Fisher Information and Traversability estimation-based Active SLAM for exploration in 3D environments

Undergraduate Thesis

Submitted in partial fulfillment of the requirements of
BITS F421T Thesis

By

Suchetan R S
ID No. 2020A3TS1760G

Under the supervision of:

Dr. Damien VIVET
&
Dr. Amalin PRINCE

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, GOA CAMPUS
December 2023

"If I have seen further than others, it is by standing upon the shoulders of Giants."

- Isaac Newton

Acknowledgements

An extraordinary musical instrument is a silent canvas, it remains of no use without the musicians. I would like to acknowledge all the musicians in my life during the 6 months of my Bachelor's thesis. Without them, I may not have been who I am today.

This thesis work would have been impossible without the immense help of my supervisor Dr. Damien Vivet. I feel incredibly fortunate to have found an advisor of your calibre and warmth. Every discussion we have had has been a memorable one. I am so very grateful to have worked under your supervision. Your wisdom, excellence and genuine passion for Robotics has not only helped me become better at carrying out research but has also left a lasting impact on my personal and professional growth.

I would like to convey my gratitude to Dr. Caroline Chanel. The comments, reviews along with the discussions that we had played a key role in drafting of my work without which it would have been unachievable. Additionally, my heartfelt thanks extends to Dr. Corentin Chauffaut. Your expertise with our robots not only helped me breeze through my experiments but also enabled me to climb a steep learning curve. I deeply appreciate your invaluable contribution to my work.

In extending my sincere appreciations, due recognition is undoubtedly owed to Dr. Amalin Prince. You have always been extremely approachable regardless of my problem. Seeking immediate help from you in many ways has played a big role in my thesis. I am profoundly thankful for your unwavering support throughout this journey.

I would like to express my sincere gratitude to my coworkers. The moments we have shared during this endeavor have truly enhanced my experience. I am thankful for the positive and cooperative environment we cultivated in our office, which has made this demanding journey all the more fulfilling. The jokes, laughs and the banter we shared has truly made this place a home away from home.

I want to express my deep gratitude to my friends back home. Our friendship has been a constant source of happiness in my life. Despite the physical distance, your presence is felt in every cherished moment.

This acknowledgement would be utterly incomplete without thanking my beloved parents and sister. Your belief in me has fueled my aspirations. I am deeply thankful for the values you've instilled, the lessons you've imparted, and the unwavering encouragement that has shaped who I am today. Your presence in my life is an enduring blessing for which I am truly grateful.

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, GOA CAMPUS

Abstract

Fisher Information and Traversability estimation-based Active SLAM for exploration in 3D environments

by Suchetan R S

Active visual SLAM finds a wide array of applications in GNSS-Denied sub-terrain environments and outdoor environments for ground robots. To achieve robust localization and mapping accuracy, it is imperative to incorporate the perception considerations in the goal selection and path planning towards the goal during an exploration mission. Through this work, we propose FIT-SLAM (Fisher Information and Traversability estimation-based Active SLAM), a new exploration method tailored for unmanned ground vehicles (UGVs) to explore 3D environments. This approach is devised with the dual objectives of sustaining an efficient exploration rate while optimizing SLAM accuracy. Initially, an estimation of a global traversability map is conducted, which accounts for the environmental constraints pertaining to traversability. Subsequently, we propose a goal candidate selection approach along with a path planning method towards this goal that takes into account the information provided by the landmarks used by the SLAM backend to achieve robust localization and successful path execution . The entire algorithm is tested and evaluated first in a simulated 3D world, followed by a real-world environment and is compared to pre-existing exploration methods. The results obtained during this evaluation demonstrate a significant increase in the exploration rate while effectively minimizing the localization covariance.

Contents

List of Figures

List of Tables

Abbreviations

SLAM	**S**imultaneous **L**ocalization **A**nd **M**apping
ASLAM	**A**ctive **S**imultaneous **L**ocalization **A**nd **M**apping
FIT-SLAM	**F**isher **I**nformation and **T**raversability-based **S**imultaneous **L**ocalization **A**nd **M**apping
UGV	**U**nmanned **G**round **V**ehicle
FOV	**F**ield **O**f **V**iew
GESTALT	**G**rid-based **E**stimation of **S**urface **T**raversability **A**pplied to **L**ocal **T**errain.
ROS	**R**obot **O**perating **S**ystem
OSRF	**O**pen **S**ource **R**obotics **F**oundation
CLI	**C**ommand **L**ine **I**nterface
URDF	**U**nified **R**obot **D**escription **F**ormat
FIM	**F**isher **I**nformation **M**atrix
GPS	**G**lobal **P**ositioning **S**ystem
GNSS	**G**lobal **N**avigation **S**atellite **S**ystem
GPU	**G**raphical **P**rocessing **U**nit

Dedicated to Maa, Pappa and Aru

Chapter 1

Introduction

This research was conducted at the Department of Electronics, Optronics and Signal Processing (DEOS) under the NAVIR²eS research team at Institut Supérieur de l'Aéronautique et de l'Espace (ISAE-SUPAERO), Toulouse, France from June 15, 2023 to December 15, 2023 for the completion of a Bachelors in Electrical and Electronics Engineering from Birla Institute of Technology and Science (BITS) Pilani, India.

1.1 Personal Motivation

My tenure at BITS Pilani in the field of Electronics, coupled with my active engagement in research-driven projects within the field of Robotics, has ignited a profound passion propelling me towards a comprehensive 6-month thesis. My fervor for the subject has been a unwavering companion, propelling me with determination throughout this half-year program. The hands-on experience I gained from constructing robots from the ground up during my degree has uniquely equipped me to successfully bring to fruition a novel project conceptualized at the start of my thesis.

Before starting the content of this document, I would like to wholeheartedly thank you for taking the time to go through my thesis. I highly value any feedback you may provide once this work becomes public. Your insights and comments will play a crucial role in my growth as a roboticist. Thank you in advance, and I hope you enjoy reading my work!

1.2 Background

When we think about robot moving autonomously, we picture an environment and the robots planning their own trajectories and performing their tasks without human intervention. For

this, the robots have to process the information received by their on-board as well as external sensors to perform a wide array of tasks. Majorly, this can divided into localization, mapping and path-planning. The problem of maintaining a good probabilistic estimate of the map as well as the location of the robot in the map was prevalent in the 1990s, when it came to notice that the two tasks were highly correlated with each other. SLAM refers, thereby, to the problem of incrementally building the map of an environment while at the same time locating the robot within it [1]. In case of Visual SLAM algorithms which mainly rely on the RGB / RGBD cameras as the main stream of sensor input, it is imperative to consider the perception requirement during the planning phase. This may involve finding potential areas of loop-closure or by observing landmarks containing high information. This results in the formation of a new problem, Active SLAM, a technique for which is addressed in this work.

1.3 Main contributions

The presented work proposes a novel exploration strategy for Active SLAM devised with the dual objectives of sustaining an efficient exploration rate while optimizing SLAM accuracy. The presented approach - denoted as *"FIT-SLAM"* - an acronym representing 'Fisher Information and Traversability-Based Active SLAM' aims to address the two objectives. In detail, the 3D space exploration task is conceptualized as a 2D traversability map, taking into account the constraints posed by the 3D terrain and obstacles. This transformation results in a notable decrease in the time taken to find the frontier clusters as well as path generation to the candidate goals. Transforming the 3D exploration space to a 2D traversability map also eliminates the computational burden posed by maintaining a 3D voxel map during exploration. Then, a goal selection approach taking into account the information gained upon reaching a given goal position, as well the information provided by the landmarks generated from SLAM during possible path execution towards this goal, is used to select the next destination point to be visited. To the best of the knowledge acquired during this research, this is the first work for UGVs exploring a 3D environment where the strategy not only accounts for the safety of the robot during exploration but also provides promising results for the reduction of the uncertainty related to both the pose of the robot as well as the map generated by a SLAM back end.

The main contributions of this work encompass two key aspects: (i) the proposition of frontier-based exploration and path planning strategies based on the use of a global traversability map and (ii) the proposition of a candidate goal selection approach which takes into account the information gained upon reaching the goal as well as the information provided by the landmarks generated from SLAM during the path execution to this goal.

1.4 Structure of this document

The presented Bachelor's thesis is structured into 6 chapters which is inclusive of this introduction chapter. This is followed by the bibliography, glossary and the appendices.

- Chapter 1: This chapter provides my personal motivation and a global picture of the work that will follow.
- In Chapter 2, a theoretical foundation that serves as the base for the future chapters along with the related work is established. A focus on state-of-the-art information acquisition techniques is a part of this chapter.
- Chapter 3 contains a detail of the GESTALT algorithm used to build the local and global traversability grids which is later used in 4.
- In Chapter 4, the Active SLAM algorithm that runs on the UGVs carrying out the exploration is presented and detailed along with a detail of the comparison methods.
- An extension of the framework presented in 4 for a multi-robot system is shown in 5.
- The results and comparison of the proposed algorithm in simulation as well as real-world experiments with two other exploration methods is shown in 6

Chapter 2

Theoretical Foundation

2.1 Active SLAM Problem

For mobile systems to be robustly localized, actively considering the perception requirement in the planning stage is essential. On-board visual sensing and computing permit systems to operate autonomously but bring additional constraints to motion planning algorithms. Specifically, the robot's motion impacts the information the visual sensors will capture and thus influences the performance of perception and localization algorithms. Therefore, the requirement of visual perception has to be taken into consideration in motion planning in order to improve localization accuracy. This problem is known as active vision or active perception. The central paradigm of active simultaneous localization and mapping (ASLAM) is to plan the sensor motion based on the information that can be attained and the covariance that can be maintained during the exploration mission.

The critical goal in UGV-based exploration is to achieve good long-term mission planning that has a competitive exploration rate while also maintaining good mapping and localization accuracy during the mission. The algorithms adeptly tackle the exploration-exploitation dilemma. That is, to strike a balance between exploring new parts of the environment and exploiting the already explored portion of the map.

2.2 Graph SLAM Approach

The proposed navigation solution uses a graph-based SLAM approach in which nodes represent robot 6D poses $\mathbf{x}$ or landmarks m_i of the map $\mathbf{M}$ poses, and edges in the map represent constraints between these poses. Such an approach makes use of a sliding window optimization that consists of finding the optimal state $\mathbf{X}^* = \{\mathbf{x}, \mathbf{M}\}$ that minimizes the summation of the

norm of the residuals, that are the errors of all factors $\mathbf{e}$ weighted by their respective covariances Σ_k, such as:

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \left(\sum_{k \in \mathcal{G}} \|\mathbf{e}_k(\mathbf{X})\|^2_{\Sigma_k} \right) \tag{2.1}$$

In the case of loop-closure detection, a similar optimization is performed but over the entire problem.

Note that for this work, RTAB-Map [2] is used as the SLAM backend. However, the proposed approach is not limited to RTAB-Map and can work with any other graph-based SLAM providing both key-frame poses and a map composed of landmarks.

2.3 The ROS 2 Navigation stack

For the navigation of our robot during exploration, we use the Robot Operating System (ROS) 2 navigation stack [3]. The navigation stack runs on the basis of a costmap layer implementation. The layers are user-programmable and a new plugin can be programmed to suit the needs. In this case, two plugins are programmed from scratch. One for the traversability estimation and the other for exploration. Apart from these two plugins, a dynamic obstacle layer can also be

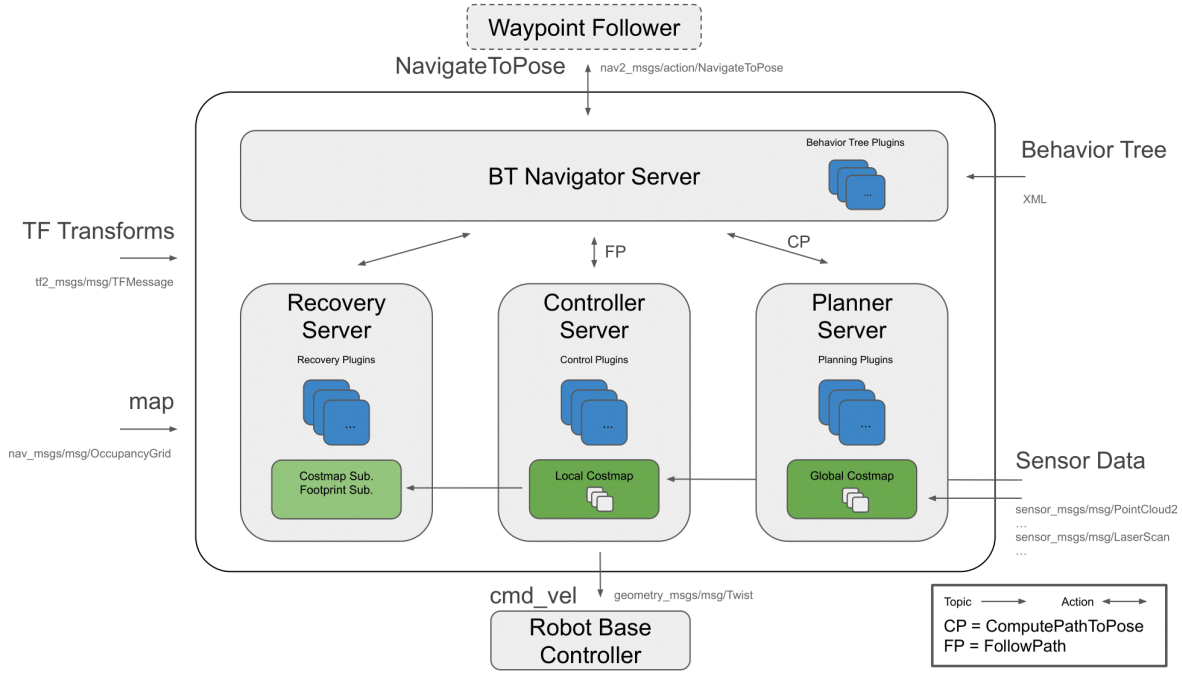

FIGURE 2.1: An overview of the ROS Navigation 2 framework showing the different servers and plugins that are in play for the robot navigation. The image is adapted from the work [3].

added to prevent the robot from colliding against fast moving dynamic obstacles which otherwise may not be mapped by the SLAM backend.

The ROS 2 navigation stack also provides integrations of the A* and the Dijkstra's algorithm for global path planning. A custom stop-and-go planner is implemented for the local planning to follow the waypoints provided by the global plan. This works well for our system since it is very light weight compared to the controller server of Nav2 and is designed for exploration in static scenes.

The ROS 2 navigation stack also allows for the use of custom controller plugins for the control of the robot. The Scout V2 is used as the robotic platform for our simulations which runs a regular differential drive controller.

The Navigation 2 stack provides a Gmapping SLAM backend. However, since the exploration we desire is in 3D, we replace it for a 3D SLAM Algorithm. The use of lifecycle nodes in the Nav2 stack enables the nodes to transition between states such as unconfigured, inactive, active and destroyed. This provides several key advantages. Most important of the advantages it provides being resource management. The further working and communication details of the Navigation stack is detailed in Appendix A.

2.4 Related works

B. Yamuchi [4] introduced the concept of *Frontiers*, a set of points which acts as the transition region between the areas of a map that are already explored from those not yet visited. Since this work, where a greedy *frontier-based* exploration was performed, several contributions and improvements have been made to the exploration strategy. In the realm of goal determination for exploration, there exists a variety of methodologies. The frontier-based approach is still popular and can be found in recent works such as [5]. An alternative method to select possible goals is proposed by Umari et al. [6], where they detect the frontier on a 2D occupancy grid map using rapidly-exploring random trees (RRTs). The use of this strategy concatenated with computer vision algorithms can be found in works such as [7]. These approaches, however are limited to the robot exploring a 2D planar environment, which vastly reduces the capabilities of a UGV. Some works in the past focus on 3D UGV exploration, such as [8]–[10]. These works showcase promising results in terms of the exploration rate of the proposed algorithm. However, they do not account for the localization accuracy estimated by the SLAM backend.

Unfortunately, high uncertainty in the robot state could lead to a significantly unreliable map. In [11], Bourgault et al. addressed this issue by incorporating a utility function that has a trade-off between the information from the SLAM backend and the reduction of the map entropy. Stachniss et al. [12] employ a Rao-Blackwellized particle filter (RBPF) for representing both

the robot poses and the map. Through this, they can determine the best possible action by estimating the entropy of the particle filter. The use of Kullback-Leibler divergence as a metric for the measure of information can be found in [13]. Different optimality criterion from Theory of Optimal Design (TOED) [14] is often used to quantify the uncertainity. For example, Carillo et al. [15] show that the D-optimality can be used as an information metric and is comparable to the A-optimality criterion. The recent approaches, especially the ones that incorporate graph-based optimization in SLAM have proven to have better long-term prediction than the RBPF approaches, which ultimately suffer from particle depletion as the size of the environment grows. [16]

2.5 Chapter summary

This chapter forms the theoretical foundation of the Graph SLAM approach and shows how it is extended to an Active SLAM problem. This chapter also explains how the navigation stack is integrated into the system that will be discussed further and also contains some very key insights to the already existing work on the Active SLAM problem.
The further chapters contain a more in depth study on the traversability estimator and the information-based planning and goal selection approach.

Chapter 3

Traversability Estimation

3.1 Estimation of Local Traversability Map

The 3D Mapping strategy begins with an estimation of a local traversability map. This solves two key problems. (i) Removes the necessity of constructing a 3D octomap unless necessary. (ii) Fast traversability computation which can be fine tuned to robot-specific parameters such as ground clearance, maximum traversal slope etc. enabling 2D planning in a 3D exploration environment. This traversability map is programmed with the help of the *grid_map* library [17].

The local traversability map is estimated in the *base_link* frame of the robot. Provided the dimensions and resolution of the local map, the *grid_map* library is used to create hazard layers for the traversability.

3.2 Grid-based Estimation of Surface Traversability Applied to Local Terrain (GESTALT) algorithm. [18]

In this work, a geometry-based traversability estimator is used similar to the works in [19]–[21]. For each cell typically the size of the UGV wheel in the gridmap, the first and second order moment statistics are collected for all the points in the pointcloud. The following statistics are stored as the grid meta data regardless of the number of points in the grid. $\sum X,\ \sum Y,\ \sum Z,\ \sum X^2,\ \sum Y^2,\ \sum Z^2,\ \sum X \cdot Y,\ \sum X \cdot Z,\ \sum Y \cdot Z.$

3.2.1 Estimation of the best-fit plane

Given a user-defined number of neighbours to consider, the best-fit plane for each grid is computed. The moment statistics stored for the grid of concern and the neighbouring cells are

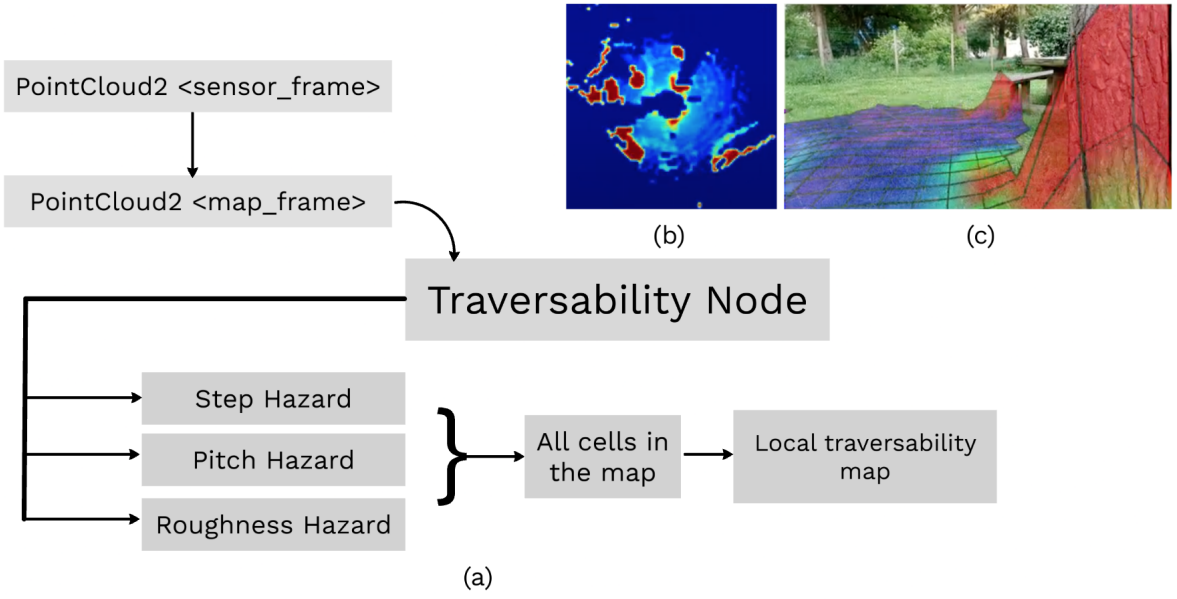

FIGURE 3.1: (a) The process behind computing the local traversability map. (b) The Local Traversability occupancy grid converted to an RGB Image. (c) The local traversability gridmap projected onto the real-world image for a comparision

merged to provide a combined statistic along with the combined total number of points N_{pts}. The normal η of best-fit plane is computed in the following manner:

$$\sigma = \sum_{allgrids} \begin{bmatrix} \sum X & \sum Y & \sum Z \end{bmatrix} / \ N_{pts} \tag{3.1}$$

$$\widehat{\sigma} = \sum_{allgrids} \begin{bmatrix} \sum X^2 & \sum X \cdot Y & \sum X \cdot Z \\ \sum X \cdot Y & \sum Y^2 & \sum Y \cdot Z \\ \sum X \cdot Z & \sum Y \cdot Z & \sum Z^2 \end{bmatrix} / \ N_{pts} \tag{3.2}$$

$$A = \widehat{\sigma} - (\sigma^T \cdot \sigma) \tag{3.3}$$

$$\vec{\eta} = eig(A) \tag{3.4}$$

Here, A represents the covariance matrix with of the concerned grid along with the neighbouring grids. The surface normal is obtained with the Eigen Vector decomposition of the covariance matrix A.

For computing the hazards, two important terms, that is, *ground_clearance* and *max_pitch* are required. The ground clearance is defined as the distance between the *base_footprint* frame and the *base_link* frame of the robot. It is the maximum height of an obstacle that the robot can pass over safely. The *max_pitch* is the highest slope angle that the robot can traverse without significant slipping and at a consistent speed.

3.2.2 Step Hazard Computation

The step hazard serves as a traversability metric to estimate if the robot can traverse a given grid with the ground clearance as a required parameter. Consider a set $G = g_1, g_2, g_3 \dots g_n$ which is the concerned grid along with the neighbours. The highest point in the Z direction in the local frame of the robot is denoted as Z_M and the lowest point in the Z direction in the same frame is denoted as Z_m. Therefore, the step hazard H_s for each grid can be computed as:

$$H_s = \frac{(Z_M - Z_m)}{ground\_clearance}; \tag{3.5}$$

3.2.3 Pitch Hazard Computation

The pitch hazard serves as a traversability metric to estimate if the robot can traverse a given grid by estimating its inclination with respect to the gravity aligned vector $\vec{G}$. This helps prevent the risk of traversing very steep slopes which otherwise may not be detected as an obstacle by a regular mapping software. The pitch hazard H_p is computed in the following manner:

$$H_p = \frac{|cos^{-1}(\vec{\eta} \cdot \vec{G})|}{max\_pitch} \tag{3.6}$$

3.2.4 Roughness Hazard Computation

The roughness hazard metric helps the mapping by estimating the variance of all the centers to the estimated plane. The centers here are the means of the grid computed in (3.1). To compute this, we first estimate the distance d to all the centers in the grid by:

$$d = \sigma \cdot \vec{\eta} \tag{3.7}$$

The roughness hazard H_r can therefore be estimated by the following.

$$H_r = \sum_{allgrids} \frac{(\sigma \cdot \vec{\eta} - d)^2}{ground\_clearance} \tag{3.8}$$

3.2.5 Hazard for the grid

Using the hazard values computed for the step, roughness and pitch, we can determine the hazard for the grid. This is given by taking into account the maximum possible risk showing a conservative approach to traversability risk assessment.

$$H_{grid} = \arg\max(Hr, Hp, Hs) \tag{3.9}$$

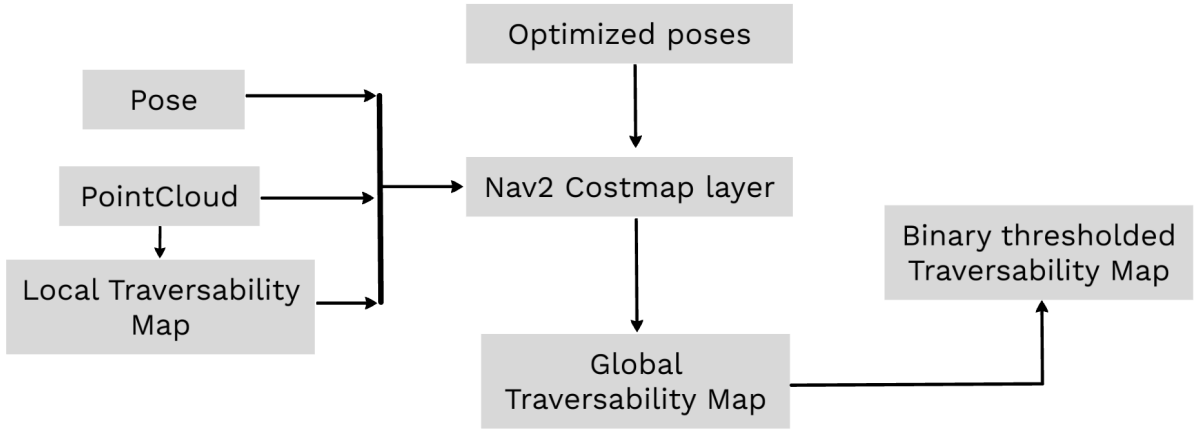

FIGURE 3.2: The communication workflow of how the global traversability map is generated.

3.3 Estimation of the Global Traversability Map

In order to carry out a successful exploration task, a global map of the environment must be maintained at all times. This can be achieved by linking the poses given by the SLAM backend to the Local Traversability Map having the closest timestamp to the pose. If a higher accuracy needs to be achieved, the poses as well as the pointcloud can be given by the SLAM backend to avoid the delay in the timestamps between the two. To achieve a smooth frontier detection and clustering, the global map is thresholded to output a Binary traversability map similar to an occupancy grid produced by a regular SLAM algorithm.

3.3.1 Effect of pose optimization on the map

The SLAM backend used in the system is a Graph-SLAM based approach which uses sliding window optimization. This results in the poses being updated and optimized as the robot moves through the environment. This makes it imperative to maintain an optimized traversability map as the Local maps that are merged are inherently linked to the poses given by the SLAM. Therefore, a functionality of providing the updated most optimized poses is also added to the Traversability costmap plugin. Whenever an updated pose is received, the corresponding pointcloud is queried from the pointcloud database and the local traversability map for that pose is regenerated thus maintaining an optimized map at all times.

The results of a real-world experiment conducted using an Ouster-128 as the primary LiDAR and a bi-monocular fisheye camera based visual SLAM algorithm [22] running on a Scout-V2 at CNES, Toulouse, is shown in Fig. 3.3.

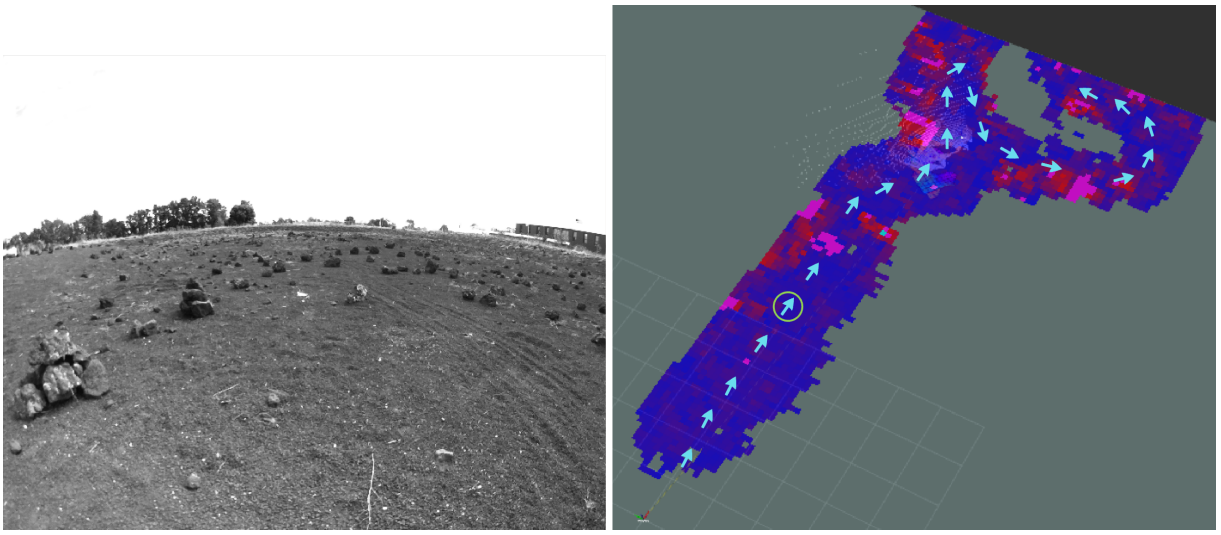

FIGURE 3.3: An example of a global traversability map generated at the Marsyard at CNES, Toulouse, France. An actual image of the Marsyard (left) and the generated map (right). The blue arrows show the robot poses and the circled blue arrow represents the pose at which the image was captured.

3.3.2 ROS 2 Costmap plugin implementation.

The entire traversability layer is implemented as a Navigation 2 costmap plugin layer for ease of use with the Navigation stack or other path planning software. The current navigation stack comprises mainly of two layers. (i) The static layer: Traversability map (ii) The inflation layer: This marks the obstacles in the traversability map with an inflation boundary to prevent robot movements with close proximity to not traversable regions. Currently, the system does not have the capability to deal with dynamic moving obstacles in the exploration environment. In future, a third obstacle layer maybe added for fast traversability estimation to be used with the local planner for dynamic obstacle avoidance.

3.4 Chapter summary

This chapter talks about the GESTALT algorithm and how it has been extended to a global planning framework for any graph-based SLAM approach. A method to optimize the map if the optimized poses are received is also discussed along with the integration of the global map with the ROS 2 navigation stack. This is followed by the presentation of the result of a real-world experiment of a global traversability map. In the next chapter, we discuss on how we can use this global traversability map to solve our Active SLAM problem for 3D environments.

Chapter 4

The Active SLAM Algorithm

4.1 Introduction to this chapter

This section contains the full details of the Active SLAM framework that has been built. A proposal of a complete ASLAM framework to deal with the active exploration problem of an unknown and unstructured environment is given. The proposed solution is based on a graph-based SLAM approach for localization and a traversability estimator for path planning risk assessment. Finally, an analysis of the Fisher information calculated for each trajectory is used to select the best path. A minimalistic figure showing the overall picture of the framework is in Fig. 4.1.

4.2 Frontier detection and clustering

4.2.1 Frontier detection

Frontiers are regions in the map that act as a boundary between explored and unexplored regions. The proposed exploration algorithm begins with plotting an exploration boundary within which the frontiers are searched. A conventional frontier search algorithm similar to the one proposed in [4] is used. In this algorithm, the cells in the costmap cells are treated as a queue which are fed to a bread-first-search module which returns the clustered frontier cells. The costmap cells are searched using a 4-neighbour approach and clustered using an 8-neighbour approach.
The proposed BFS module is shown in the Algorithm 1 which takes in the costmap layer and the current robot position as input and outputs a list of clustered frontier cells.

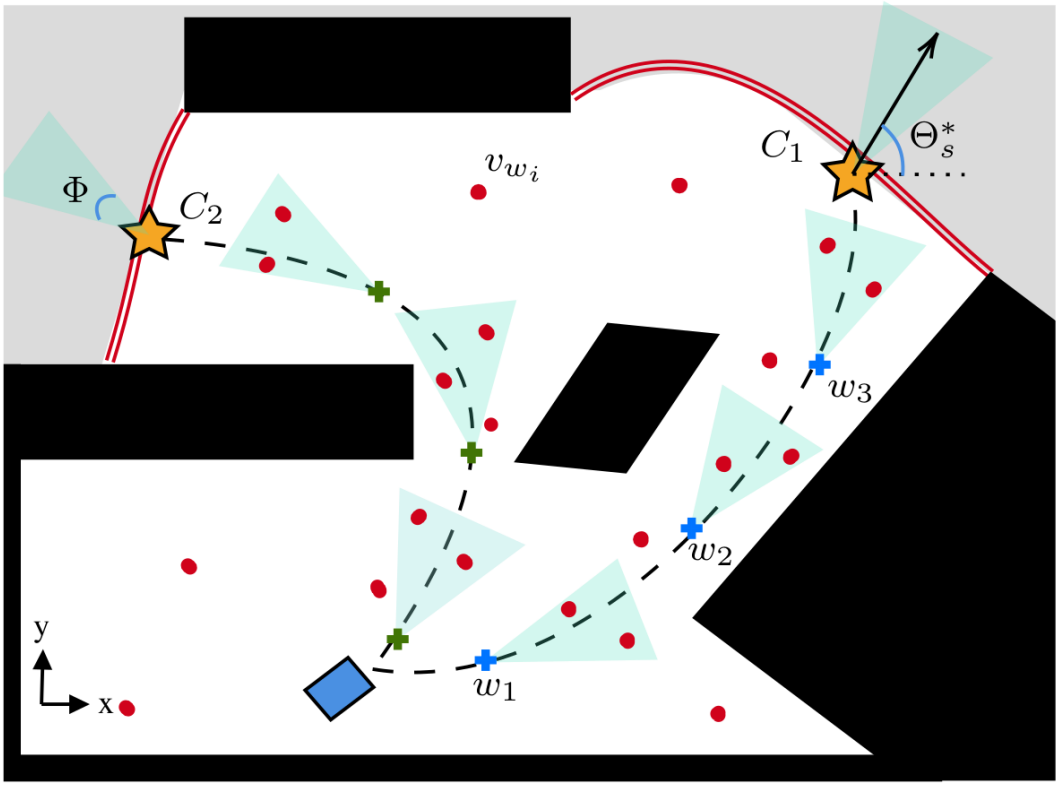

FIGURE 4.1: Overview of the proposed framework: A 2D thresholded traversability map is analyzed to extract frontiers goals (stars). Each goal candidate C_i is ranked based on the information computed for the planned path. Such information is linked to the observed landmarks (red dots) in the camera field-of-view (FOV) Φ (blue area) but also the maximal reduction in entropy after reaching the goal with robot orientation Θ_s^*. This allows to ensure good localizability while exploring new areas, resulting in a more accurate mapping.

4.2.2 Frontier clustering and candidate goal determination

In this case, the conventional frontier search algorithm is modified to perform frontier clustering constrained to a maximum size. All linked frontier cells are treated together as one cluster, and the frontier point with the median index of the cluster is chosen as a candidate goal C_i. The path to reach each candidate goal is then generated. The 2D global traversability map coupled with the A* path planning algorithm from the ROS 2 navigation stack is used for this purpose. In cases where it is impossible to calculate a path to the goal or when the goal has been previously identified as a frontier point in an earlier iteration, it is designated for exclusion in a blacklist, indicating that the goal is currently unreachable. If there are no frontiers available in the map or in the area of interest, the exploration mission is considered to be a success and the mission ends.

Algorithm 1 Frontier search algorithm on the costmap plugin layer

Require: Costmap Layer, Robot current position(*pos*)

```
// Iterate over 4-connected neighbourhood
bfs1.push(pos)
idx1 = bfs1.pop()
while !bfs1.empty() do
  for nbr1 ∈ nhood4(idx1) do
    if nbr1 ∈ free OR unvisited then
      bfs1.push(nbr1)
    else if isNewFrontierCell(nbr1) then
      // Build new frontier with all connected cells
      bfs2.push(nbr1)
      while !bfs2.empty() do
        idx2 = bfs2.pop()
        // Iterate over 8-connected neighbourhood
        for nbr2 ∈ nhood8(idx2) do
          if isNewFrontierCell(nbr2) then
            bfs.push(nbr2)
            frontier.push(nbr2)
          end if
        end for
      end while
      // Cluster the frontier vector
      frontier_list.push(cluster(frontier))
    end if
  end for
end while
```

4.3 Utility computation and candidate goal selection

The utility of any candidate goal C_i is processed at two levels: The first-level (u_1) is based on the computation of the total distance of the path from the current robot position to the candidate frontier and on the information gained upon reaching the said frontier. The second level (u_2) is estimated by the information gained along the path.

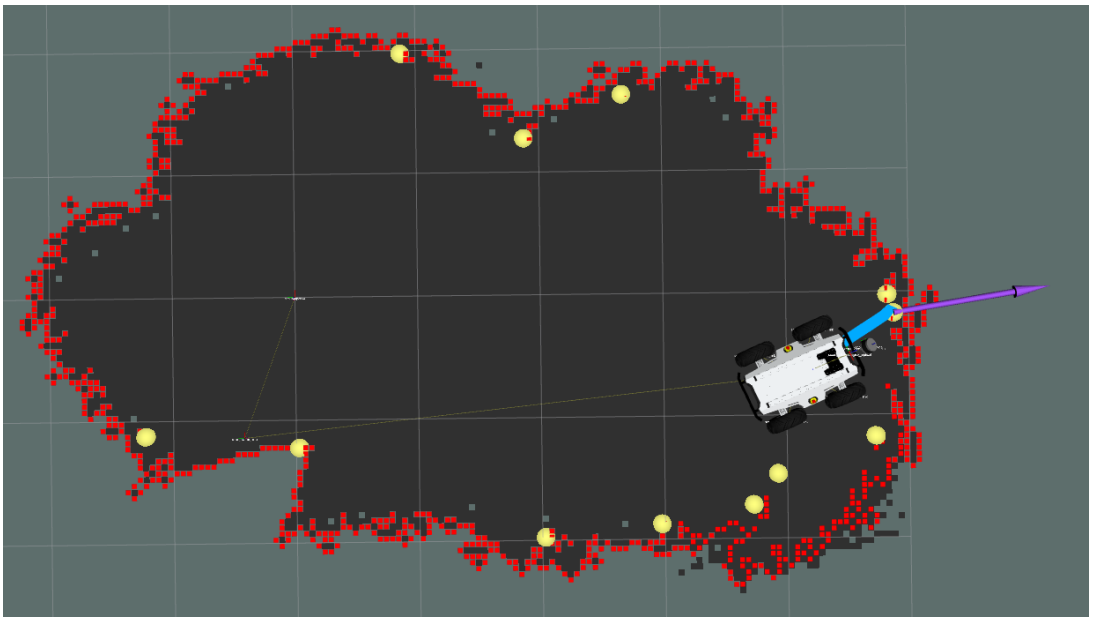

FIGURE 4.2: This figure shows the detected frontier cells (red) and clustered frontier cells (yellow). The chosen goal candidate and the orientation is marked with a purple arrow.

4.3.1 First-level utility computation (u_1)

Let $C_x = \{C_0, C_1, \ldots, C_n\}$ be a set of candidate goals. For each candidate $C_i \in C_x$, we have an associated path p_i with a length ρ_i and a measure of information gained upon reaching the candidate goal ΔE_i, which represents the change in map entropy upon observing the unknown cells after reaching the candidate goal.

More precisely, to compute the possible information gain upon reaching the candidate goal, a conventional ray tracing algorithm is used to get the set of potentially observable cells. Initially, we first determine the optimal arrival sensor orientation Θ_s^* that maximizes the information gain. In practice, to control the spatial density of the ray-tracing and expedite the process, we approximate ΔE_i using a finite number of rays separated by $\Delta\theta$.

Let $\Theta = \{0, \Delta\theta, 2\Delta\theta, \cdots, 2\pi\}$ be the set of discretized ray directions and Φ is the camera field-of-view, we have the optimal arrival sensor orientation Θ_s^* given by:

$$\Theta_s^* = \underset{\Theta_s \in \Theta}{\arg\max} \left(\sum_{\Theta_s - \frac{\Phi}{2}}^{\Theta_s + \frac{\Phi}{2}} \Delta E_G^{\Theta_s} \right) \tag{4.1}$$

where, $G = \{c_1, c_2, \cdots, c_n\}$ represents a 2D occupancy grid composed of n cells. $\Delta E_G^{\Theta_s}$ is the information gained along the ray in the direction Θ_s. The information gain can be calculated as the change in the occupancy grid entropy before and after observing the cells along the ray in the direction Θ_s.

Following [23], the entropy $E_G^{\Theta_s}$ of a given occupancy grid-map G is computed based on the *Shannon entropy* as a measure of map uncertainty. It is given by:

$$E_G^{\Theta_s} = \sum_{i=0}^{n} E^{\Theta_s}[c_i] = -\sum_{i=0}^{n} \left(p(c_i) \cdot \log_2(p(c_i)) + (1 - p(c_i)) \cdot \log_2(1 - p(c_i))\right) \tag{4.2}$$

where, c_i represents a cell in G, and $p(c_i)$ represents the occupancy probability of the cell c_i. If the cell c_i is unknown then $p(c_i) = 0.5$. It is manifest that such a computation requires the estimation of probabilities of all cells that can be observed along the ray. The farther a ray travels into unknown space, the more likely it is to be obstructed by an obstacle. Thus, as proposed in [24], the observability of the cell is dependent on the previous cells traversed along the ray, such as:

$$p(x_r|x_{r-1}) = \begin{cases} 1 & \text{if ray intersects an occupied cell} \\ \gamma^N & \text{otherwise} \end{cases} \tag{4.3}$$

where $p(x_r|x_{r-1})$ is the observability of a cell lying along the ray composed of r cells, γ is the degradation parameter which controls how fast the probability degrades along the ray, N is the previous number of cells traversed by the ray. Given the observability of a cell, we can estimate the posterior occupancy probability of the cell c_r as:

$$p(c_r) = \frac{1 + p(x_r|x_{r-1})}{2} \tag{4.4}$$

Finally, ΔE_i, which is the information gained upon reaching the candidate goal C_i can be processed with the optimal arrival sensor orientation Θ_s^* by:

$$\Delta E_i = \sum_{\Theta_s^* - \frac{\Phi}{2}}^{\Theta_s^* + \frac{\Phi}{2}} \Delta E_G^{\Theta_s^*} \tag{4.5}$$

At this step, the first-level utility (u_1) can be computed for each candidate goal C_i as a trade-off between the path length to be traversed by the robot ρ_i and the information gained ΔE_i upon reaching the goal:

$$u_1(C_i) = \alpha \; N_{\rho^{-1}} \; \rho_i^{-1} + (1 - \; \alpha) \; N_{\Delta E} \; \Delta E_i \tag{4.6}$$

where $\alpha \in [0, 1]$ (resp. $(1 - \alpha)$) represents the weight assigned to ρ_i (resp. ΔE_i) and $N_{\rho^{-1}}$, $N_{\Delta E}$ are normalization factors.

By ranking the set of candidate goals C_x based on their corresponding first-level utility (u_1) values, we can determine the N most promising candidate goals in terms of distance to the goal and overall map entropy reduction.

However, this selection criterion does not take into account the information acquired during the travel to the intended goal. In instances where the information gathered during the travel phase is insufficient, the robot runs the risk of getting lost in the map and becoming disoriented. This could lead to a high localization uncertainty, which inherently could lead to a poor map accuracy. The inclusion of the second utility level mitigates this limitation.

4.3.2 Second level utility computation (u_2)

We propose to add to the previous utility function the information gathered during path traversal. The objective is to maximize the robot's overall pose accuracy to the utmost degree and as a consequence, improve the map quality.

This level of utility computation will be used to determine the most optimal candidate goal by estimating the information gathered during the path execution. Given a path p_i to a candidate goal C_i, we first sample the trajectory with a sampling distance equal to the max-depth of the camera FOV to obtain a set of waypoints $W_i = \{w_1, w_2, \ldots, w_n\}$. For each $w_k \in W_i$, we compute the information of all the landmarks from the map that lie within the FOV.

To do so, the Fisher Information Matrix (FIM) is used. FIM represents the minimum reachable covariance of an unbiased estimator [25] and allows to quantify the estimation uncertainty. In the proposed approach, we use the bearing vector representation of the 3D landmark. Given a 3D landmark that lies in a voxel v_{w_i} in the world frame with a covariance Q_i, the observation function can be modeled using the bearing vector b_i of the detection v_{w_i} such as:

$$b_i = \frac{v_{c_i}}{\|v_{c_i}\|_2} \quad \text{with} \quad v_{c_i} = T_{cw} v_{w_i} \tag{4.7}$$

where v_{c_i} is the i^{th} voxel in the camera frame, T_{cw} the affine transform matrix from world to camera frame.

The Fisher information matrix I_i corresponding to the considered landmark lying in the voxel v_{w_i} can be derived as:

$$I_i = J_i \; Q_i \; J_i^T \tag{4.8}$$

With J_i the jacobian of the observation model (4.7) given by:

$$J_i = \frac{\partial b_i}{\partial T_{wc}} = \frac{\partial b_i}{\partial v_{c_i}} \frac{\partial v_{c_i}}{\partial T_{wc}} \tag{4.9}$$

where the elements in Eq. (4.9) are given using the Special Euclidean group SE(3) [26] by:

$$\frac{\partial b_i}{\partial v_{c_i}} = \frac{1}{\|v_{c_i}\|_2} I_3 - \frac{v_{c_i} (v_{c_i})^T}{(\|v_{c_i}\|_2)^3} \tag{4.10}$$

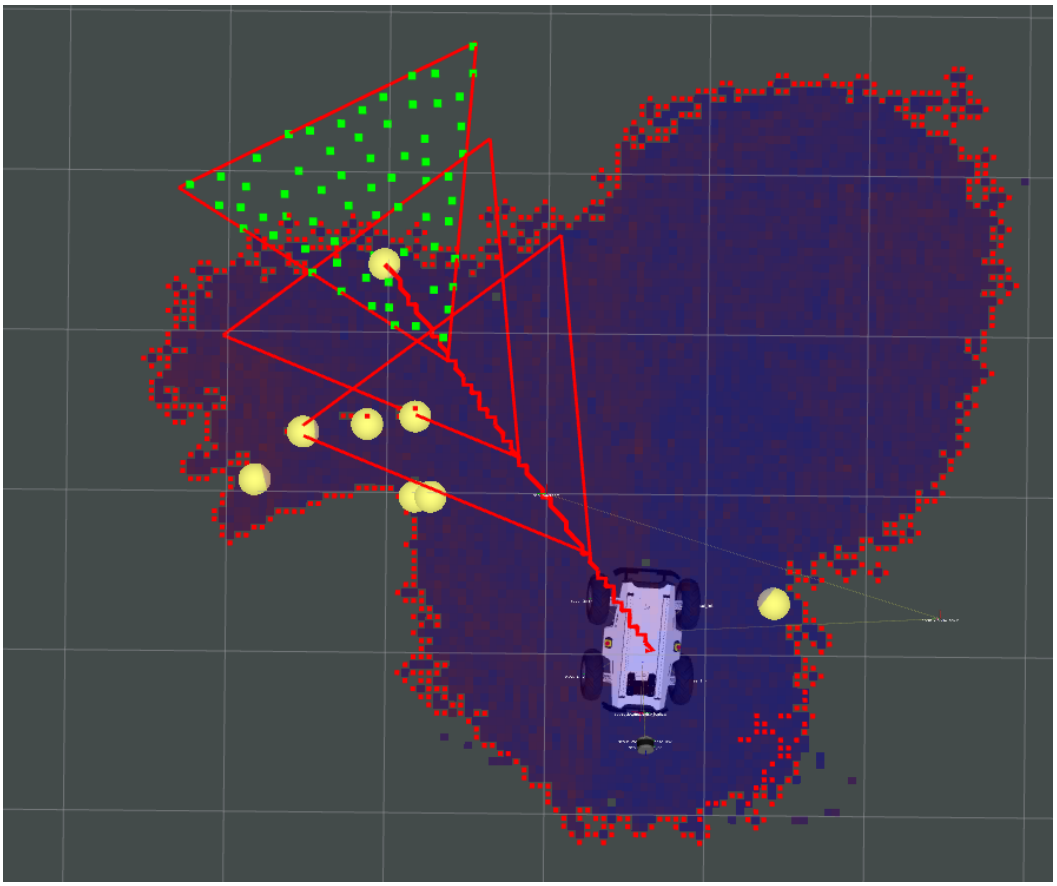

FIGURE 4.3: A representation of the u_2 computation stage. All the possible FOVs in the path are shown as red triangles. The landmarks from the SLAM backend in the final prospective camera FOV are shown as green tiles.

$$\frac{\partial v_{c_i}}{\partial T_{wc}} = R_{cw} \left[-I_3, [v_{w_i}]_\times \right] \tag{4.11}$$

where R_{cw} is the rotation from world to camera frame and $[v]_\times$ is the cross-matrix of vector v.

Storage of the FIMs for all the voxels arguably consumes a lot of memory. A common way to bypass this memory usage is to use the theory of optimal experimental design (TOED) [14], which utilizes the T-opt optimality criterion, i.e., the trace of the FIM can be used to convert FIM to a scalar value significantly reducing the memory usage.

Therefore, we can estimate the information of the path p_i by summing over all waypoints $w_k \in W_i$.

$$I_{p_i} = N_I \sum_{w_k \in w_x} I_{w_k} = N_I \sum_{w_k \in w_x} \sum_{v_{w_i} \in v_{w_k}} I_i \tag{4.12}$$

where I_{p_i} is the information of the path, I_i is the information of the voxel v_{w_i} and N_I is the normalization factor. However, it is important to note that even though the memory usage is

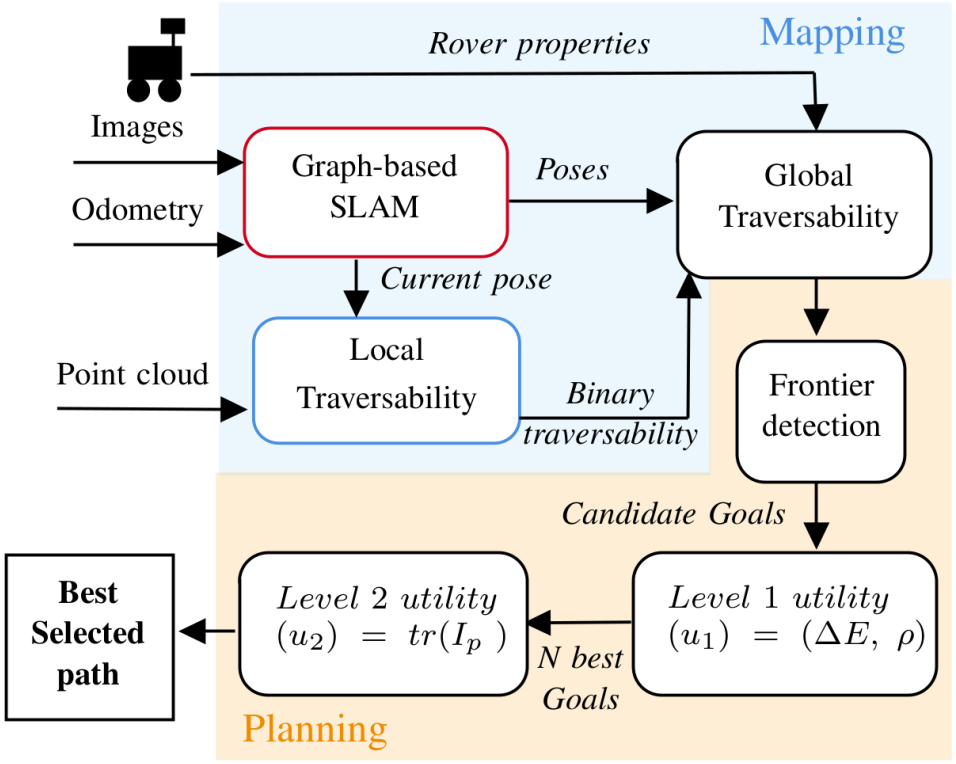

FIGURE 4.4: A global overview of the order of processes in the proposed work.

significantly reduced due to the inclusion of T-opt criteria, the computation of the FIM, even with the incorporation of voxelization is a computationally intensive task.

Following the computation of information along the path, we are able to compute the second level utility (u_2) for a candidate goal C_i, such as:

$$u_2(C_i) = \beta \cdot u_1 + \ (1 - \beta) I_{p_i} \tag{4.13}$$

where $\beta \in [0, 1]$ is a weighting parameter between u_1 and I_{p_i}. We compute this information solely for the N best candidate goals ranked after the computation of u_1. Let C_x^* represents the N best candidate goals in C_x, the best candidate goal C_{best} is simply the one with the biggest utility value (u_2), such as:

$$C_{best} = \underset{C_i \in C_x^*}{\arg\max} \left(\beta \cdot u_1 + \ (1 - \beta) I_{p_i}\right) \tag{4.14}$$

Finally, the path selected based on u_2 is the one that minimizes the localization and map uncertainty as it has the most informative landmarks observed during traversal while also taking into account the distance of the goal and information gained upon reaching the said goal.

4.4 The global overview and chapter summary

A global picture of how the traversability estimation, global traversability mapping, frontier detection and the goal selection along with the clear differentiation between the mapping and planning phase of the exploration mission in the proposed work is shown in Fig. 4.4
This chapter forms the core research done during my thesis. It gives the in depth explanation on various methods used for the exploration including frontier detection, clustering and goal selection using the reduction in map entropy and fisher information matrices.

In the upcoming chapter, we discuss the extension of this algorithm to a multi-robot approach and in the chapter following that, we discuss our results for the experiments conducted.

Chapter 5

Multi-Robot Approach

5.1 Introduction to this chapter

The algorithm explained in Chapter 4 can be extended to a multi-robot approach. In this case, there are two instances of the SLAM backend and a common shared map synced with the GPS. A centralized goal allocation server is used to assign the best goals to the robots in play.

5.2 The Hungarian algorithm for goal allocation

One of the initial algorithms proposed for exploration is the Hungarian algorithm. [27] presented an algorithm to choose the best combination of assignment for a fixed set of machines. That is, given a n x n cost matrix, which is a mapping of the cost of all the jobs to all the machines, the method is able to find the most optimal solution.
However, this algorithm has a very high complexity. That is, $O(n^3)$. For this cost matrix, the costs are constructed using the FIT-SLAM algorithm described in Chapter. 4.

In our case, it will almost always be the case that the number of possible goal locations in the common shared map is much higher compared to the number of robots exploring the environment. This mandates us to employ a set of *dummy robots* in order to fill the cost matrix correctly. For these robots, the cost to all the candidate goal locations is infinity.

$$H_m = \begin{array}{c} \\ R_1 \\ R_2 \\ R_3 \\ \vdots \\ R_n \end{array} \begin{array}{c} \begin{array}{ccccc} G_1 & G_2 & G_3 & \dots & G_n \end{array} \\ \begin{pmatrix} c_{11} & c_{12} & c_{13} & \dots & c_{1n} \\ c_{21} & c_{22} & c_{23} & \dots & c_{2n} \\ c_{31} & c_{32} & c_{33} & \dots & c_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & \dots & c_{nn} \end{pmatrix} \end{array} \tag{5.1}$$

An example of a hungarian matrix is shown in Eqn. 5.1. The columns represent the candidate robots and the rows represent the candidate goals with the matrix filled with the corresponding costs.

5.3 Goal allocation and Map merge server

Instead of the robots taking the decisions independently, there are two servers which handle the merging of the maps and the goal allocation server which runs the Hungarian algorithm in the back end. The Fig. 5.1 represents how the communication takes place between different entities in the system. The keywords which follow the "/" represent the namespace and the ROS 2 topic names.

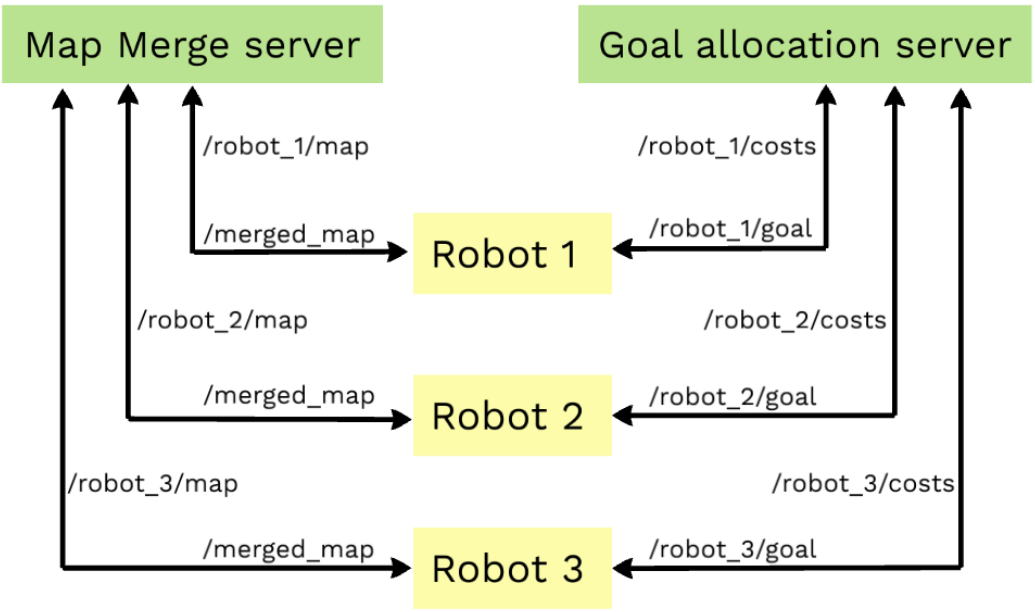

FIGURE 5.1: A connected diagram representing the communication links between the robots and the servers.

Chapter 6

Results

6.1 Introduction to this chapter

In this section of the report we explicit the results of our simulation as well as real-world experiments to validate our approach of exploration.

6.2 Platform and details of the experiment

We validated our approach by comparing two metrics: (i) the percentage of unexplored map with respect to time and (ii) the evolution of the localization covariance (the marginal error obtained

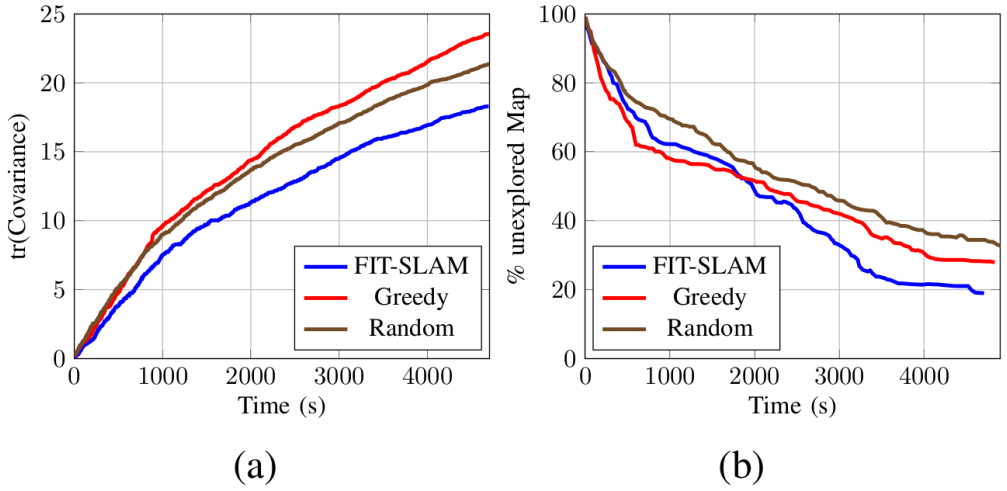

FIGURE 6.1: Evaluation of the proposed approach for the experiment conducted in simulation
(a) Evolution of the trace of the robot state covariance over time.
(b) Evolution of the exploration rate

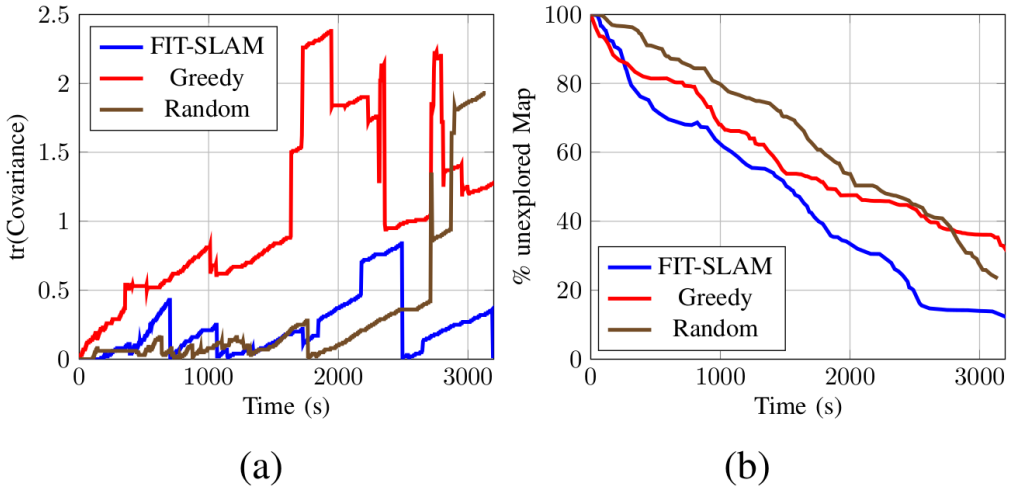

FIGURE 6.2: Evaluation of the proposed approach for the real- world experiment. (a) Evolution of the trace of the robot state covariance over time. (b) Evolution of the exploration rate. The jumps in the covariance trace correspond to the loop closure detections.

after graph optimization from the SLAM) with respect to time during exploration. The entirety of our system is programmed using the ROS 2 framework and tested in a 3D unstructured simulated environment on Gazebo running on a standard computer. The simulation worlds are the standard inspection world and the MarsYard2020 world from the European Rover Challenge (ERC).

Our real-world experiment was conducted in a planar environment with the Nvidia Jetson AGX Xavier (CPU: 8 Core @ 2.26 GHz, RAM: 32 GB, GPU: unused) as our on-board computer and the LeoRover as our robotic platform. The Intel Realsense D435 and the Velodyne VLP-16 LiDAR was used as our primary sensors. The rover used is shown in Fig. 6.3. The parameters used in the experiments are shown in table 6.1.

TABLE 6.1: Experimental Parameters

Parameter	Value
α	0.35
β	0.4
$\Delta\theta$	8.5°
N	7
Traversability Map Resolution	0.05 m
Voxel Size	0.25 m

The plot in Fig. 6.1 (a) shows that our approach has the best accuracy. In Fig. 6.2 (a), several drops in covariance can be observed. These indicate the points of loop-closure. The high number of loop closures detected in our method is a direct consequence of using the information provided

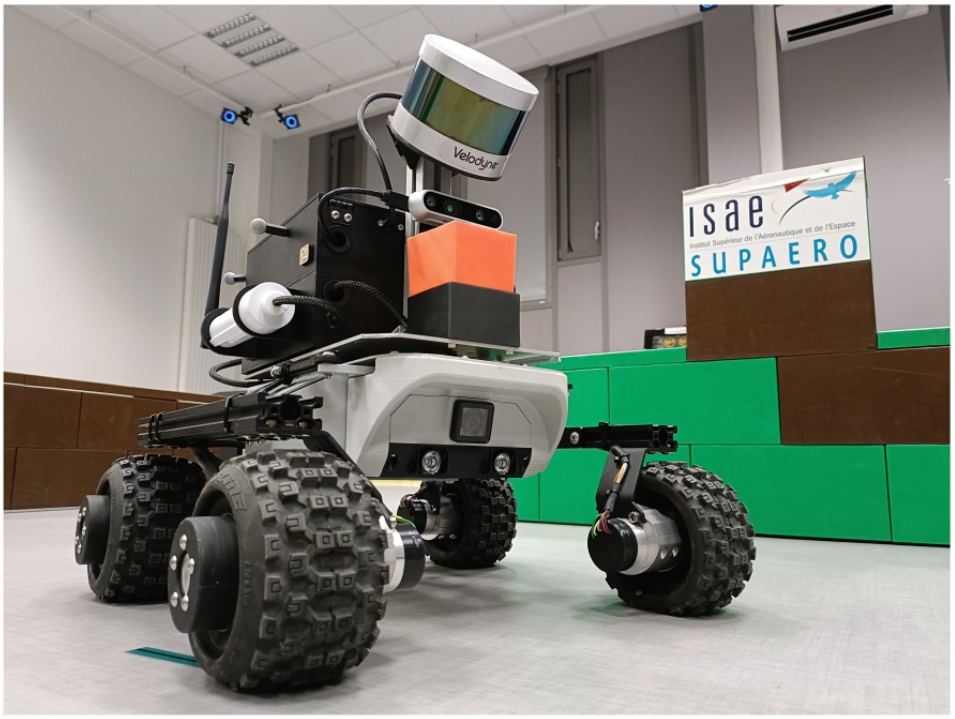

FIGURE 6.3: Our robotic platform equipped with the sensors (Depth Camera, 3D LiDAR, IMU and wheel odometry) required for the algorithm.

by the landmarks of the map in the goal selection. Regarding exploration speed results, illustrated in Fig. 6.1 (b) and 6.2 (b), it is interesting to observe that the greedy frontier approach maps the environment very quickly in the initial phase. However, during long-term planning, the greedy frontier exploration is stuck exploring frontiers that yield very low information gain. This result suggests that our first-level utility would play a key role in exploration speed improvement. The results for the real-world experiment are consistent with the results of the simulation.

The real-world tests for the exploration were carried out at the Volière, Department of Aerospace Vehicles Design and Control (DCAS) which hosts a test area for drones and 2D environment for ground robots. The area is equipped with a state-of-the-art infrared motion capture system which provide *cm* level accuracy for the ground truth of the robots. A birds eye view of the area is shown in Fig. 6.4 and Fig. 6.5.

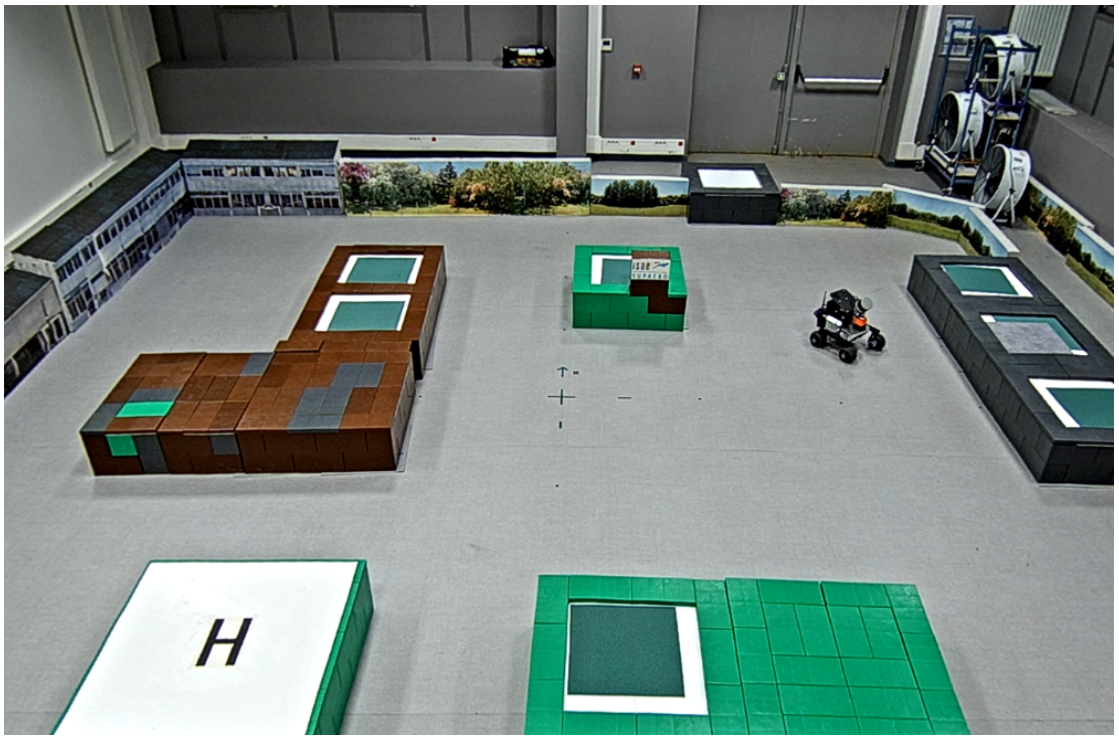

FIGURE 6.4: A birds-eye view of the Volière, DCAS, ISAE.

FIGURE 6.5: The Volière from a different angle showing the motion capture sensors (blue).

Chapter 7

Conclusion and future perspectives

7.1 Thesis conclusion

Thanks a lot for reading my Bachelor's thesis and if you have stuck around till the end, I cannot thank you enough! I have been given this amazing opportunity to present an in-depth report through this thesis for which I am really grateful.

This report started with an introduction, followed by a discussion of already existing works which serves as a basis for the traversability and Active SLAM explained in Chapter 3 and Chapter 4. This work proposes a novel Active-SLAM approach to explore a 3D unstructured environment based on a traversability map. The solution uses both the Shannon entropy to measure the amount of information that could be gained by mapping new areas and the Fisher information matrix as an information metric to estimate the information gained during path execution by observing the known and mapped landmarks. The entire approach has been tested in a 3D setting on simulation and we showed substantial improvements in the exploration rate and accuracy of the SLAM. We also validated the solution with a real-time experiment with a real robot in a controlled 2D environment. This work has been accepted at the 10th International conference on Automation, Robotics and Applications (IEEE ICARA 2024) which will be held in Athens, Greece in February 2024. [28]

7.2 Future perspectives

In the future, we plan on testing this algorithm in a real-world 3D environment. Apart from this, an in-depth study on the multi-robot extension of this approach will be carried out followed by real-world multi-robot experiments. Apart from this, we also aim to extend this algorithm to other standard and custom Visual SLAM based approaches.

Bibliography

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005, ISBN: 0262201623 9780262201629. [Online]. Available: http://www.amazon.de/gp/product/0262201623/102-8479661-9831324?v=glance&n=283155&n=507846&s=books&v=glance.

[2] M. Labbé and F. Michaud, "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of field robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[3] S. Macenski, F. Martin, R. White, and J. Ginés Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[4] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, 1997, pp. 146–151. DOI: 10.1109/CIRA.1997.613851.

[5] E. Bonetto, P. Goldschmid, M. Pabst, M. J. Black, and A. Ahmad, "Irotate: Active visual slam for omnidirectional robots," *Robotics and Autonomous Systems*, vol. 154, p. 104 102, 2022, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2022.104102. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889022000550.

[6] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1396–1402. DOI: 10.1109/IROS.2017.8202319.

[7] C.-Y. Wu and H.-Y. Lin, "Autonomous mobile robot exploration in unknown indoor environments based on rapidly-exploring random tree," in *2019 IEEE International Conference on Industrial Technology (ICIT)*, 2019, pp. 1345–1350. DOI: 10.1109/ICIT.2019.8754938.

[8] Y. Tang, J. Cai, M. Chen, X. Yan, and Y. Xie, "An autonomous exploration algorithm using environment-robot interacted traversability analysis," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4885–4890. DOI: 10.1109/IROS40897.2019.8967940.

[9] S. Ahmad and J. S. Humbert, "Efficient sampling-based planning for subterranean exploration," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 7114–7121. DOI: 10.1109/IROS47612.2022.9982169.

[10] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient autonomous exploration planning of large scale 3d-environments," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, Feb. 2019. DOI: 10.1109/LRA.2019.2897343.

[11] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2002, 540–545 vol.1. DOI: 10.1109/IRDS.2002.1041446.

[12] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," in *Proceedings of Robotics: Science and Systems*, Jun. 2005, pp. 65–72. DOI: 10.15607/RSS.2005.I.009.

[13] C. Luca, D. Jingjing, K. Miguel, B. Basilio, and I. Marina, "Active slam and exploration with particle filters using kullback-leibler divergence," in *Journal of Intelligent & Robotic Systems*, 2014, pp. 291–311. DOI: 10.1007/s10846-013-9981-9.

[14] F. Pukelsheim, *Optimal Design of Experiments* (Classics in Applied Mathematics). Society for Industrial and Applied Mathematics, 2006, ISBN: 9780898716047. [Online]. Available: https://books.google.fr/books?id=5ZcfDZUJ4F8C.

[15] H. Carrillo, I. Reid, and J. A. Castellanos, "On the comparison of uncertainty criteria for active slam," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 2080–2087. DOI: 10.1109/ICRA.2012.6224890.

[16] B. Mu, L. Paull, A. Agha-mohammadi, J. J. Leonard, and J. P. How, "Information-based active SLAM via topological feature graphs," *CoRR*, vol. abs/1509.08155, 2015. arXiv: 1509.08155. [Online]. Available: http://arxiv.org/abs/1509.08155.

[17] P. Fankhauser and M. Hutter, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation," in *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, A. Koubaa, Ed., Springer, 2016, ch. 5, ISBN: 978-3-319-26052-5. DOI: 10.1007/978-3-319-26054-9{_}5. [Online]. Available: http://www.springer.com/de/book/9783319260525.

[18] D. F. Joseph Carsten Arturo Rankin and A. Stentz, "Global path planning on board the mars exploration rovers," 2007.

[19] T. H. Chung, V. Orekhov, and A. Maio, "Into the robotic depths: Analysis and insights from the darpa subterranean challenge," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 6, no. 1, pp. 477–502, 2023. DOI: 10.1146/annurev-control-062722-100728.

[20] C. Cao, H. Zhu, F. Yang, *et al.*, "Autonomous exploration development environment and the planning algorithms," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 8921–8928.

[21] N. Hudson, F. Talbot, M. Cox, *et al.*, "Heterogeneous ground and air platforms, homogeneous sensing: Team csiro data61's approach to the darpa subterranean challenge," *arXiv preprint arXiv:2104.09053*, 2021.

[22] C. Debeunne, J. Vallvé, A. Torres, and D. Vivet, "Fast bi-monocular Visual Odometry using Factor Graph Sparsification," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Detroit (MI), United States, Oct. 2023. [Online]. Available: https://hal.science/hal-04185948.

[23] R. Sim, G. Dudek, and N. Roy, "Online control policy optimization for minimizing map uncertainty during exploration," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 2, 2004, 1758–1763 Vol.2. DOI: 10.1109/ROBOT.2004.1308078.

[24] C. Potthast and G. Sukhatme, "A probabilistic framework for next best view estimation in a cluttered environment," *Journal of Visual Communication and Image Representation*, vol. 25, pp. 148–164, Jan. 2014. DOI: 10.1016/j.jvcir.2013.07.006.

[25] Y. Chen, L. Zhao, Y. Zhang, S. Huang, and G. Dissanayake, "Anchor selection for slam based on graph topology and submodular optimization," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 329–350, 2022. DOI: 10.1109/TRO.2021.3078333.

[26] Z. Zhang and D. Scaramuzza, "Beyond point clouds: Fisher information field for active visual localization," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 5986–5992. DOI: 10.1109/ICRA.2019.8793680.

[27] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. DOI: https://doi.org/10.1002/nav.3800020109. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.3800020109. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109.

[28] S. Saravanan, C. Chauffaut, C. Chanel, and D. Vivet, "FIT-SLAM - Fisher Information and Traversability estimation-based Active SLAM for exploration in 3D environments," in *2024 10th International Conference on Automation, Robotics and Applications*, 2024.

[29] T. Moore and D. Stouch, "A generalized extended kalman filter implementation for the robot operating system," in *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*, Springer, Jul. 2014.

Appendices

Appendix A

Working using ROS 2

A.1 Publisher-Subscriber Model

The interaction between nodes (separate programmable entities) in this work is majorly carried out using the publisher-subscriber model. The publishers are responsible for publishing messages on a common-topic and the subscribers run multi-threaded callbacks to receive new messages that are published onto a topic. The topics can also be name spaced customized to felicitate ease of use. The ability for the publishers and subscribers to be turned on and turned off on the go is what makes the ROS 2 communication protocol a very handy tool.

A.2 Services and Parameters

ROS 2 also provides an API to program asynchronised server-client requests. The idea is very similar to a web server client except in this case, an instance of ROS interfaces (programmed with custom **.srv** files) acts as a medium of communication. ROS 2 also allows to setup global as well as node specific parameters which can be either static or dynamically re-configurable.

A.3 Actions

Although server-client links satisfy most needs, it might be important to receive feedback from the server during processing the requests. Here is where actions come in handy. In ROS 2, the action library simply runs a publisher-subscriber model in the back-end. An action client sends a request to the action server and the action server has the option to provide consistent feedback about its state through a hidden topic.

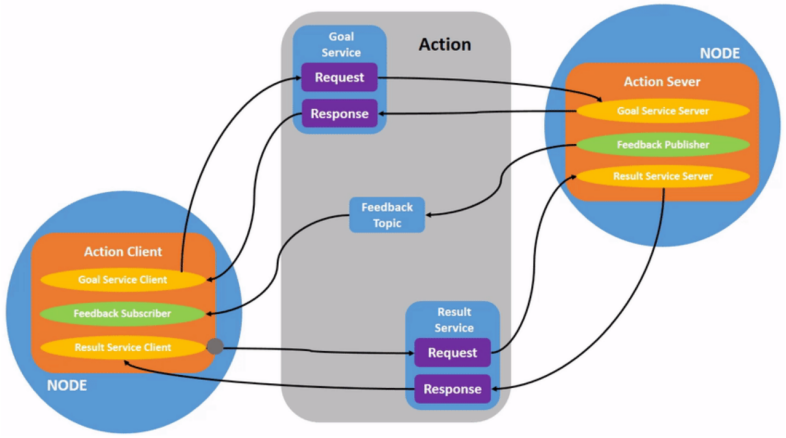

FIGURE A.1: An image representing how messages are passed in ROS 2. The action server-client runs a service-client for processing requests and uses a topic in the backend for providing feedback. Image is courtesy of https://www.ros.org

A.4 Using Gazebo for simulations

Gazebo is a 3D physics-simulator that is very widely used with ROS. It possesses the capacity to precisely simulate in very complicated environments. The integration of gazebo with ROS and the ability to quickly model robots and transformations using simple URDF files makes it a powerful tool in simulating the exploration problem addressed. Fig. B.1 shows two gazebo simulated worlds with the robot used for exploration.

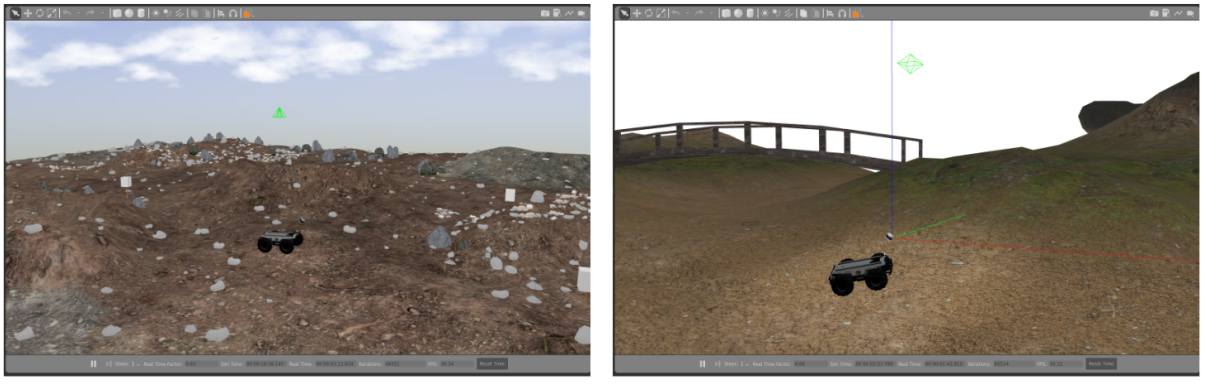

FIGURE A.2: Two images showing the simulation of the Scout-V2 in a gazebo environment. Left: Marsyard of the European Rover Challenge (ERC) and Eight: Gazebo inspection world with the lakes and water removed.

Appendix B

Sensor Data fusion and the Transform Tree

B.1 Fusing data from the wide array of sensors.

Although the research of sensor fusion is a highly profound. It is out of the scope of the research documented in this thesis. However, a very simple Extended Kalman filter based sensor fusion from the robot_localization package is used. The continuous odometry is provided by the on-board sensors. In this case, the encoders on the wheel and the IMU. The data from these are fused to provide the odometry using [29].

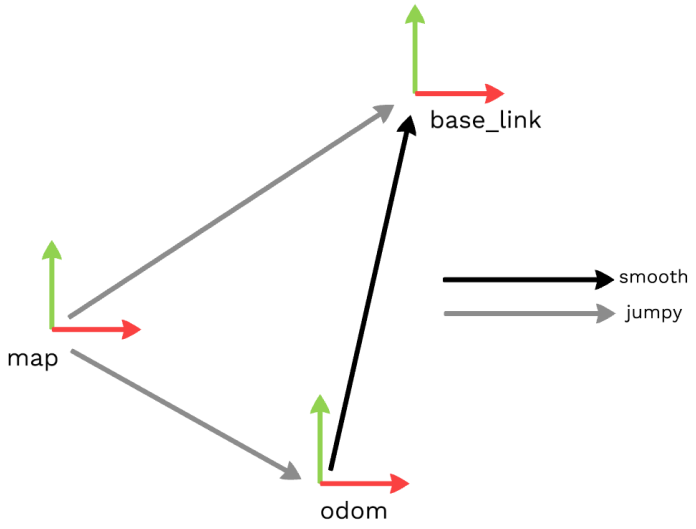

FIGURE B.1: A picture depicting the continous and jumpy odometries that act as a link between the three main frames used in navigation.

B.2 Need for continuous odometry.

The robot frame is usually referred to the frame named *base_link*. The world frame in our case is termed as the *map* frame. There is an intermediate frame between the *map* and the *base_link* called as the *odom* frame. The link between the *base_link* and the *odom* frame provided by the continous odometry is highly prone to drift and cannot be used as an accurate estimation of the robot's position in the environment. However, in relative terms, the established transform from fusing the data from the IMU and wheel is relatively accurate.

The SLAM backend provides the transform between the *base_link* frame and the *map* frame. This could also be provided using a GPS. Directly using this transform to estimate the relative position between *base_link* and *odom* can allow the estimate of the robot's position in the environment to jump around. This leads to the introduction of the *map* frame. In ROS, a frame is allowed to have only a single parent frame. Therefore, using the estimate between *base_link* and *odom* as well as *base_link* and *map*, we estimate the transform between *odom* and *map*. This is also the reason why single-shot global planning can be done on the discontinuous map frame and the local planning is done in the odom frame.

B.3 The full transform tree

In the transform tree depicted in Fig. B.2, the frames are not name-spaced with the robot name. However, in case of multiple robots, they can be interfaced like so, $robot\_name/frame\_name$. Every frame has a single parent frame. The frames of the wheels of the robots are excluded to avoid confusion.

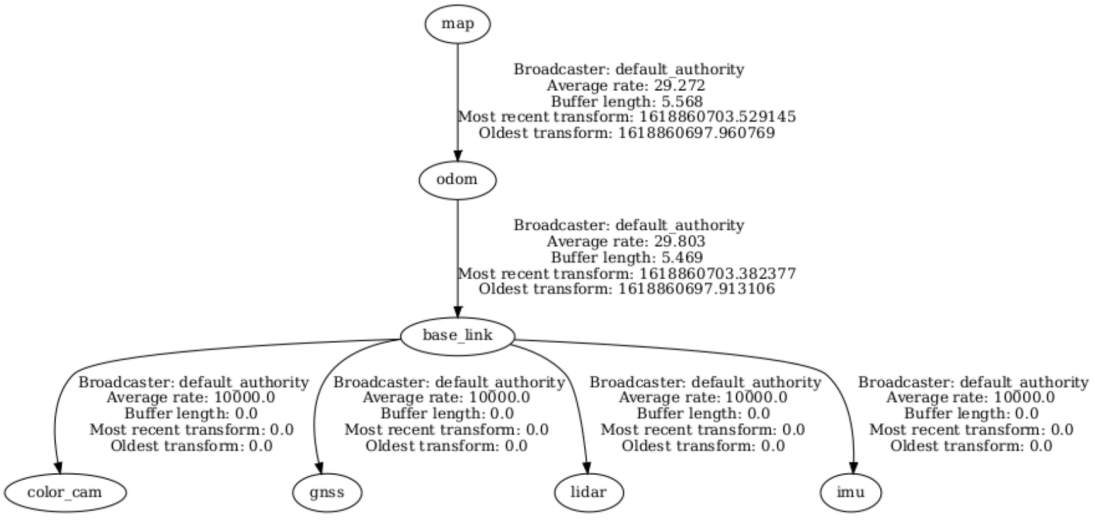

FIGURE B.2: An image showing a simple transform tree for explanatory purposes.

Appendix C

Docker-based implementations

To increase ease of use of the exploration software on different computers running different operating systems without the overhead burden of a virtual machine, docker was integrated into the pipeline. This was majorly helpful when there was a need to very easily run the exploration on the Jetson AGX Xavier running Ubuntu 20.04 on a ARM-64 processor while the entire stack was tested on computer running Ubuntu 22.04 and an AMD-64 processor. Docker Buildx is a tool that extends the capabilities of the Docker CLI (Command Line Interface) to enable cross-platform builds which enabled us to perform the build for the Jetson AGX a standard AMD-64 based processor.

The docker images built for ROS 2 usually inherit from the images already built by the Open source robotics foundation (OSRF). An example of a custom docker image creation for ROS Noetic is shown in Fig. C.1. The custom implementation of the docker image used in this work for ROS 2 Humble is freely available at https://github.com/suchetanrs/docker-files.

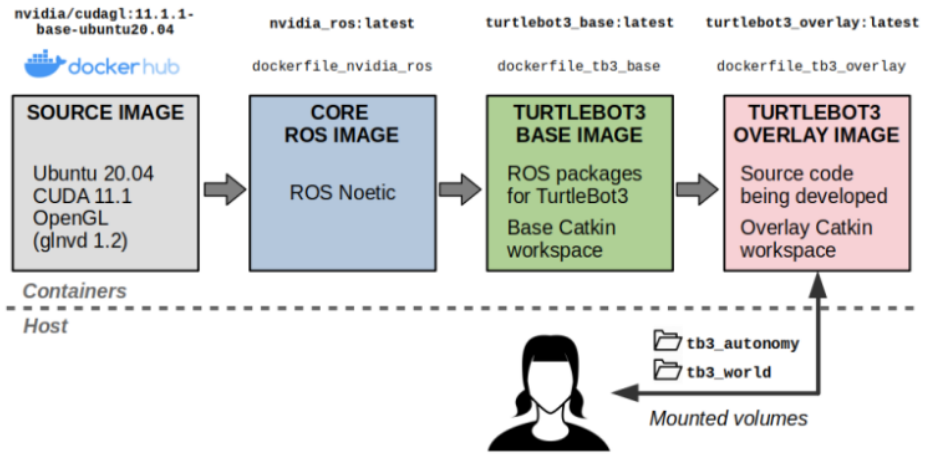

FIGURE C.1: An image depicting the typical workflow to build a docker container for ROS. Courtesy of https://robohub.org/an-updated-guide-to-docker-and-ros-2/