRA) (2011)
19. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Int. Joint Conf. Artificial Intell. (IJCAI), pp. 674–679 (1981)
20. Millane, A., Taylor, Z., Oleynikova, H., Nieto, J.I., Siegwart, R., Cadena, C.: TSDF manifolds: A scalable and consistent dense mapping approach. arXiv e-prints (2017). URL `http://arxiv.org/abs/1710.07242`
21. Moravec, H., Elfes, A.: High resolution maps from wide angle sonar. In: IEEE Int. Conf. Robot. Autom. (ICRA), vol. 2, pp. 116–121 (1985). DOI 10.1109/ROBOT.1985.1087316
22. Oleynikova, H., Millane, A., Taylor, Z., Galceran, E., Nieto, J., Siegwart, R.: Signed distance fields: A natural representation for both mapping and planning. In: RSS Workshop: Geometry and Beyond - Representations, Physics, and Scene Understanding for Robotics (2016)
23. Papachristos, C., Khattak, S., Alexis, K.: Uncertainty-aware receding horizon exploration and mapping using aerial robots. In: IEEE Int. Conf. Robot. Autom. (ICRA), pp. 4568–4575 (2017). DOI 10.1109/ICRA.2017.7989531
24. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An efficient alternative to SIFT or SURF. In: Int. Conf. Comput. Vis. (ICCV) (2011)
25. Schmuck, P., Scherer, S.A., Zell, A.: Hybrid metric-topological 3d occupancy grid maps for large-scale mapping. IFAC-PapersOnLine **49**(15), 230–235 (2016). DOI 10.1016/j.ifacol.2016.07.738
26. Sünderhauf, N., Protzel, P.: Switchable constraints for robust pose graph SLAM. In: IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (2012)
27. Teixeira, L., Chli, M.: Real-time mesh-based scene estimation for aerial inspection. In: IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), pp. 4863–4869 (2016). DOI 10.1109/iros.2016.7759714
28. Wallgrun, J.O.: Hierarchical Voronoi Graphs: Spatial Representation and Reasoning for Mobile Robots. Springer Publishing Company, Incorporated (2010). DOI 10.1007/978-3-642-10345-2
29. Wurm, K.M., Hornung, A., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In: Proc. ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation (2010)
30. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: IEEE Int. Conf. Robot. Autom. (ICRA), pp. 146–151 (1997)