ARAS: Ambiguity-aware Robust Active SLAM based on Multi-hypothesis State and Map Estimations

Ming Hsiao*, Joshua G. Mangelson*, Sudharshan Suresh*, Christian Debrunner†, and Michael Kaess*

***Abstract*— In this paper, we introduce an *ambiguity-aware robust active SLAM (ARAS)* framework that makes use of multi-hypothesis state and map estimations to achieve better robustness. Ambiguous measurements can result in multiple probable solutions in a *multi-hypothesis SLAM (MH-SLAM)* system if they are temporarily unsolvable (due to insufficient information), our ARAS aims at taking all these probable estimations into account explicitly for decision making and planning, which, to the best of our knowledge, has not yet been covered by any previous active SLAM approach (which mostly consider a single hypothesis at a time). This novel ARAS framework 1) adopts local contours for efficient multi-hypothesis exploration, 2) incorporates an active loop closing module that revisits mapped areas to acquire information for hypotheses pruning to maintain the overall computational efficiency, and 3) demonstrates how to use the output target pose for path planning under the multi-hypothesis estimations. Through extensive simulations and a real-world experiment, we demonstrate that the proposed ARAS algorithm can actively map general indoor environments more robustly than a similar single-hypothesis approach in the presence of ambiguities.**

I. Introduction

Active SLAM is the problem of actively exploring an environment with moving sensors while simultaneously estimating the state of the sensors and reconstructing a map that satisfies certain requirements (e.g. bounded uncertainty). In general, an active SLAM system takes sensor measurements as inputs just like a passive SLAM system, yet it outputs a sequence of online decisions/actions to influence the future measurements in addition to the state and map estimates from passive SLAM only. The decisions/actions tend to fall into one of two complementary groups: *exploration* and *active loop closing*. The goal of exploration is to move the robot in the environment so that new information can be observed. A popular approach is to jointly optimize the information gain and motion cost based on a global map [2][29]. On the other hand, active loop closing is responsible for finding mapped places to revisit for loop closures that correct the accumulated drift and uncertainty [36][22][4][42].

Just as in other robotic solutions, robustness is a crucial challenge for active SLAM, especially when encountering the unavoidable problem of *ambiguous measurements* (or *ambiguities* in this paper). Ambiguity is the situation where

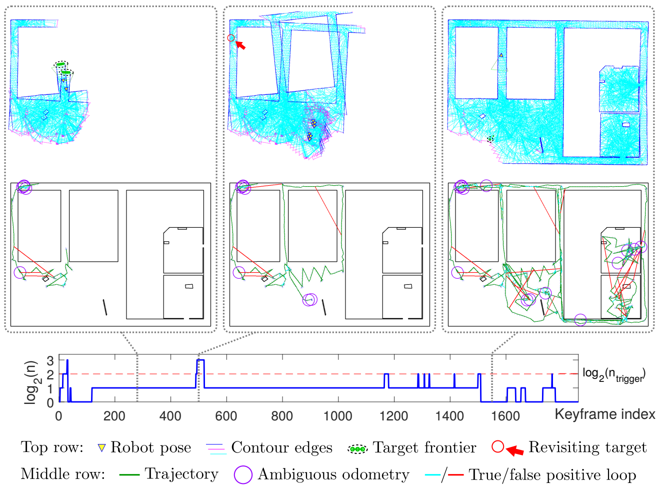
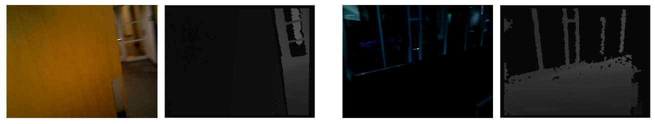


Fig. 1: This simulation example shows that the proposed ARAS framework can make use of multi-hypothesis state and map estimations for exploration (left), actively revisiting mapped areas when the number of hypotheses n goes beyond a threshold n_{trigger} (middle), and pruning unlikely hypotheses automatically when sufficient information is observed through loop closures (right). The top row shows all highly likely hypotheses of the robot pose and the global map reconstructed by local contours (see Sec. V-A). The middle row shows the ground truth environment and robot trajectory. The bottom row shows how n changes throughout the active SLAM process until the environment is fully covered (see Fig. 9-b).



(a) Dynamic scene (opening a door) (b) Lack of texture (too dark)

Fig. 2: Examples of ambiguities for fast dense RGB-D odometry [11].

more than one interpretation is plausible for the same observations. It stems from various sources including insufficient information, conflicts among different sensor measurements, uncertain data association, loop closing based on appearance only, and etc (see Fig. 2). Most existing active SLAM algorithms [40][3][5] consider unimodal state and map estimates in exploration and active loop closing, as they are built upon state estimation [14][6][39] and mapping methods [24][9] that only consider one single hypothesis (and possibly uncertainties). Therefore, when ambiguities occur, the single estimate can be polluted by wrong information, which result in poorly-informed actions or even full system failures.

Several nonparametric SLAM or multi-hypothesis SLAM (MH-SLAM) solutions have been proposed to handle ambiguities and output multimodal solutions accordingly when the ambiguities are temporarily unsolvable [23][8][25][13][10]. However, to the best of our knowledge, no existing *active SLAM* algorithm makes use of both multimodal state and map estimations explicitly in both

*M. Hsiao, J. G. Mangelson, S. Suresh and M. Kaess are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. {mhsiao, jmangels, suddhu, kaess}@cmu.edu

†C. Debrunner is with Lockheed Martin Corporation, Bethesda, MD 20817, USA. chris.debrunner@ieee.org

This work is partially supported by Lockheed Martin Corporation under research agreement A020478. We would like to thank K. Anderson, C. Dorosky, D. Fanning, A. Petersen, and S. Tangirala for discussions, and E. Westman for feedback on the experimental design.

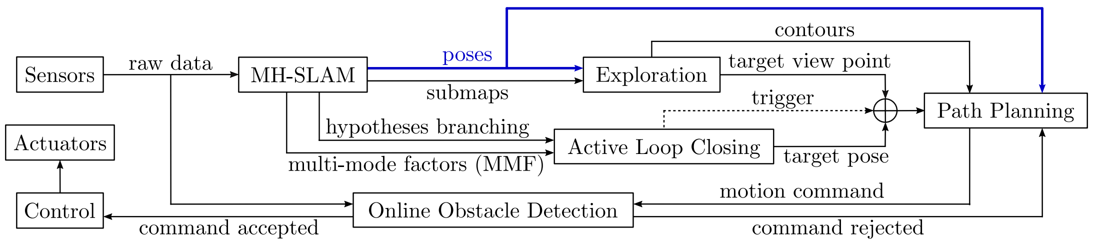


Fig. 3: Block diagram of the ARAS framework, which consists of four main modules: MH-SLAM, exploration, active loop closing, and path planning. Notice that the poses $\mathtt{T}_i$ (blue arrows) are the only multi-hypothesis information passed between modules.

exploration and active loop closing. Therefore, in this paper, we propose the *ambiguity-aware robust active SLAM (ARAS)* framework that takes all the multi-hypothesis estimations from a MH-SLAM system as inputs and outputs single decisions/actions to take at a given time (see Fig. 1).

Similar to other active SLAM solutions, each action selected by ARAS falls into one of two categories: exploration and active loop closing. Since maintaining all the global maps in each hypothesis for exploration is very inefficient in both memory and computation, we develop a multi-hypothesis exploration algorithm based on *ambiguity-free submaps* to achieve better efficiency. As for active loop closing, because the exponential growth of the number of hypotheses can be intractable to track, we aim at bounding the number of hypotheses based on the constraints of the system while keeping track of the correct hypothesis within. Therefore, the goal of our active loop closing algorithm is to find some places to revisit so that the detected loop closures can provide sufficient information to distinguish and prune enough wrong hypotheses to reduce computational cost, or even preserve the correct hypothesis only.

Finally, we implement a simple path planning method for both exploration and active loop closing in each hypothesis, which can be replaced by an advanced planner as needed. Notice that we assume the environment we want to map lies roughly on a horizontal plane (e.g.: one floor of a building), and the sensors also move roughly on a horizontal plane. These assumptions simplify the active SLAM problem into 2D (although the states and maps can still be 3D), which helps us to focus on making use of the multi-hypothesis estimations in ARAS efficiently.

The contributions of this work are:

1. Introducing the novel ambiguity-aware robust active SLAM (ARAS) framework (see Fig. 3) that uses the multi-hypothesis state and map estimates from MH-SLAM.

2. Developing an exploration algorithm based on ambiguity-free submaps to make use of the multi-hypothesis estimates efficiently.

3. Designing the active loop closing strategy to reduce the number of hypotheses for tractability.

4. Demonstrating using the output decision/action for path planning considering multiple hypotheses.

5. Evaluating the robust active SLAM algorithm through extensive simulations and discussing its properties.

6. Applying the proposed algorithm in a real-world task.

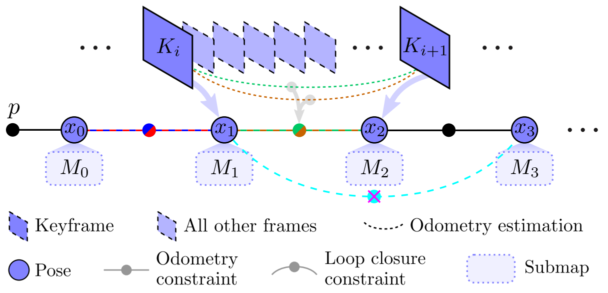


Fig. 4: The system structure of the MH-SLAM algorithm. The factors with more than one color are multi-mode factors (MMF). Please refer to Fig. 2 in [10] for the representations of various types of MMFs.

II. Related Work

While *partially-observable Markov decision processes (POMDP)* [35][16] offer a theoretical foundation to solve active SLAM problems, considering all possible scenarios resulting from ambiguous observations is computationally intractable. Practical solutions based on *belief space planning (BSP)* [15][1][31][30][32][34] have been developed to take ambiguous information into account. However, none of them considers the coexistence of multi-hypothesis states and map.

More recently, a robust exploration algorithm [41] based on *multiple hypothesis JCBB* [26] handles multi-hypothesis feature-based map and decides the future path based on the weighted combination of all the hypotheses. Another approach [33] focused on robust homing considers only two hypotheses: whether the up-to-date 2D map is consistent or not. The proposed ARAS takes both multi-hypothesis states and maps into account in both exploration and active loop closing, which increases the robustness of the entire system.

III. Multi-hypothesis SLAM Preliminaries

A. *MH-iSAM2*

The MH-SLAM in the ARAS framework adopts MH-iSAM2 [10] as its back-end solver, which is a nonlinear incremental optimizer that computes multi-hypothesis solutions based on input *single-/multi-mode factors (SMF/MMF)*. Ambiguous measurements are modeled as MMFs f_r^{M} with corresponding types and modes (see Sec. III-C in [10]), and the hypotheses of all *multi-hypothesis variables (MHV)* can be optimized accordingly and associated with each other through a *Hypo-tree* (see Sec. IV in [10]). In addition, only the highly probable hypotheses are preserved after *hypotheses pruning* (see Sec. VI in [10]) during each update.

B. *Submap-based MH-SLAM*

We adopt a keyframe-based framework similar to [11] in our MH-SLAM, which selects a new *keyframe* K_i whenever

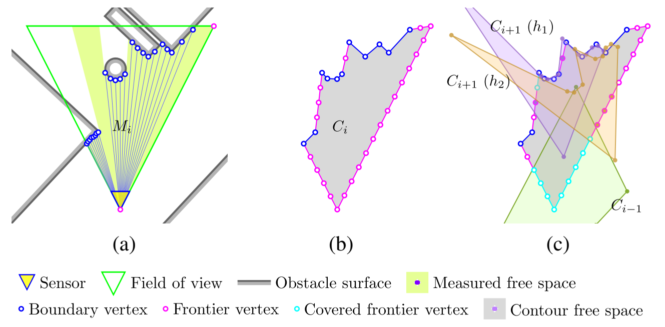


Fig. 5: The extraction and update of a local contour C_i. In this example, we visualize the algorithm steps in the local coordinates of the range/depth sensor at the most recent keyframe with its submap reconstructed already. (a) Boundary vertices $\mathbf{v}_p^{\mathrm{B}}$ are extracted from the submap M_i, and two frontier vertices $\mathbf{v}_p^{\mathrm{F}}$ are added at the sensor origin and the maximum range on the right edge. (b) More $\mathbf{v}_p^{\mathrm{F}}$ are interpolated with equal spacing, and some $\mathbf{v}_p^{\mathrm{B}}$ are removed to smooth the edge. (c) The frontier vertices can be updated as covered ($\mathbf{v}_p^{\mathrm{F}} \rightarrow \mathbf{v}_p^{\mathrm{F,c}}$) based on the neighboring submaps and their relative pose estimations in all hypotheses.

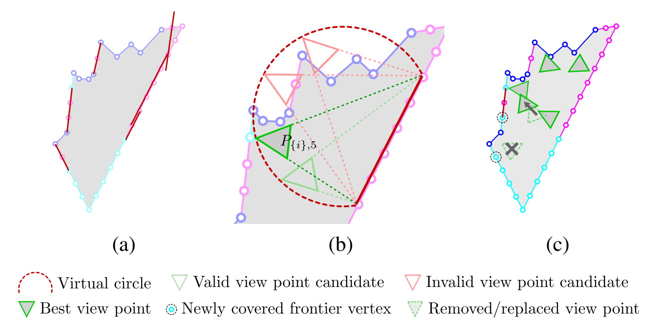

$P_{\{i\},5}$
(a)
(b)
(c)
Virtual circle
Valid view point candidate
Invalid view point candidate
Best view point
Newly covered frontier vertex
Removed/replaced view point

Fig. 6: The extraction and update of view points $P_{\{i\},k}$. (a) Frontier segments (dark red lines) are defined based on each sequence of uncovered frontier vertices. (b) To cover each frontier segment, candidate view points can be sampled on the virtual circle (only on one side) since all the *inscribed angles* that subtend the same arc have the same angle, which can be regarded as the FoV of the sensor. (c) Each selected best view point $P_{\{i\},k}$ can be updated or removed based on the updates of its $\mathbf{v}_p^{\mathrm{F}}$.

the overlapping scene between the current frame and the previous keyframe is less than a threshold in any hypothesis. We further assume that all ambiguities between any two keyframes can be summarized into an ambiguous odometry factor (defined as *type #1 MMF* in [10]), which models different measurements between the same two poses, or an ambiguous loop closure factor (defined as *type #3 MMF* in [10]), which models whether the measurement between two poses is valid or not. As a result, only the multi-hypothesis poses $\mathtt{T}_i$ (each modeled as a MHV as described in Sec. III-A) of all K_i are optimized globally in MH-iSAM2 for simplicity and efficiency. Moreover, we can generate an *ambiguity-free* local dense submap M_i at each K_i, by fusing the raw range or depth information from consecutive keyframes given their estimated poses into a less noisy, outlier-free 3D map representation (e.g.: *local depth fusion* in [11]). As each K_i is anchored to $\mathtt{T}_i$, the occupancy information at any global location in any hypothesis h can be preserved efficiently in all nearby $\mathtt{T}_{i,[h]}$ and their M_i without the need of generating individual global maps for each h.

IV. ARAS Framework

Based on the MH-SLAM preliminaries in Sec. III, the ARAS framework is developed with three other main modules: exploration, active loop closing, and path planning (see Fig. 3). In every iteration, the exploration module takes the submaps M_i and multi-hypothesis poses $\mathtt{T}_i$ as inputs from the MH-SLAM module, and generates a *target view point* P^* that aims at exploring unknown area (see Sec. V). And if the active loop closing is triggered, a *target submap* M^* will be found based on the hypotheses branching and modes in each MMF (see Sec. VI). Once a P^* or M^* is determined, it will be passed to the path planning module as a target pose, and a *motion command* will be generated accordingly (see Sec. VII). The motion command has to be verified as valid (no conflict with the current measurements) by an online obstacle detection module before being conducted.

Since both P^* are M^* are single outputs anchored to the multi-hypothesis pose of their corresponding keyframes, their global poses can be updated inherently and efficiently when the keyframe poses are updated in each hypothesis (see Sec. III). Moreover, as the only two modules that deal with the multi-hypothesis poses $\mathtt{T}_i$ directly, both exploration and path planning are designed to operate on the ambiguity-free M_i. As a result, expensive computations such as maintaining multi-hypothesis global maps can be avoided.

V. Multi-hypothesis Exploration

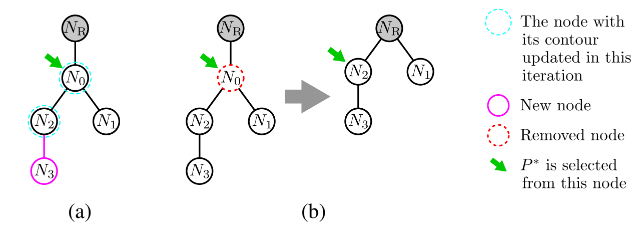
To design an efficient exploration algorithm considering multiple hypotheses, we take advantage of the ambiguity-free submap assumption in Sec. III to compute all *valid view points* in each submap based on its *local contour*, and design an *exploration tree* algorithm to choose one target view point at a time. We will go though these algorithms in this section.

A. Local Contours Extraction from Submaps

From each submap M_i (defined in Sec. III), we extract a *local contour* C_i to represent the local free space and encode the *frontier* [43] and obstacle boundary information. Each C_i consists of a set of vertices $\mathbf{v}_p$, and each $\mathbf{v}_p$ is labeled as on a boundary ($\mathbf{v}_p^{\mathrm{B}}$) or on a frontier ($\mathbf{v}_p^{\mathrm{F}}$).

To compute C_i, we first extract boundary vertices $\mathbf{v}_p^{\mathrm{B}}$ from the fused range/depth measurements in M_i on a given height (see Fig. 5-a) assuming known gravity $\mathbf{g}_i$ in the local coordinates (e.g.: the sensors are rigidly attached on a mobile platform that guarantees fixed $\mathbf{g}_i$, or equipped with an inertial sensor that can measure $\mathbf{g}_i$ correctly at any given time). Then, a frontier vertex $\mathbf{v}_p^{\mathrm{F}}$ is added at the sensor origin, while one or two other $\mathbf{v}_p^{\mathrm{F}}$ could be added at the maximum sensor range along the edges of the sensor's *field of view* if no $\mathbf{v}_p^{\mathrm{B}}$ is extracted near them. After sorting all these vertices in the clockwise angular ordering, we interpolate more frontier vertices $\mathbf{v}_p^{\mathrm{F}}$ between each pair of consecutive vertices that are farther apart than a threshold for better coverage on long straight frontiers (see Fig. 5-b). Finally, C_i is refined by removing some overly detailed $\mathbf{v}_p^{\mathrm{B}}$ to smooth its edges.

Every C_i has a *pseudo timestamp* t_i, which is initialized as the timestamp of its corresponding K_i. Any two contours C_i and C_j can be added as *neighbors* of each other if t_i and t_j are close enough, and at least the distance $d_{ij,[h]}$ between



N_R
N_0
N_1
N_2
N_3
(a)
(b)
The node with its contour updated in this iteration
New node
Removed node
P^* is selected from this node

Fig. 7: The construction and update of the exploration tree. (a) A new node N_3 that represents the new sets of view points $\mathcal{P}_3$ is added. (b) Assuming that C_3 covers all the remaining $\mathbf{v}_p^{\mathrm{F}}$ in C_0, $\mathcal{P}_0$ becomes empty and N_0 will be removed.

their poses ($\mathtt{T}_{i,[h]}$ and $\mathtt{T}_{j,[h]}$) in one of the hypotheses h is within a threshold. Whenever a loop between K_j and K_i (with $j > i$) is closed in all hypotheses, t_i will be updated as t_j so that every newly created nearby contour and C_i can become neighbors of each other. The same update is also applied to all the neighbors of C_i.

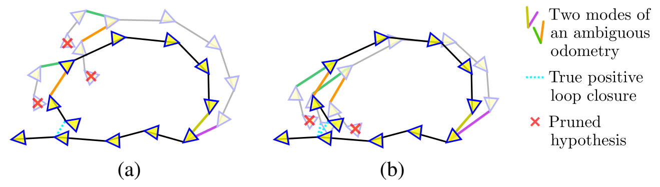
Whenever the number of neighbors of a contour increases, we check though each of its $\mathbf{v}_p^{\mathrm{F}}$ if it is inside at least one other neighboring contour in each hypothesis. If so, that $\mathbf{v}_p^{\mathrm{F}}$ will be updated as *covered*, which is denoted as $\mathbf{v}_p^{\mathrm{F}} \rightarrow \mathbf{v}_p^{\mathrm{F,c}}$ (see Fig. 5-c). Since the covered frontier vertices $\mathbf{v}_p^{\mathrm{F,c}}$ no longer represent the horizon of the explored region in any hypothesis, we will only use the uncovered $\mathbf{v}_p^{\mathrm{F}}$ to decide future view points for exploration (see Sec. V-B).

B. View Points Selection

In each contour C_i, a set of valid view points $\mathcal{P}_i = \{P_{\{i\},k},\, k \in \mathbb{N}\}$ in 2D is found to observe all the uncovered $\mathbf{v}_p^{\mathrm{F}}$ and the unknown space beyond them. Since C_i have summarized the local effects from all hypotheses (see Sec. V-A), each view point $P_{\{i\},k}$ just has to be computed once as a pose $\mathtt{T}_{\{i\}k}$ in the local coordinates of M_i, and transformed to different global or relative poses in each hypothesis using $\mathtt{T}_{i,[h]}$ (see Sec. III).

Starting from $\mathbf{v}_0^{\mathrm{F}}$ at the sensor origin and following the clockwise order, we define one *frontier segment* for each sequence of consecutive uncovered $\mathbf{v}_p^{\mathrm{F}}$, which is a line segment connecting the two frontier vertices at the two ends of the sequence with certain margin extended on both sides (see Fig. 6-a). Any frontier segment with its length exceeding a threshold is divided into several shorter ones until all of the frontier segments satisfy the length threshold. Then, we adopt the *inscribed angle theorem* to sample view point candidates on the *virtual circle* defined by each frontier segment given known open angle of the adopted range/depth sensor(s) (see Fig. 6-b). A view point candidate is valid only if it is inside C_i and its view is not blocked by any edge of C_i. Finally, the best view point $P_{\{i\},k}$ for each frontier segment is naively selected as the valid candidate with its viewing angle most perpendicular to the frontier segment. If all candidates are invalid, the frontier segment will be divided into two shorter segments, and the same algorithm will be repeated onto each of them until every frontier segment finds a $P_{\{i\},k}$.

When any $\mathbf{v}_p^{\mathrm{F}}$ is updated to $\mathbf{v}_p^{\mathrm{F,c}}$ (see Sec. V-A), the frontier segment that contains it will be updated (shortened if it is at one of the two ends). Then, a new view point $P'_{\{i\},k}$



Two modes of an ambiguous odometry
True positive loop closure
Pruned hypothesis
(a)
(b)

Fig. 8: Two examples of how a loop closure disambiguates the ambiguities modeled in the MMFs. (a) Usually, multiple undisambiguated MMFs $f_r^{\mathrm{M,u}}$ can be disambiguated by a single loop closure. (b) In special cases, some combinations of the modes of different $f_r^{\mathrm{M,u}}$ in different hypotheses can all seem to be correct even after the loop is closed.

will be computed according to the new frontier segment and substituted for $P_{\{i\},k}$ (see Fig. 6-c). If all $\mathbf{v}_p^{\mathrm{F}}$ in a frontier segment are removed, so are the corresponding $P_{\{i\},k}$.

C. Exploration Tree

A simple heuristic for efficient exploration it to finish an entire local area first (e.g.: one room) before exploring previously observed frontiers (e.g. entrance to another room). Therefore, we develop the exploration tree algorithm based on the update of local contours and view points (see Sec. V-A and Sec. V-B) to approximate this heuristic by first sorting $\mathcal{P}_i$ implicitly and then choose $P^*_{\{i\}}$ from the selected $\mathcal{P}_i$.

In the exploration tree, every node N_i represents one $\mathcal{P}_i$ except for the root N_{R}. Whenever a new submap M_i and its C_i and $\mathcal{P}_i$ are computed, a corresponding N_i will be added as the *first child* of the latest node that has its contour updated by C_i (see Fig. 7-a). And if any $\mathcal{P}_j = \{\emptyset\}$ due to the view point removal process, its corresponding node N_j will be deleted (see Fig. 7-b). In each iteration after updating the exploration tree, the first view point of the first child node of N_{R} is selected as the target view point $P^*_{\{i\}}$ (see the green arrows in Fig. 7), and its pose $\mathtt{T}^*_{\{i\}}$ in the local coordinates of the submap M_i is defined as the *local target pose* and passed to the path planning module (see Sec. VII). If all nodes are deleted except N_{R}, which means that no view point has to be visited, we can conclude the exploration and terminate the entire active SLAM algorithm. The process can be understood as always choosing $P^*_{\{i\}}$ from the oldest contour that is updated in the most recent step, which in practice performs very similar to the desired heuristic.

VI. Active Loop Closing for Correct Pruning

Since the adopted MH-iSAM2 conducts pruning automatically whenever the number of hypotheses exceeds a threshold (see Sec. III-A), the key to avoid pruning the correct hypothesis accidentally is to acquire sufficient information for correct pruning at proper timings. A simple active loop closing algorithm is introduced to cover it in this section.

A. Triggering

Assume that n_{limit} is the upper bound of the number of hypotheses our entire system can handle (see Sec. I) and n is the number of equally likely hypotheses (see Sec. III) in each iteration. Whenever $n > n_{\mathrm{limit}}$, the correct hypothesis might be pruned accidentally due to the lack of information. So, the active loop closing must be triggered at the right times so that loop closures can be detected and registered to provide

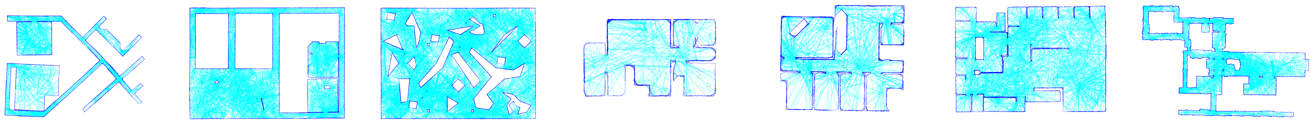


Fig. 9: Plotting all C_i in the global coordinates shows that our algorithm can achieve full coverage in various indoor environments in simulation with default settings: $n_{\text{trigger}} = 16$, $n_{\text{limit}} = \infty$, $p_{\text{a}} = 1\%$ and $p_{\text{f}} = 2\%$. The edges formed by $\mathbf{v}_p^{\text{B}}$ and $\mathbf{v}_p^{\text{F,c}}$ are shown in blue and cyan respectively.

sufficient information for *correct pruning* (keep tracking the correct hypothesis) while always satisfying $n \leq n_{\text{limit}}$.

Since new ambiguities might occur on the way to revisit M^* and further increase n before any true positive loop is detected, the ideal approach is to predict future ambiguities and loop closures, and trigger active loop closing accordingly beforehand. Even though there are existing algorithms that help predicting future loop closures [19], it is still an open question on how to predict future ambiguities. As a result, we simply choose a threshold $n_{\text{trigger}} < n_{\text{limit}}$ and trigger active loop closing when $n > n_{\text{trigger}}$ as a baseline approach. And if n really goes beyond n_{limit}, we choose to prune some of the hypotheses before obtaining sufficient information, and test various combinations of n_{trigger} and n_{limit} in the simulation in Sec. VIII-B to offer a good reference for choosing n_{trigger} based on n_{limit} for real-world applications. More discussion of this issue can be found in Sec. VIII-D.

B. *Target Submap Selection*

When the active loop closing is triggered, a target submap M^* will be selected from the existing submaps for revisiting. Then, at least one loop that can result in correct pruning is expected to be detected and closed when the pose of M^* is successfully revisited, or even earlier on the way to M^*.

To select a proper M^*, we first check for each ambiguous odometry factor f_r^{M} (see Sec. III) if more than one of its modes are selected in all existing hypotheses. If so, we can tell that the ambiguity modeled in f_r^{M} is not disambiguated yet, and flag f_r^{M} as *undisambiguated* (denoted as $f_r^{\text{M,u}}$). In this case, a loop closure measurement from the current submap M_i to any submap M_j prior to $f_r^{\text{M,u}}$ is very likely to provide the information that distinguishes the correct mode from the wrong ones, and allows correct pruning right after. If there are multiple $f_r^{\text{M,u}}$ between M_i and M_j, it is possible to disambiguate all of them together with one single loop closure connecting M_i and M_j (see Fig. 8-a). However, it can fail in some special cases (see Fig. 8-b).

Based on the above discussions, we randomly select one target submap M_j^* from all the submaps prior to the earliest $f_r^{\text{M,u}}$. And if no loop is detected before M_j^* is reached, we will select another $M_{j'}^*$ and repeat this process until $n \leq n_{\text{trigger}}$. This simple approach can be replaced by more advanced methods as needed (see Sec. VIII-D).

VII. Path Planning with Multi-hypothesis Poses

This section describes how to conduct simple path planning with multi-hypothesis state and map estimates. In each hypothesis h, we first compute the *global target pose* $\mathtt{T}_{[h]}^* = \mathtt{T}_{i,[h]}\mathtt{T}_{\{i\}}^*$ in the case of exploration towards $P_{\{i\}}^*$ (see Sec. V-C) or $\mathtt{T}_{[h]}^* = \mathtt{T}_{j,[h]}$ in the case of active loop closing towards M_j^* (see Sec. VI-B). Then, we check if there exists a valid path (collision-free against any mapped obstacles) from the current pose $\mathtt{T}_{N,[h]}$ to the target pose $\mathtt{T}_{[h]}^*$ that consists of one or multiple *straight paths* between existing poses. If such path exists, the output motion command will be moving directly along the straight line starting from $\mathtt{T}_{N,[h]}$ with a certain distance. If no path is found in h, we switch to the next hypothesis $h+1$ and repeat the same process. Finally, when $\mathtt{T}_{[h]}^*$ is reached, a new P^* or M^* will be passed into the path planning module since either the current $P_{\{i\}}^*$ is covered and removed, or the switch between exploration and active loop closing occurs. If the growing or pruning of hypotheses happens on the way to the same target, the new hypothesis h' will be selected as the first child of h (if it exists), or the first child of the next valid sibling of h (if h is pruned by the *backward pruning* [10]). Finally, if no motion command can be found in any hypothesis, or all the commands are rejected by the obstacle detection module (see Sec. IV), random small motions will be conducted repeatedly until a valid path is found again to avoid the robot being stuck in corner cases. Notice that we can easily replace this simple method with more advanced trajectory planning algorithms (e.g.: RRT [21] or PRMs [18]) as needed.

VIII. Experimental Results

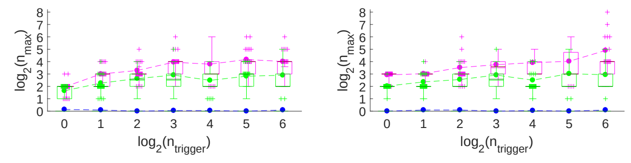
A. *Implementations and Settings*

Both the simulation and the system for real-world application are implemented in C++ and executed on a laptop with an Intel Core i7-8850H processor. Exploration and active loop closing are computed in two parallel threads, and both of them are one keyframe behind the MH-SLAM process to wait for the latest submap being generated.

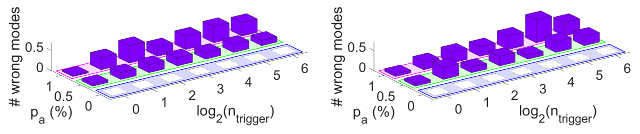
B. *Simulation*

In all simulations, Gaussian noise is added to all depth data, odometry, loop closures, and robot motions. We adopt two different ways to generate ambiguities in odometry. In the first two parts of our simulations, they are set to occur randomly with probabilities p_{a}, which simulates the types of ambiguities resulting from aggressive motion or dynamic scenes. And in the last part, they are set to occur when the robot is close to one of the several randomly selected places, which simulates the other types of ambiguities resulting from lack of texture or repeated patterns in the environment. False positive loop closures are set to occur randomly with probabilities p_{f} in all these simulations. The values of all the wrong modes in these simulated ambiguities are randomly generated within reasonable ranges.

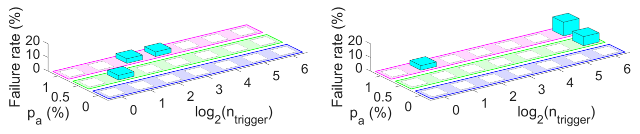
We first show that ARAS can achieve full coverage in various simulated environments in Fig. , then evaluate it with



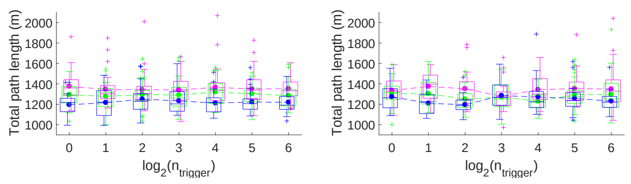
(a) Box plot and means (dotted curves) of $n_{\max}$



(b) Bar graph of the average number of wrong modes taken



(c) Bar graph of failure rate



(d) Box plot and means (dotted curves) of the total path lengths

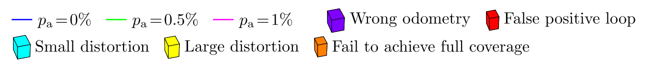


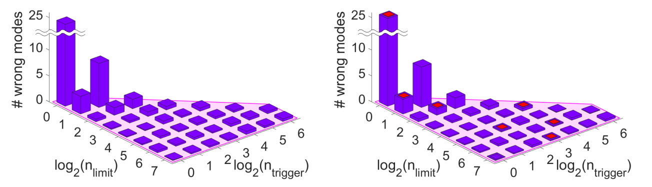
Fig. 10: Simulation results with $n_{\text{limit}} = \infty$ and various n_{trigger}. Three different p_{a} (0%, 0.5% and 1%) are tested with $p_{\text{f}} = 0\%$ (left column) and $p_{\text{f}} = 2\%$ (right column). Different levels of distortions in (c) are categorized based on thresholds on the *absolute trajectory error (ATE)* [37].

various test cases in the same multi-loop environment (see Fig. 9-b) and generate statistical results from 30 runs of each case. The results are shown in Fig. 10, 11, and 12.

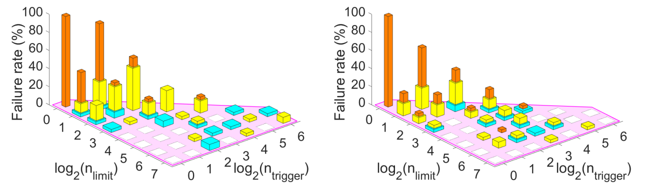
Fig. 10 shows the fundamental properties of ARAS with $n_{\text{limit}} = \infty$ and various n_{trigger}. Even though $n_{\text{limit}} = \infty$, the maximum number of hypothesis ever tracked (denoted as $n_{\max}$) does not grow unbounded (see Fig. 10-a) since true positive loop closures are very likely to be detected shortly after the active loop closing is triggered. The correct hypothesis can still be pruned occasionally (see Fig. 10-b) since some wrong hypotheses that take wrong modes (especially in the ambiguous odometry factor) might seem more likely than the correct one temporarily due to the accumulated drift. Therefore, some reconstructed maps are polluted (see Fig. 10-c). However, ARAS can still explore the entire environment with these slightly distorted maps. Finally, Fig. 10-d shows that larger p_{a} results in longer path length, but neither n_{trigger} nor p_{f} has a strong effect on it.

Fig. 11 shows the advance evaluation on robustness based on various combinations of n_{trigger} and n_{limit}. Comparing to the results in Fig. 10-b and Fig. 10-c, bounding n_{limit} increases the number of wrong modes taken and the failure rate as expected. In some cases when the map is extremely distorted, ARAS even fails to complete the task (shown in orange in Fig. 11-b). Since larger n_{limit} and smaller n_{trigger} results in less failure, setting a smaller n_{trigger} under the same n_{limit} is better for the overall robustness in general.

Lastly, in Fig. 12 we change the way of generating

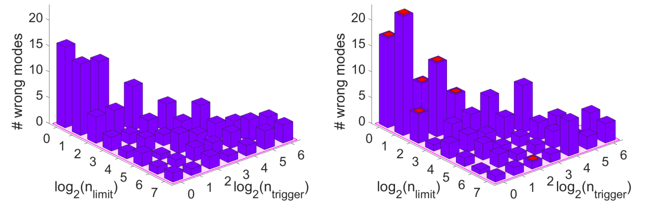


(a) Bar graph of the average number of wrong modes taken

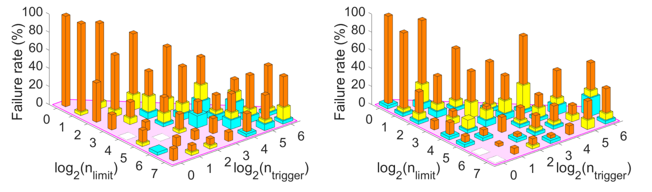


(b) Bar graph of failure rate

Fig. 11: Simulation results with various n_{trigger} and n_{limit} given $p_{\text{a}} = 1\%$. The legends are the same as in Fig. 10. $p_{\text{f}} = 0\%$ and $p_{\text{f}} = 2\%$ are again shown in the left and right columns respectively. The leftmost bars in each graph represent the results of active SLAM with single-hypothesis estimates, where many wrong modes are taken and all the runs fail drastically. The bars are colored in the same way as defined in Fig. 10.



(a) Bar graph of the average number of wrong modes taken



(b) Bar graph of failure rate

Fig. 12: Simulation results with ambiguous odometry measurements generated based on locations in the simulated environment. The bars are colored in the same way as defined in Fig. 10.

ambiguities in odometry estimation to simulate ambiguities that result from the environments instead. In this simulation, whenever an ambiguity occurs, the robot is very likely to observe similar types of ambiguities when it is still moving around the same area, and has to travel a longer distance (leave the area where the ambiguity occurs) before being able to detect a valid loop closure to disambiguate the ambiguities. As a result, more wrong modes are taken as shown in Fig. 12-a, which results in higher failure rates in general as shown in Fig. 12-b. Moreover, we can see that for the cases with n_{trigger} and n_{limit} are close to each other, their failure rates increase more than other cases comparing to the results in Fig. 11-b, which implies that to deal with environment-oriented ambiguities, a larger buffer for the number of growing hypotheses from n_{trigger} to n_{limit} is desired to maintain correct pruning.

C. Real-world Experiment

We develop an *assistive mapping system* based on ARAS that guides a human user to explore and map indoor environments with hand-held sensors though instructions (locations of P^* and M^*) on an augmented reality (AR) view and a top-down view (see Fig. 13-a). The adopted MH-SLAM system is modified from DPI-SLAM [12], where the same sensor setup is used to capture RGB-D and inertial data. However, the front-end is modified to detect and model ambiguities, and the back-end is replaced by MH-iSAM2 [10]. Any joint estimate of the fast dense RGB-D odometry [11] and IMU preintegration [7] is regarded as ambiguous if the estimated IMU bias is larger than a threshold. In this case, we assume that either of them (RGB-D or IMU only) still estimates the states correctly, and model their individual estimates as two modes in a type #1 MMF [10]. For generality, every loop closure is set as a type #3 MMF [10].

Unlike in the simulation, additional planar constraints [17] that satisfy all hypotheses are jointly optimized with the keyframe poses globally, which reduces rotational drift and helps correct pruning. Moreover, since a human user can plan a path and avoid obstacles, these two functions are disabled for this application. Finally, since we cannot or might not want to map certain areas beyond the frontiers, e.g.: areas behind a glass window, private spaces (offices or labs), or the bridge to another building, we add a function that allows the user to manually delete the current P^*, which will update the exploration tree accordingly and select a new P^*.

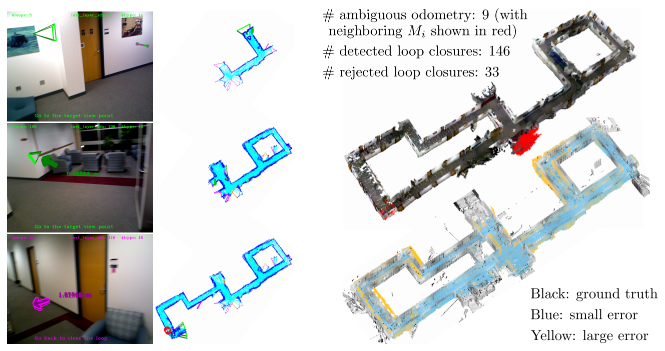
Since we cannot tell if the correct hypothesis is preserved in the real-world task, we calculate the point-to-plane mean absolute error (MAE) and root-mean-square error (RMSE) of the output 3D model with respect to a ground truth survey LiDAR model instead for evaluation. The result in Fig. 13-b shows that our assistive mapping system successfully helps the user to explore a large indoor environment and reconstruct its dense 3D model even with multiple ambiguous odometry estimations and false positive loop closures. Notice that the active SLAM process is conducted in real-time with the human user walking and rotating at normal pace and speed, and the ambiguities in this experiment are resulting from lack of features (close up to a blank wall that can be seem in the right half of the bottom AR viewer in Fig. 13-a) during fast motion and the appearance-based loop closure detector (similar scenes at different locations). We also modify the assistive mapping system to work with only one single hypothesis, and it fails quickly because it includes outlier measurements in the optimization.

D. Discussion

Based on the simulation results, we can tell that when the number of ambiguities is large but the computation is limited, we have to sacrifice speed for robustness. Else, ARAS would still fail to complete the active SLAM task. As a result, when designing a real world robotic system, we should try to solve the ambiguity problems during the state and map estimation process or even earlier, e.g. adopting proper sensor combination based on the motions, environments, and tasks to reduce ambiguities, and only passing the unsolvable ambiguities to the other modules in ARAS.

In our real-world application, we find that active loop closing is triggered more often with a small n_{trigger}, which results in bad user experience if the user follows the revisiting instructions to walk back and forth frequently. So, we set $n_{\text{trigger}} = 2$ instead of 1 in this task to make the process more comfortable. However, more studies on how to trigger active loop closing considering all aspects is still of interest. In addition, we also want to try if taking visual or structural saliency [27] into account during target submap selection helps detect more true positive loop closures. Finally, ARAS is only robust to ambiguities but not to some other problems. Therefore, integrating more functions to handle various real world challenges (e.g.: relocalization [20] for tracking failures in all sensors) is still desired for better robustness of the entire system.

Lastly, extending current ARAS implementation to conduct active SLAM in full 3D is possible. However, it might require the usage of polyhedrons, which might not be as efficient as the current contour-based approach in 2D. We are looking forward to studying more on this direction for other applications such as underwater exploration [28][38].


# ambiguous odometry: 9 (with neighboring M_i shown in red)
# detected loop closures: 146
# rejected loop closures: 33
Black: ground truth
Blue: small error
Yellow: large error

(a) AR and top-down viewers (b) Dense 3D model and evaluation

Fig. 13: The application of ARAS for assistive mapping. (a) Examples of the AR (left) and top-down (right) views showing P^* or M^* during the active SLAM process. (b) The output colored 3D models of all hypotheses (top) and the evaluation of the most likely model (bottom). The MAE and RMSE are 0.187m and 0.269m respectively. Given that the entire model is about 58m×15m×3m, the average error ratio is less than 2%.

IX. Conclusion

We proposed the first ambiguity-aware robust active SLAM (ARAS) framework that makes use of multi-hypothesis state and map estimations from a MH-SLAM system to handle ambiguities and achieve better robustness. Its exploration module selects possible view points based on local submaps, which avoids the complexity of computing the view points in each of the hypotheses explicitly. And under reasonable conditions, the active loop closing algorithm helps reduce the growing number of hypotheses by providing loop closure information when needed, which makes maintaining the multi-hypothesis solutions computationally efficient. In addition, a simple path planning method is adopted to compute motion commands that move towards the target locations. The experimental results show that explicitly considering multiple highly possible hypotheses significantly improves the robustness of active SLAM, and that it is possible to achieve robustness and efficiency at the same time with a carefully designed system and fine-tuned parameters under certain scenarios. In the future, we will address the remaining issues discussed in Sec. VIII-D, and try different combinations of sensors and algorithms in ARAS. The ultimate goal is to integrate ARAS into a fully autonomous robotic system that conducts tasks robustly.

REFERENCES

[1] A. Agha-mohammadi, S. Agarwal, A. Mahadevan, S. Chakravorty, D. Tomkins, J. Denny, and N. M. Amato, "Robust online belief space planning in changing environments: Application to physical mobile robots," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2014, pp. 149–156.

[2] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2016, pp. 1462–1468.

[3] L. Carlone, J. Du, M. K. Ng, B. Bona, and M. Indri, "Active SLAM and exploration with particle filters using Kullback-Leibler divergence," *Journal of Intelligent and Robotic Systems*, vol. 75, pp. 291–311, 2014.

[4] S. M. Chaves, A. Kim, and R. M. Eustice, "Opportunistic sampling-based planning for active visual SLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sep. 2014, pp. 3073–3080.

[5] Y. Chen, S. Huang, R. Fitch, L. Zhao, H. Yu, and D. Yang, "On-line 3D active pose-graph SLAM based on key poses using graph topology and sub-maps," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2019, pp. 169–175.

[6] A. Doucet, N. de Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic bayesian networks," in *Conf. on Uncertainty in Artificial Intelligence*, 2000, pp. 176–183.

[7] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Trans. Robotics*, vol. 33, no. 1, pp. 1–21, Feb 2017.

[8] D. Fourie, J. Leonard, and M. Kaess, "A nonparametric belief solution to the Bayes tree," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 2189–2196.

[9] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, vol. 34, 04 2013.

[10] M. Hsiao and M. Kaess, "MH-iSAM2: Multi-hypothesis iSAM using bayes tree and hypo-tree," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2019, pp. 1274–1280.

[11] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar SLAM," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2017, pp. 5110–5117.

[12] M. Hsiao, E. Westman, and M. Kaess, "Dense planar-inertial SLAM with structural constraints," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2018, pp. 6521–6528.

[13] G. Huang, M. Kaess, J. J. Leonard, and S. I. Roumeliotis, "Analytically-selected multi-hypothesis incremental MAP estimation," May 2013, pp. 6481–6485.

[14] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose SLAM," *IEEE Trans. Robotics*, vol. 26, no. 1, pp. 78–93, Feb 2010.

[15] L. P. Kaelbling and T. Lozano-Pérez, "Unifying perception, estimation and action for mobile manipulation via belief space planning," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2012, pp. 2952–2959.

[16] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[17] M. Kaess, "Simultaneous localization and mapping with infinite planes," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2015, pp. 4605–4611.

[18] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[19] A. Kim and R. M. Eustice, "Perception-driven navigation: Active visual SLAM for robotic area coverage," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2013, pp. 3196–3203.

[20] G. Klein and D. Murray, "Improving the agility of keyframe-based SLAM," in *Eur. Conf. on Computer Vision (ECCV)*, 2008, pp. 802–815.

[21] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Tech. Rep.*, 1998.

[22] W. L. D. Lui and R. Jarvis, "An active visual loop closure detection and validation system for topological SLAM," in *Australasian Conference on Robotics and Automation*, 2010.

[23] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, vol. 2, Sept 2003, pp. 1985–1991.

[24] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI Magazine*, vol. 9, pp. 61–74, 1988.

[25] B. Mu, S. Y. Liu, L. Paull, J. Leonard, and J. P. How, "SLAM with objects using a nonparametric pose graph," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 4602–4609.

[26] J. Neira and J. D. Tardos, "Data association in stochastic mapping using the joint compatibility test," *IEEE Trans. Robotics and Automation*, vol. 17, no. 6, pp. 890–897, Dec 2001.

[27] P. Newman and K. Ho, "SLAM-loop closing with visually salient features," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, April 2005, pp. 635–642.

[28] N. Palomeras, M. Carreras, and J. Andrade-Cetto, "Active SLAM for autonomous underwater exploration," *Remote Sensing*, vol. 11, p. 2827, 11 2019.

[29] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2017, pp. 4568–4575.

[30] S. Pathak, A. Thomas, and V. Indelman, "Nonmyopic data association aware belief space planning for robust active perception," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2017, pp. 4487–4494.

[31] S. Pathak, A. Thomas, A. Feniger, and V. Indelman, "Robust active perception via data-association aware belief space planning," in *Eur. Conf. on AI (ECAI)*, 2016.

[32] S. Pathak, A. Thomas, and V. Indelman, "A unified framework for data association aware robust belief space planning and perception," *Intl. J. of Robotics Research*, vol. 37, no. 2-3, pp. 287–315, 2018. [Online]. Available: https://doi.org/10.1177/0278364918759606

[33] D. Perea Strm, I. Bogoslavskyi, and C. Stachniss, "Robust exploration and homing for autonomous robots," *J. of Robotics and Autonomous Systems*, vol. 90, no. C, pp. 125–135, Apr. 2017. [Online]. Available: https://doi.org/10.1016/j.robot.2016.08.015

[34] S. C. S. Agarwal, A. Tamjidi, "Motion planning for active data association and localization in non-Gaussian belief spaces," in *Workshop on the Algorithmic Foundations of Robotics*, 2016.

[35] E. J. Sondik, "The optimal control of partially observable markov processes over the infinite horizon: Discounted costs," *Oper. Res.*, vol. 26, no. 2, pp. 282–304, Apr. 1978. [Online]. Available: http://dx.doi.org/10.1287/opre.26.2.282

[36] C. Stachniss, D. Hahnel, and W. Burgard, "Exploration with active loop-closing for FastSLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, Sept 2004, pp. 1505–1510 vol.2.

[37] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct. 2012, pp. 573–580.

[38] S. Suresh, P. Sodhi, J. G. Mangelson, D. Wettergreen, and M. Kaess, "Active SLAM using 3D submap saliency for underwater volumetric exploration," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[39] S. Thrun and M. Montemerlo, "The graph SLAM algorithm with applications to large-scale mapping of urban structures," *Intl. J. of Robotics Research*, vol. 25, pp. 403–429, 05 2006.

[40] R. Valencia, J. Valls Miró, G. Dissanayake, and J. Andrade-Cetto, "Active pose SLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2012, pp. 1885–1891.

[41] J. Wang and B. Englot, "Robust exploration with multiple hypothesis data association," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.

[42] S. Wenhardt, B. Deutsch, E. Angelopoulou, and H. Niemann, "Active visual object reconstruction using D-, E-, and T-optimal next best views," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, June 2007, pp. 1–7.

[43] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, July 1997, pp. 146–151.