2-1-2022

Heterogeneous Collaborative Mapping for Autonomous Mobile Systems

Sooraj Sunil
University of Windsor

Heterogeneous Collaborative Mapping for Autonomous Mobile Systems

by

Sooraj Sunil

A Thesis
Submitted to the Faculty of Graduate Studies
through the Department of Electrical and Computer Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science at the
University of Windsor

Windsor, Ontario, Canada

Heterogeneous Collaborative Mapping for Autonomous Mobile Systems

by
Sooraj Sunil

APPROVED BY:

J. Ahamed
Department of Mechanical, Automotive and Materials Engineering

---

A. H. Sakr
Department of Electrical and Computer Engineering

S. Alirezaee, Co-Advisor
Department of Electrical and Computer Engineering

B. Shahrrava, Co-Advisor
Department of Electrical and Computer Engineering

January 28, 2022

Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

An accurate map of the environment is essential for autonomous robot navigation. During collaborative simultaneous localization and mapping, the individual robots usually represent the environment as probabilistic occupancy grid maps. These maps can be exchanged among robots and fused to reduce the overall exploration time, which is the main advantage of the collaborative systems. Such fusion is challenging due to the unknown initial correspondence problem. This thesis presents a novel feature-based map fusion approach through detecting, describing, and matching geometrically consistent features present in the overlapping region between the maps. The main drawback of usual feature-based approaches is the incapability to establish adequate valid feature correspondence primarily due to noisy sensory observation. Further, many existing map fusion approaches neglect the heterogeneity which arises due to different map resolutions and types of mapping sensors. This thesis shows that exploiting the probabilistic spatial information to refine the maps and utilizing nonlinear diffusion filtering to detect features can drastically improve the feature-matching performance. Additionally, this thesis presents a certainty grid fusion approach based on Bayesian inference to fuse pair-wise grid information. It also presents an extensive comparison of traditional feature detection methods to register map images at different scales. Finally, the effectiveness of the proposed method is illustrated based on the following map fusion assumptions using real-world data: homogeneous, hierarchical, and heterogeneous (fusing different resolution maps and maps generated using different types of mapping sensors).

Acknowledgment

Firstly, I would like to express my appreciation and gratitude to my co-advisor, Dr. B. Shahrrava, for the guidance and motivation throughout this thesis. Next, I thank my co-advisor, Dr. S. Alirezaee, for his invaluable support, constructive scholarly advice, and sharing his expertise in the field of robotics. Further, I thank Dr. S. Alirezaee for providing the lab space, time, and necessary resources to conduct this research. I extend my gratitude to Dr. J. Ahamed and Dr. A. H. Sakr for their invaluable insights and active participation in my committee.

Further, I would like to thank all the members of the Mechatronics Research Lab for giving me a perfect research environment to work in with all the fun and offering support whenever needed. I thank Dr. S. Mozaffari for introducing me to this area of research and spending countless hours discussing my work, the papers, and this thesis. I thank Andria Ballo for all the administrative support throughout my research period.

I thank the University of Windsor for numerous scholarships, work-study opportunities, and graduate teaching assistantships that I received during my course of study. I also thank Dr. S. Alirezaee for the research assistantship.

Special thanks to my family and friends for their endless love, support, and motivation. Thank you for always being there.

Contents

List of Tables

List of Figures

Chapter 1

THESIS OVERVIEW

1.1 Introduction

An accurate model of the environment and relative position of the robot are substantial for robotic autonomy, especially for mobile autonomous systems. Usually, these two unknowns are jointly estimated when given the sensory observations (e.g. RGB-D cameras, laser range finders etc.). The process of joint estimation of the pose and the map is popularly known as simultaneous localization and mapping (SLAM) [1]. Solution to the SLAM problem is considered as one of the fundamental requirements to achieve true robot autonomy. SLAM could be extremely useful during scenarios when the global positioning system (GPS) is inaccessible, e.g., in environments such as indoor, underwater [2], forest canopy [3], etc. However, the interdependence nature of localization and mapping makes SLAM a complex and challenging problem in practical applications. Despite the complexity, SLAM has been a theoretically well-studied field of research that still draws significant attention due to its vast practical importance.

The extension of SLAM problem to multiple agents is known as collaborative SLAM (also referred to as multi-agent, cooperative or multiple-robot SLAM) [4, 5]. When compared to single-robot SLAM, the collaborative SLAM systems have the following advantages:

More Robust: failure of a single-agent from the team does not affect the entire

robotic mission. Thus, multiple-agent systems are more robust.

Faster exploration: task allocation among agents reduce the overall time required for environment exploration and execution of mapping tasks. Hence, collaborative systems are faster.

Low Resources: distribution of processing tasks among agents can reduce the computational burden and power consumption of individual robots. Therefore, collaborative systems are efficient.

Minimize accumulation of errors: single-agent systems usually accumulate errors over time which can be minimized to a great extend by using collaborative systems.

Applications of collaborative SLAM include disaster management [6], autonomous underwater vehicles [7], subterranean exploration [8], rescue robots [9, 10], security surveillance, etc. Moreover, using autonomous robot swarms is an inexpensive alternative to having humans performing dangerous and hazardous tasks [4].

The rest of the chapter is divided into two sections. The following section discusses the challenges that are introduced by the collaborative systems. Section 1.3 provides an overview of thesis contributions, validation method and structure.

1.2 Challenges

While single-robot SLAM is challenging enough, extending it to multiple robots introduces a new set of problems [4, 5]. Out of all challenges, the primary problem is to incorporate all the available information obtained from individual robots into a common global reference frame. This problem can be solved easily if the initial positions of the individual robots are known. However, this assumption doesn't hold true for many real-world applications. Therefore, one need to first find the *unknown initial correspondence* between the robots to transform the individual map information and combine them to construct a consistent global map. Existing solutions to the

initial correspondence problem can be broadly categorized based on their underlying assumptions into the following two cases [5]:

Robot-to-robot observations (direct solution): the reference frame transformations are estimated using the raw sensor data exchanged between the robots. This type of solutions generally assume either uninterrupted communication [11], certain coordination conditions for the meeting [12], or both [13].

Map overlaps (indirect solution): each robot generate independent maps in its own reference coordinates which is known as local maps. When the local maps consist of some amount of overlapping, a set of candidate transformation can be computed to align the maps. Such integration of local maps is known as map fusion (also referred to as map merging or alignment or registration) problem [4]. Here, the processed maps are exchanged between the robots rather than raw measurements. Examples for indirect solutions include feature-based matching [14], spectra-based matching [15], scan matching [16,17], and sub-map matching [9].

Some approaches also utilize both raw measurements and processed maps to find the relative transformations [7, 18]. In general, indirect solutions require lower communication and data processing. Further, it doesn't require any coordination among robots therefore, it is a more flexible [4]. Hence, indirect solutions well suit real-time collaborative applications when compared to the direct solutions. On the downside, indirect approaches rely completely on the quality of individual maps. Therefore, in order to obtain a consistent global map, it is important to use robust single-robot SLAM solutions to construct the local maps.

1.2.1 Heterogeneity During Collaborative Mapping

Many existing methods [4, 15, 19–22] primarily focuses on homogeneous information for merging. However, there arises heterogeneous scenarios during collaborative mapping due to the differences in map properties, type of sensors and even the type of the robot itself [4]. Heteregenous map merging is relatively less addressed area of

research when compared to homogeneous case. There exists relatively few literature even for the 3D sensors [17, 23].

Out of all the existing robotic environment representation, occupancy grid maps are the most common choice [1, 4, 14–16, 24]; especially for mobile robot navigation. A critical variable in the representation of an environment as grid maps is the grid resolution. It might appear to be a minor detail but it impacts the overall memory (size of the map), computation time, and quality of the maps [24]. Similarly, the rate at which the scans are fed to the SLAM algorithm can also have a notable impact on the computation time and quality of the maps. These variables can be different for the individual robots in a collaborative system due to the available computational resources. Such differences in grid resolutions and sampling frequencies introduces heterogeneity in the collaborative system. A special instance of heterogeneous map fusion is when prior information is integrated to the occupancy grid maps. This include floor maps [25–27], semantic blueprints [25, 28], underground pipe network maps [26], or even hand-drawn sketches. The integrated information is known as hybrid maps. It has many advantages such as semantic classification [25] and accurate localization [28] which suits time critical applications such as search and rescue [26].

In mobile robotics, due to limited computation and sensor cost either 2D LIDARs, low-cost cameras or both are predominantly preferred over 3D LIDAR for navigation and mapping tasks. The LIDAR and camera sensory data has different resolutions and error models. Thus, this introduces a new challenge in heterogeneous map fusion which is integration of data obtained from different type of perception sensors. To the best of our knowledge, there exist no literature that deals with 2D occupancy map fusion for heterogeneous sensors.

1.3 Thesis Contributions

This thesis focuses on the integration of partial occupancy grid maps obtained from individual robots operating in different parts of the same environment. An indirect map fusion method based on *image registration* [14–16, 19, 21, 22, 29, 30] is presented.

In general, these approaches utilize the overlapping area between the local maps to establish feature correspondences. Then, a set of transformation parameters to align the maps are computed based on the accepted correspondences. Figure 1.1 illustrates the map fusion problem schematically. A major drawback in such feature-based fusion methods is that it usually fails to find sufficient number of valid correspondences. To overcome this, we present a method which is fast, robust, and highly reliable in terms of establishing geometrically consistent feature correspondences across various mapping conditions. This is achieved by means of processing the spatial occupancy probabilities and detecting features based on locally adaptive nonlinear diffusion filtering. We also present a procedure to verify and accept the correct transformation to avoid ambiguous map merging. Further, a global grid fusion strategy based on Bayesian inference which is independent of order of merging is also provided.

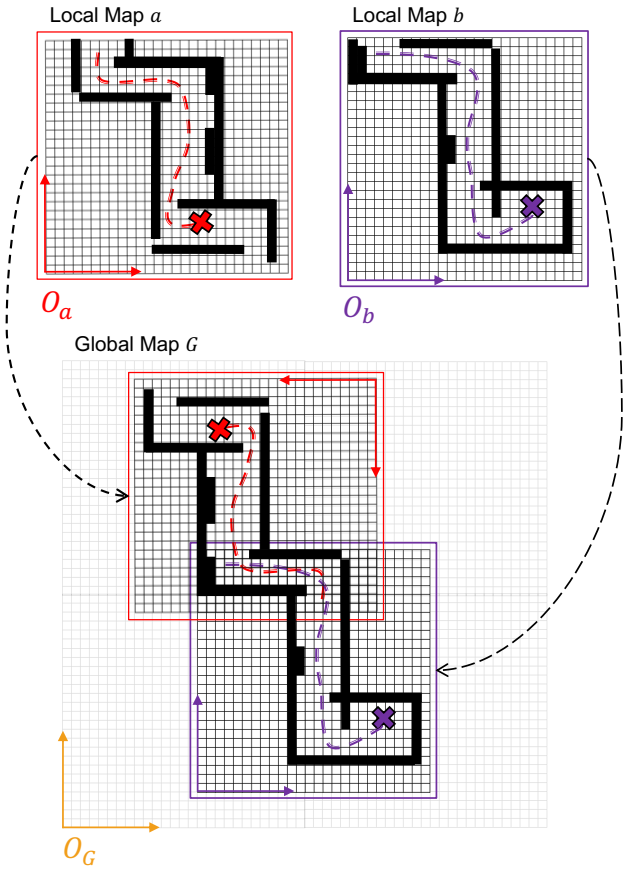


Figure 1.1: **Occupancy grid map fusion based on map overlaps.** Two local maps a, b with origins O_a, O_b are being fused into the global grid with origin O_G. The overlapping area across the local maps fall squarely on top of each other.

The summary of the presented method based on the challenges it addresses is given as follows:

- *Finding correspondences across maps:* To estimate the required transformation, common regions or points in the unknown overlapping area between the maps should be extracted and matched. This problem is solved by extracting high probable grid information from each map to find distinctive interest points based on locally adaptive nonlinear diffusion filtering. The detected points are then presumptuously matched using the nearest-neighbor criterion to establish pairwise correspondence. Then, the required transformation matrix to align the maps is computed using the robust M-estimator SAmple Consensus (MSAC) algorithm by eliminating the outliers in the nearest-neighbor set.

- *Transformation verification:* To avoid ambiguity in the map merging process, the estimated transformation is verified using an map acceptance index and the cardinality of the largest inlier set returned by the MSAC algorithm.

- *Unknown relative poses of robots:* Each robot involved in a collaborative system should be localized globally. The computed transformation can be used to transform the pose information of the robots into the global frame.

- *Heterogeneous maps:* To deal with the scaling problem (i.e. merging maps with different scales), the known grid resolution information is used to find the required scaling between maps. Thus, the parameters are only estimated for a rigid transform model (only rotation and translation).

- *Updating global information:* After accepting a transformation, the local occupancy grids are fused using Bayesian inference. Thus, the uncertainty in the probabilities are taken into account in the global map. Further, this fusion procedure can be easily extended to arbitrary number of robots.

The effectiveness of the proposed map fusion approach is experimentally validated using real-world data. The data collection procedure is described in the upcoming subsection. Experimental results shows the applicability of the presented method

for noisy, low overlapping, and different grid resolutions maps. Unlike any previous methods, map merging is also applied to heterogeneous sensor maps (fusing camera and LIDAR maps) as well. Most importantly, the suitability of the feature-based map fusion method to perform hierarchical large-scale collaborative mapping is illustrated.

1.4 Thesis Validation

The experimental results in this thesis is validated based on the data collected using a scaled model car called QCar by Quanser. The Qcar is equipped with a wide range of sensors such as 2D LIDAR (RPLIDAR-A2 laser range scanner), RGB-D camera (Intel D435), 9 axis IMU (Inertial Measurement Unit), wheel encoders, and an onboard computer to carry out various mapping and navigation tasks. Figure 1.2 illustrates the QCar along with it's components.

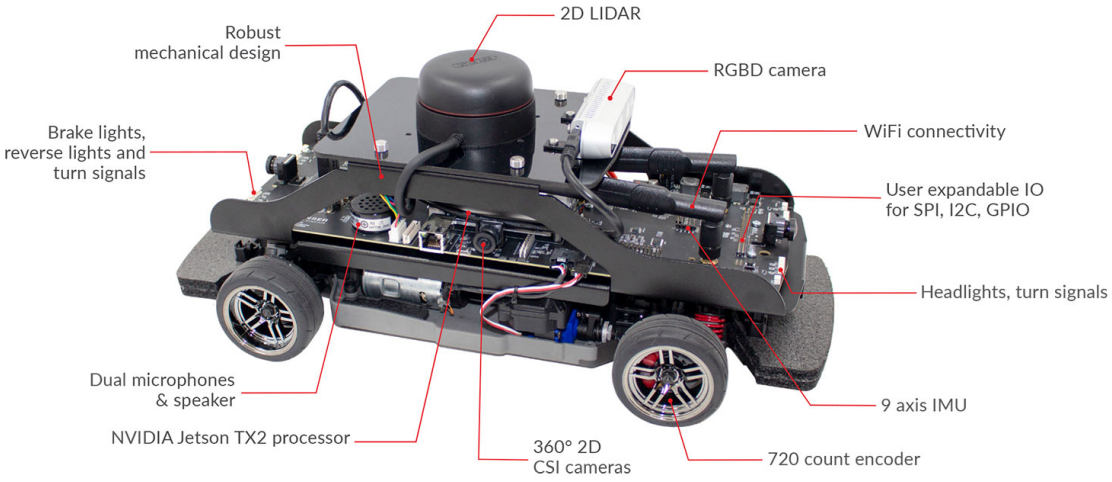


Figure 1.2: **Qcar used for experimental validation.** [31]

1.5 Thesis Structure

The rest of the thesis is organized as follows:

- **Chapter 2** provides background on the single-robot SLAM and multiple-robot SLAM problems. Further, it reviews notable solutions on feature-based map

fusion methods.

- **Chapter 3** describes the stages involved in the proposed feature-based occupancy map merging algorithm. The stages are as follows: (i) occupancy map processing stage to accurately localize the features in a local map, (ii) feature detection, (iii) feature matching, and (iv) pairwise occupancy grid fusion.

- **Chapter 4** provides a comparative study on different feature detection methods for registering occupancy map images. The feature detectors are studied to test their detectability, matching efficiency and reliability to find valid correspondences across heterogeneous mapping conditions.

- **Chapter 5** demonstrates the effectiveness of the presented method using the QCar for different real-world scenarios.

- **Chapter 6** summarizes the research and outlines few interesting research topics for future direction.

Chapter 2

SIMULTANEOUS LOCALIZATION AND MAPPING

The chapter is subdivided into three sections. Section 2.1 provides an overview of the single-robot SLAM problem and reviews notable solutions. Section 2.2 describes the occupancy mapping algorithm. Section 2.3 extends the SLAM problem to collaborative systems, highlights the necessity for map fusion, and summarizes existing methods in detail.

2.1 Single-robot SLAM

SLAM enables robot navigation which is the most crucial task in mobile autonomous systems. The motive of SLAM algorithms is to simultaneously estimate the spatial map of an environment and the relative pose of a robot based on available sensory observations. It is extremely useful to localize a robot in an unknown environment, especially in the absence of GPS.

To formulate the SLAM problem, consider the state sequence of the robot $\mathbf{x}_{1:t} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t\}$ that comprises the pose (position and orientation) information. The robot input control (or odometry) sequence $\mathbf{u}_{1:t} = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_t\}$ accounts for

the change in $\mathbf{x}$ within the environment. The sequence $\mathbf{z}_{1:t} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t\}$ denotes sensory observations made by the robot.

Given the robot controls, sensory observations, and the initial pose information $\mathbf{x}_0$; the goal is to estimate either the momentary pose $\mathbf{x}_t$ or the entire trajectory $\mathbf{x}_{1:t}$ along with the map $\mathbf{m}$. From probabilistic point of view, SLAM problem requires estimation of the following joint posterior probability distributions [24]:

$$p(\mathbf{x}_t, \mathbf{m}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{x}_0) \tag{2.1}$$

$$p(\mathbf{x}_{1:t}, \mathbf{m}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{x}_0) \tag{2.2}$$

where the equations (2.1) and (2.2) are known as *online SLAM* and *full SLAM* respectively.

In general, the SLAM posterior in (2.1) and (2.2) can recursively estimated when an observation and motion model is provided. However, the interdependence nature of the map and the robot pose makes SLAM an extremely challenging problem for real-time applications. Figure 2.1 illustrates a graphical model of the relationship between the variables involved in the full SLAM problem.

2.1.1 Related Literature - Single-robot SLAM

The earlier solutions model the SLAM problem as an online state estimation. Under the highly restrictive Gaussian assumption, EKF-SLAM [32] applies the extended Kalman filter (EKF) recursively to the online SLAM problem. However, EKF-SLAM requires distinctive features to be present in the environment and high computation for updating the filter. In order to overcome these limitations, EIF-SLAM [33] based on sparse extended information filters (EIF) was proposed. The EIF is based on the inverse of the covariance matrix and addresses full SLAM problem. But in practice, this method require large matrix inversions to recover map and pose information.

The next set of SLAM paradigm are based on Monte Carlo methods (particle filters) which ease the Gaussian assumption. These approaches can perform

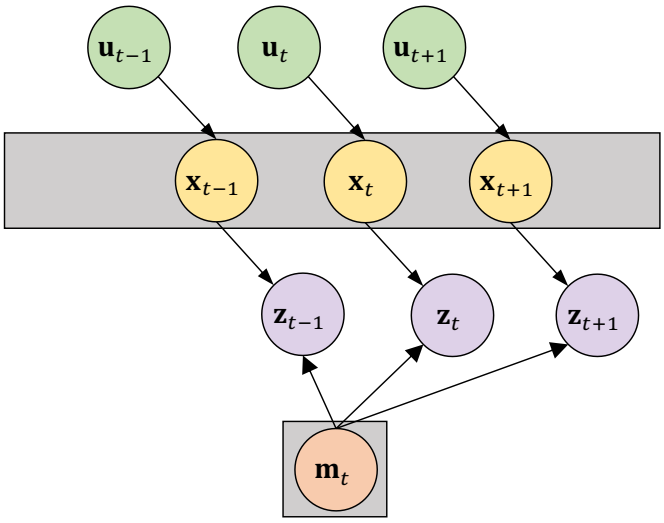


Figure 2.1: **Graphical model of the single-robot full SLAM problem.** The arrows indicate how different blocks impact another block. Variables in the bounding boxes depict the unknowns (unobservable variables present in the system).

multi-modal state estimation using the Rao-Blackwellized particle filter (RBPFs). The RBPFs factorization of the SLAM posterior probability decomposes the problem into separate localization and mapping problems. A few notable particle filter based SLAM methods include FastSLAM [34] and FastSLAM 2.0 [35]. The main drawback for such algorithms lie in their complexity measured in terms of the number of particles which is required to learn an accurate map. Consequently, an alternative approach that reduces the memory requirements and computational burden of mapping with RBPFs was proposed in [36].

Lately, there has been enormous interest in graph-based approaches which solves SLAM problem through nonlinear sparse optimization. The basic intuition here is that two poses (*nodes*) are linked by a soft spatial constraint (*edge*) which describes the best guess of how far the poses are apart. The graph-based SLAM aims to find a configuration that minimize the following cost function [37]:

$$F(\mathbf{x}, \mathbf{m}) = \sum_{ij} \mathbf{e}_{ij}(\mathbf{x}, \mathbf{m})^T \Omega_{ij} \mathbf{e}_{ij}(\mathbf{x}, \mathbf{m}) \tag{2.3}$$

where the minimization to obtain optimal values $\mathbf{x}^*$ and $\mathbf{m}^*$ is given as

$$(\mathbf{x}^*, \mathbf{m}^*) = \arg\min_{\mathbf{x},\mathbf{m}} F(\mathbf{x}, \mathbf{m}) \tag{2.4}$$

where Ω_{ij} in (2.3) indicates the information matrix between the node i and j; $\mathbf{e}_{ij}$ is the difference between the predicted and actual observation $\mathbf{z}$. In general, the minimization of non-linear function $F(\mathbf{x}, \mathbf{m})$ is usually simplified by standard optimization methods such as Gauss-Newton, Levenberg-Marquardt, Gauss-Seidel relaxation or gradient descent.

The SLAM solutions for mobile systems can be further categorized based on the sensor as follows:

- **Laser SLAM:** The inputs from laser range sensors (distance sensors) such as LIDAR is primarily used to perform SLAM. The laser sensor produces precise observations of the environment thereby it can be effectively used for accurate mapping of the environment. The pose information is usually retrieved by sequential scan matching (lidar odometry) [38].

- **Visual SLAM:** These methods primarily utilize inputs from low cost cameras and other image sensors to execute SLAM. Technology related to visual SLAM include structure from motion, visual odometry, and bundle adjustment [39].

2.2 Occupancy Grid Maps

To utilize the knowledge about the environment, one requires a mathematical model. Existing robotic map representations include grid maps, feature maps, topological maps, semantic maps, appearance maps, and hybrid maps. Out of all representations, the occupancy grid map is the most common choice [4, 14–17, 24, 29, 30]; especially for mobile robot navigation [24, 40].

Occupancy grid maps can be generated from noisy sensor observations while assuming that the pose information is known. The primary idea is that the robot's environment is represented as a discrete grid thereby it can be used to find (planning)

collision-free paths for autonomous navigation.

To formulate the grid mapping approach, consider a grid cell $\mathbf{m}_i$ with index i. A grid map partitions the space into finitely many grid cells. Each $\mathbf{m}_i$ has an associated binary random variable which specifies the spatial value of that cell. The probability occupancy grid approach aims to calculate the following posterior [24]:

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \prod_i p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \tag{2.5}$$

Whenever the robot makes an observation that is in the perceptual field of the cells, the occupancy probabilities can be recursively calculated using the *binary Bayes filter* [24]. The recursive relation can be given as

$$l_{t,i} = l_{t-1,i} + \textbf{inverseSensorModel}(m_i, \mathbf{x}_t, \mathbf{z}_t) - l_0 \tag{2.6}$$

where $l_{t,i}$ is the log-odds ratio which is defined as

$$l_{t,i} = \log \frac{p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{1 - p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})} \tag{2.7}$$

The function **inverseSensorModel** in (2.6) indicates the sensor observation model in its log-odds form which is specific to the type of range sensor used in the mapping process. The constant l_0 is the occupancy prior. The Algorithm 1 [24] summarizes the update steps involved in the probability occupancy grid mapping.

Algorithm 1 $[\{l_{t,i}\}] = \textbf{occupancyGridMapping}(\{l_{t-1,i}\}, \mathbf{x}_t, \mathbf{z}_t)$

for all cells $\mathbf{m}_i$ **do**
  if $\mathbf{m}_i$ in perceptual field of $\mathbf{z}_t$ **then**
    $l_{t,i} = l_{t-1,i}+$ **inverseSensorModel**$(m_i, \mathbf{x}_t, \mathbf{z}_t) - l_0$
  else
    $l_{t,i} = l_{t-1,i}$
  end if
end for

Figure 2.2 demonstrate an example of an occupancy grid map of a simulated house layout (Gazebo simulator). The map was generated using a laser-based SLAM Robot Operating System (ROS) package called gmapping [36].

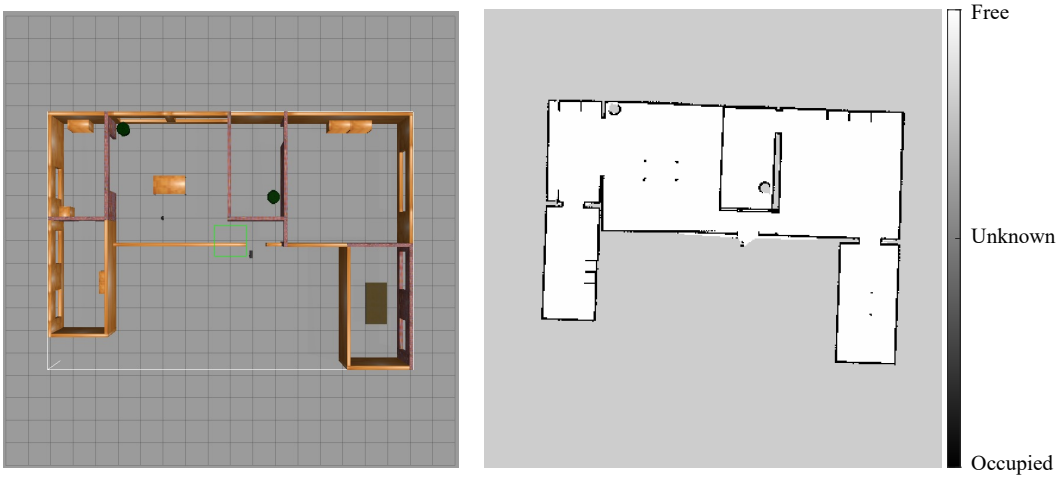


Figure 2.2: **Occupancy grid map example.** White cells are obstacle-free area, black cells are occupied area, and gray cells indicate the unknown area.

2.3 Collaborative SLAM

Based on the terminology introduced in section 2.1, the posterior of collaborative SLAM involving two robots a and b based on direct observations can be denoted as follows:

$$p(\mathbf{x}^a_{1:t}, \mathbf{x}^b_{1:t}, \mathbf{m} \mid \mathbf{z}^a_{1:t}, \mathbf{z}^b_{1:t}, \mathbf{u}^a_{1:t}, \mathbf{u}^b_{1:t}, \mathbf{x}^a_0, \mathbf{x}^b_0) \tag{2.8}$$

where $\mathbf{x}_0$ indicate the initial pose information. Figure 2.3 depicts the direct collaborative SLAM problem involving two robots graphically.

In order to formulate the indirect solution (i.e. map fusion), consider two local maps ${}^a\mathbf{m}$ and ${}^b\mathbf{m}$ generated by robots a and b. For simplicity, the time index t is disregarded. Assuming that the individual maps are modeled as occupancy grids, then each grid map can be viewed as $a(x, y)$ and $b(x, y)$ map images. Therefore, the

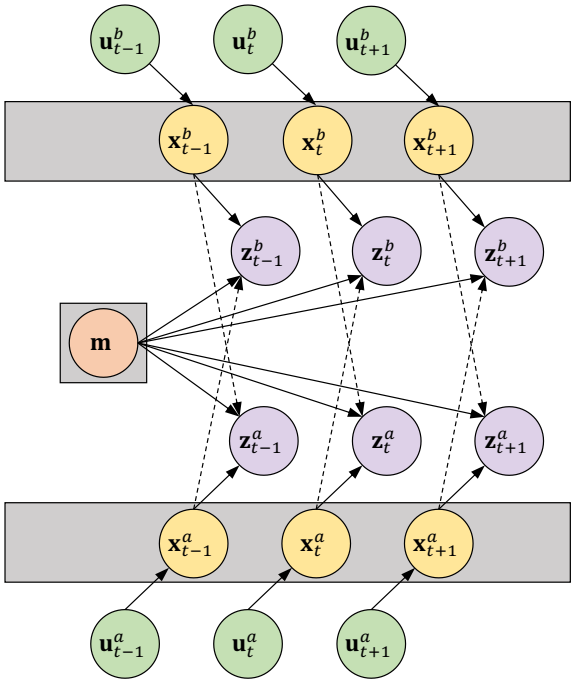


Figure 2.3: **Graphical model of the collaborative SLAM problem (direct solution) for two robots a and b.** The arrows indicate how different blocks impact one another. The region in bounding boxes depict the unknown variables in the system. Dashed lines indicate the line-of-sight observations.

pixel co-ordinates of the maps can be denoted as

$$ {}^a\mathbf{m} = \begin{bmatrix} {}^a x \\ {}^a y \end{bmatrix} \quad \text{and} \quad {}^b\mathbf{m} = \begin{bmatrix} {}^b x \\ {}^b y \end{bmatrix} \tag{2.9} $$

Further, assuming that there exist an overlapping between the maps, the motive now is to find the geometric transformation that transforms $[{}^b x, {}^b y] \rightarrow [{}^a x, {}^a y]$ in such a way that the overlapping area falls squarely on top of each other. This transformation is $\mathbf{T}$ and the expression to transform from ${}^b\mathbf{m}$ to ${}^a\mathbf{m}$ can be written as follows:

$$ {}^a\mathbf{m}_b = {}^a\mathbf{T}_b({}^b\mathbf{m}) \tag{2.10} $$

Based on the resolutions ${}^a r, {}^b r$ of the individual grid maps, the transformation $\mathbf{T}$ can take two models. If ${}^a r \neq {}^b r$, then $\mathbf{T}$ takes the *similarity transform* model for

which the transformation parameters such as translation $\mathbf{l}$, rotation $\mathbf{R}_\theta$, and scale change λ need to be estimated. Given the estimated parameters, the map $\mathbf{m}$ can be transformed into $\overline{\mathbf{m}}$ as

$$\overline{\mathbf{m}} = \lambda \mathbf{R}\mathbf{m} + \mathbf{l} \tag{2.11}$$

where

$$\lambda = \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix}, \mathbf{R} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}, \text{and} \quad \mathbf{l} = \begin{bmatrix} l_x \\ l_y \end{bmatrix} \tag{2.12}$$

If the grid maps are constructed using same scale (i.e. ${}^a r = {}^b r$), then $\mathbf{T}$, then the scaling factor $\lambda = 1$. Now, $\mathbf{T}$ reduce to a special instance of similarity transformation which is the *rigid transform*. Therefore, (2.11) can be simplified as

$$\overline{\mathbf{m}} = \mathbf{R}\mathbf{m} + \mathbf{l} \tag{2.13}$$

It must be noted that during occupancy grid mapping, the resolutions of maps are known. Hence, even for the case of heterogeneous resolutions, only the rotation and translation is required to be estimated. Whereas the scaling factor can be directly calculated based on the given resolutions.

For convenience, the transformation is represented using the homogeneous coordinates as

$$\mathbf{T}(l_x, l_y, \theta) = \begin{bmatrix} cos\theta & -sin\theta & l_x \\ sin\theta & cos\theta & l_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

where $\mathbf{T}$ is commonly referred as map transformation matrix; and it is generally estimated by pairing two grid maps. Similarly, transformation for (2.11) can be

represented as

$$\mathbf{T}(\lambda, l_x, l_y, \theta) = \begin{bmatrix} \lambda cos\theta & -sin\theta & l_x \\ sin\theta & \lambda cos\theta & l_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

Figure 2.4 depicts the indirect collaborative map merging problem involving $i = 1, \ldots, n$ robots. It is important to note that each robot during indirect collaborative mapping maintains its own individual SLAM maps.

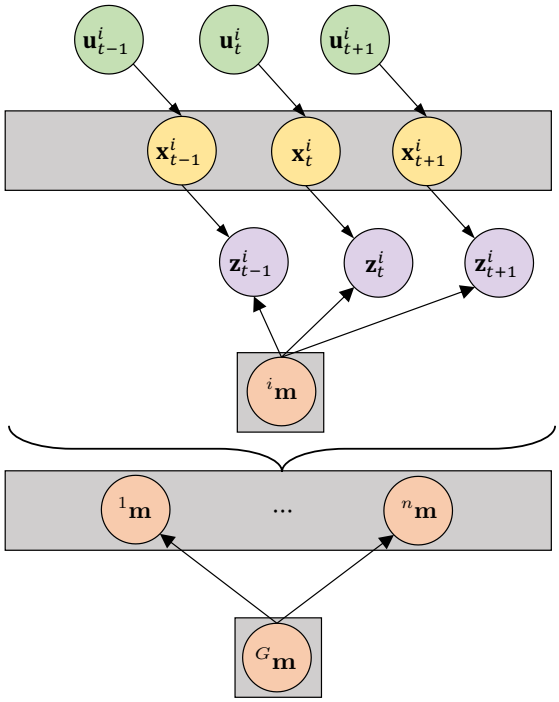


Figure 2.4: **Graphical model of the map fusion problem (indirect solution) for n robots.** The arrows indicate how different blocks impact one another. The region in bounding boxes depict unknown variables in the system.

2.3.1 Related Literature - Collaborative SLAM

Initial experimental results in [11] based on manual feature extraction depicted the potentiality of feature-based techniques to perform successful map merging. Since then, numerous methods based on feature matching has been proposed to solve the

collaborative mapping problem. In this section, the state-of-the-art map merging approaches that utilizes geometric features, specifically for the case of occupancy grid maps is exclusively reviewed.

2.3.1.1 Map Fusion Based on Spectral Features

The usage of spectral information for the map merging problem is motivated by the difficulties involved in determining the map overlaps in Euclidean space [4, 15]. Common spectrum-based map merging methods uses either are the Hough transform [41] or the Random transform [42] to mainly detect line segments in the grid maps.

A deterministic, non-iterative and fast map merging approach based on Hough spectrum was proposed in [15]. The algorithm extracts line features in an occupancy map using the Discretized Hough transform (DHT) and performs cross correlation to determine the map similarities. It produce a set of ranked map transformation parameters for a *rigid transformation* by assuming the mapping environment is linearly structured. However, this assumption necessitates a significant degree of the overlapping area to be presented across the individual maps. Therefore, the algorithm is not suitable for maps with less overlaps. Another drawback is that the method is only applicable to the maps which are constructed with the same grid resolution. In response, the Hough transform was combined with visual landmarks matching to reduce the requirement of large overlapping area between maps in [43]. The detectability of the spectral line-feature was enhanced further by employing a refinement procedure based on the thinning algorithm. However to implement this method, the model of the visual landmarks has to be provided in advance which might not practical for many applications. Further in [44], the properties of Hough peak matching was exploited to overcome some of the drawbacks in previous spectrum-based methods. Although spectrum-based methods were improved to merge maps with low-overlapping in [44], the results in [45] showed that the involved parameter discretization may cause inaccurate merging. Hence in [46], a sampling-based optimization and direct observations were used to increase the accuracy of spectrum-based map merging.

2.3.1.2 Map Fusion based on Image Features

The occupancy grid map of an indoor environment typically consists of distinguishable features such as corners, edges, columns, or doorways. This is also true for specific outdoor maps which consists of walls, buildings, vehicles etc. Since the environment is assumed to be static, the features are well defined in an occupancy map. Hence, the grid maps can interpreted as gray scale raster images and image registration techniques can be employed to solve the map fusion problem. The overall steps involved in map image registration can be divided into the following stages [47]:

Feature (keypoint) Detection: during this stage, the map image is searched for locally distinctive locations that are likely to match with other images.

Feature Description: the region around each detected feature is converted into a compact and stable descriptor that can be used to match against other descriptors.

Feature Matching: lastly, at this stage we efficiently search for likely matching candidates between two set of descriptors to establish pair-wise correspondence.

Figure 2.5 depicts a simplified schematic of the occupancy grid map fusion based on feature detection and matching. This technique can be straightforwardly extended to n number of robots where the computation increase based on the combinations $C(n, k = 2)$.

A probabilistic approach to the map merging problem was formulated in [30] using the multi-hypothesis Random sample consensus (RANSAC) [48] and the iterative closest point (ICP) [49]. To select the best feature detector, the performance of scale invariant feature transform (SIFT) [50], speeded up robust features (SURF) [51], Kanade–Lucas–Tomasi (KLT) [52], and Harris corner detectors [53] were collectively studied. The results mainly highlighted the distinctiveness of different feature detectors and descriptors in terms of image filtering, minimum error, repeatability, and computation time. Specifically, it was shown that the different feature detectors required different filter size (Gaussian filter and median filter) to achieve their best

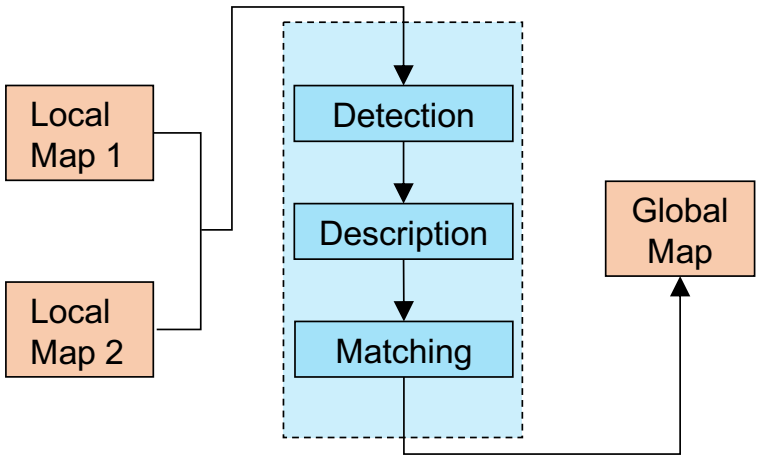


Figure 2.5: **Map merging based on image feature matching.**

performance. The SIFT feature detector was also used in [29] to detect the overlapping regions across maps. In this approach, the scale invariant property of the SIFT detector was exploited to merge maps with different resolutions.

The problem of map fusion is viewed as an point set registration problem in [16, 40, 45]. These approaches used a variant of the ICP algorithm to find the required transformation to align the overlapping maps. The ICP algorithm is local convergent, hence it requires good initialization. Therefore, in [40, 45], the image feature matching using SIFT was used to provide the initial transformation. Whereas [45] used the Harris corner detector to extract edge point set. In [45], a multi-scale context-based descriptor based on eigenvalues and eigenvectors was designed to describe the detected edge features. All three approaches used the RANSAC algorithm to find geometrically consistent feature correspondences. Then the scaling trimmed iterative closest point (STrICP) was used to perform the point set registration to refine the transformation. Additionally, a robust motion averaging was used in [40] to recover the global transformation from a set of relative transformations obtained by matching numerous grid map pairs.

In [22], maps were merged using the ORB (oriented features from accelerated segment test (FAST) and rotated binary robust independent elementary features

(BRIEF)) [54] feature detector. The detected ORB keypoints were matched using the brute-force algorithm to establish putative feature matches. Then, the RANSAC algorithm is fitted to an affine transform model to estimate the required transformation to align the maps.

A computationally fast corner detector for grid maps and a cylindrical descriptor [14] was used in [19] to enable collaborative SLAM. This method utilized the RANSAC algorithm to estimate and refine the transformation parameters. Further, it also presented a decision-making algorithm which reduces the overall exploration time.

Even though the SIFT-based methods are very popular, the Gaussian blurring involved in scale space construction may affect the feature localization at the pixel level [29, 55]. As mentioned earlier, a major drawback in the feature-based methods is that it usually fail to find sufficient number of geometrically consistent correspondences. Hence, many approaches [16, 29, 30, 40, 45] used an additional refinement step to improve the final transformation. In general finding the correct combination of feature detector, descriptor and matching technique has been the primary problem of feature-based map fusion approaches [56].

Chapter 3

PROPOSED FEATURE-BASED MAP FUSION

The aim of this chapter is to describe a solution to the map fusion problem from an *image registration* viewpoint when the local maps are represented as occupancy grid maps. The overall steps involved in estimating the transformation **T** required to align the maps is summarized as follows:

- In section 3.1, the probability layer of a grid map is systematically processed to separate high probable grid information. This allows us to perform accurate feature detection and matching between the maps.

- In section 3.2, the KAZE features [55] which utilize locally adaptive nonlinear diffusion filtering to construct the scale space is described. The choice of interest feature is different from previous feature-based map merging methods [16, 19, 21, 22, 29, 45].

- In section 3.4, the nearest-neighbor matching technique to find putative correspondences between the detected features is described.

- In section 3.5, we describe a robust method to obtain the map transformation matrix by eliminating outliers present in the putative correspondence using a variant of the RANSAC algorithm.

- In section 3.6, we describe the verification procedure based on map acceptance index and the cardinality of the largest inlier set to accept a valid transformation. It also presents the gird fusion strategy to fuse the local cell probabilities using Bayesian inference.

Figure 3.1 illustrates an example cycle of successful map registration between two laser-based grid maps with reasonable overlaps using the proposed method.

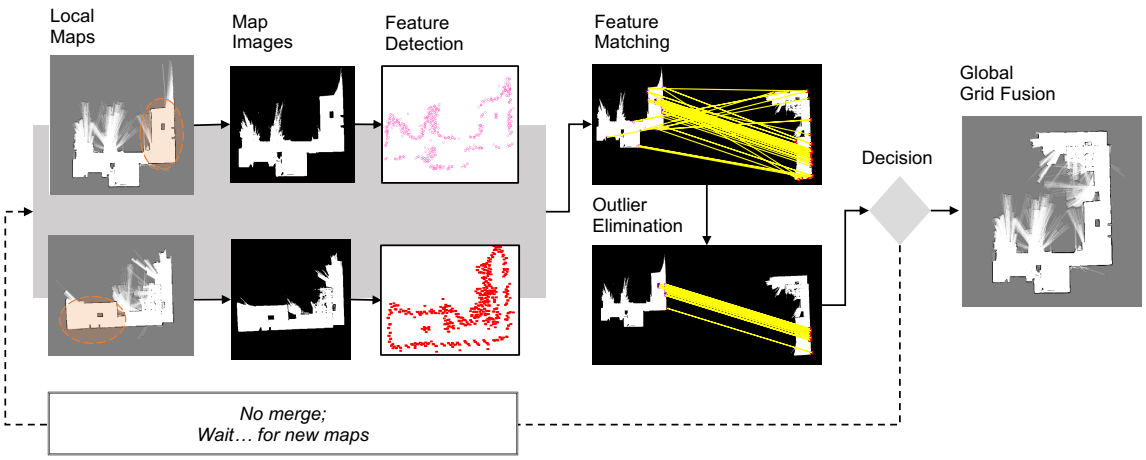


Figure 3.1: **One cycle of the presented map fusion approach.**

3.1 Processing Occupancy Maps

Most of the existing map merging techniques represent the occupancy maps as map images by sampling the occupied cells to an almost uniformly distributed point set. However, grid maps usually consist of artifacts which can be viewed as high-frequency noises in images. Hence, some approaches [21, 29, 30] applied image filters (blurring) to refine them. However, filtering map images may affect the pixel level accuracy of keypoint localization. Therefore to avoid these drawbacks, instead of using the occupied cells we use the obstacle-free cells. This has two main advantages. Firstly, the number of obstacle-free cells in a map is more when compared to the occupied cells. Secondly, the obstacle-free layer will usually have higher grid probabilities. By

exploiting these two general assumptions, we can apply a global threshold to the obstacle-free layer of an occupancy map to eliminate cells with less probabilities.

The obstacle-free cells in a probabilistic occupancy map can be easily extracted by exploiting the cell-independence assumption. To do so, consider that the probability layer of an occupancy map $\mathbf{m}$ is viewed as a matrix $\mathbf{M}$ with r rows and c columns. Each grid cell in $\mathbf{M}$ denotes a binary random variable which indicates the probability of occupancy $p(m_i)$ of the cell. The index i indicate the pixel coordinate position ($i = [x, y]^T$, where $x = 1, \ldots, r$ and $y = 1, \ldots, c$). The occupancy information such as whether a cell is occupied, obstacle-free or unknown can be inferred based on the probability value as

$$p(m_i) = \begin{cases} 1 & \textbf{then} \quad p(m_i = occ) \\ 0 & \textbf{then} \quad p(m_i = free) \\ 0.5 & \textbf{then} \quad p(m_i = unknown) \end{cases} \tag{3.1}$$

We are interested in $p(\neg m_i)$ which is the probability of free cells in the map. Assuming $p(m_i) = 1 - p(\neg m_i)$, we can easily calculate the probability of the i^{th} cell being obstacle-free as

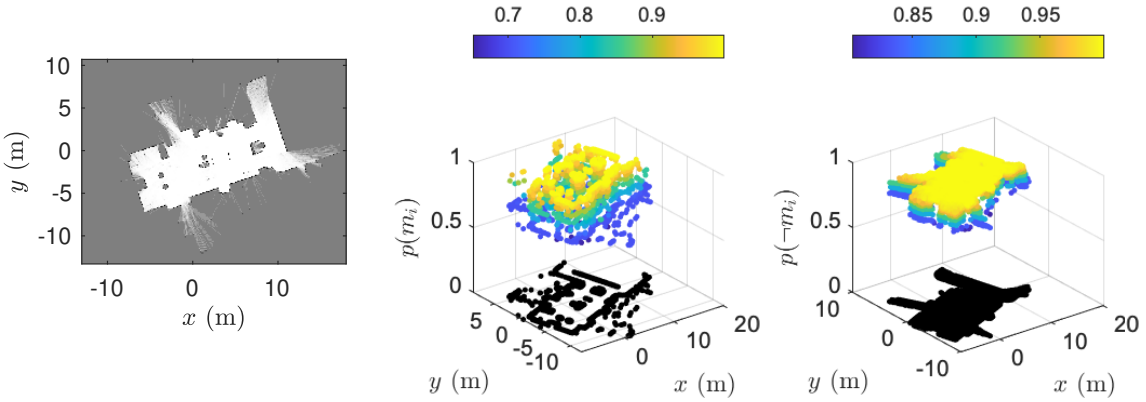
$$p(m_i = free) = p(\neg m_i) = 1 - p(m_i) \tag{3.2}$$

Figure 3.2 illustrates an exemplar occupancy grid map alongside the extracted occupied and obstacle-free probability layer. It can be clearly observed that the obstacle-free layer consist of more informative as well as high probable spatial information. It also preserves the structural boundaries od the environment in comparison with the occupied layer.

Now, the structural details present in the free layer can be refined by removing the cells (setting to 0) whose probability values $p(\neg m_i)$ are low. For example, we can set all values of $p(\neg m_i)$ to 0 expect for $p(\neg m_i) > \alpha$ where $0.01 \leq \alpha \leq 0.99$ is a user-controlled threshold factor. Let the map image after thresholding be denoted as I. Then , I will only consist of obstacle-free cells whose probability is greater

than α. That is, the values of occupied and unknown cells in the map will be 0. This simple but effective manipulation of the gird maps highly favors the process of feature detection and matching. The effectiveness of such representation is further emphasized experimentally in subsection 3.5.

Figure 3.2 illustrates an exemplar occupancy grid map alongside the extracted occupied and obstacle-free probability layer. It can be clearly observed that the obstacle-free layer consist of more informative as well as high probable spatial information. It also preserves the structural boundaries od the environment in comparison with the occupied layer.



(a) Occupancy map (b) Occupied layer (c) Obstacle-free layer

Figure 3.2: **An example of probabilistic grid map alongside the position of occupied and obstacle-free cells with their respective spatial probabilities.**

Now, the structural details present in the free layer can be refined by removing the cells (setting to 0) whose probability values $p(\neg m_i)$ are low. For example, we can set all values of $p(\neg m_i)$ to 0 expect for $p(\neg m_i) > \alpha$ where $0.01 \leq \alpha \leq 0.99$ is a user-controlled threshold factor. Let the map image after thresholding be denoted as I. Then , I will only consist of obstacle-free cells whose probability is greater than α. That is, the values of occupied and unknown cells in the map will be 0. This simple yet effective manipulation of the gird maps highly favors the process of feature detection and matching. Figure 3.3 illustrates the effect of thresholding using α. The effectiveness of such representation is further emphasized experimentally in subsection 3.5.

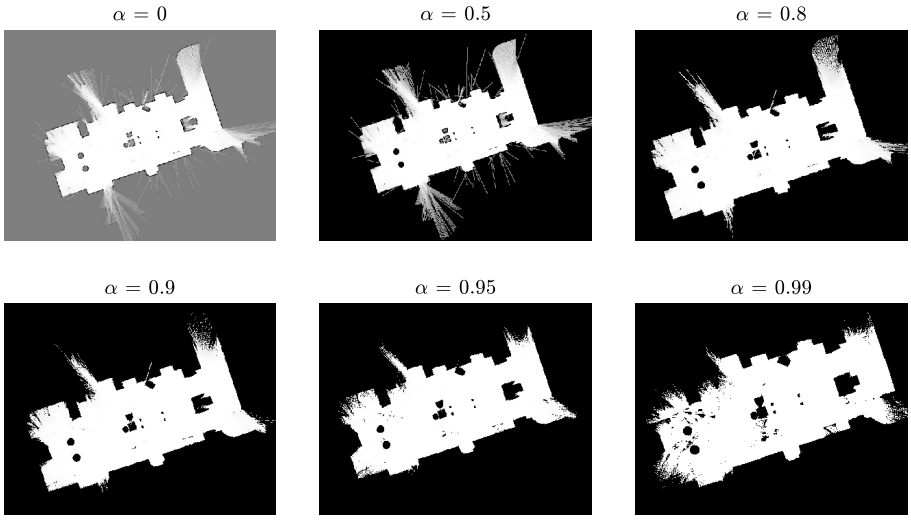


Figure 3.3: **One cycle of the presented map fusion approach.**

Remark. *The threshold factor α can be chosen arbitrarily or it can be made as a function of any performance measure (e.g., acceptance index [15]) involved in the feature matching process. It is also worth mentioning that the threshold is applied only to the extracted probability layer (copy) of the occupancy map. This assumption is valid as only the copy of maps is exchanged between robots during collaborative exploration. Thus, the loss in map information due to α doesn't affect the merged global map.*

3.2 Feature Detection

A keypoint in general is a locally distinctive point (also region or line) in an image based on the nature of intensity values around it. Repeatable keypoints can be identified using keypoint detection methods. Classical 2D feature detectors construct the scale space of an image by filtering the original image with an appropriate function over increasing time or scale. For instance, the Gaussian scale space [50] is constructed by convolving the image with a Gaussian kernel of increasing standard deviation. With multi-scale image representation, image features can be detected at different scale levels or resolutions. However, Gaussian blurring does not respect the natural

boundaries of objects thus filter both details and noise at all scale levels to the same degree [55, 57]. This impacts the localization accuracy of the detected keypoints [55].

3.2.1 Nonlinear diffusion filtering

Whenever discrete images are viewed as continuous objects, powerful calculus of variations can be applied. This enables us to perform filtering that depends on the local content of the original image. The nonlinear diffusion [57] enables locally adaptive filtering which preserves the crucial edge details in occupancy images. The filtering can be described by the following partial differential equation:

$$\frac{\partial I}{\partial t} = div(c(x, y, t) \cdot \nabla I) \tag{3.3}$$

where div and ∇ are the divergence and gradient operators respectively and I is the image (luminance). The conductivity function c enables to make the diffusion locally adaptive to the image structure. The Perona-Malik diffusion in [57] proposed to make the function c to be dependent on the gradient magnitude as

$$c(x, y, t) = g(\mid \nabla I_\sigma(x, y, t) \mid) \tag{3.4}$$

where ∇I_σ is the gradient of a Gaussian smoothed version of the original image I with the standard deviation σ. Here we use the following diffusion coefficient g from [**?**]:

$$g = \frac{1}{1 + \frac{|\nabla I_\sigma|^2}{c^2}} \tag{3.5}$$

where the parameter c is the contrast factor that controls the sensitivity to edges. The value for c is usually estimated or made as a function of the noise in the image. It is important to note there exist different diffusion functions other than (3.5). Further, with constant diffusion coefficient, the diffusion equations reduce to *heat equation* which is equivalent to Gaussian blurring.

Since there exist no analytical solution to the partial differential equations

(PDEs), diffusion in (3.3) should be approximated. Here, we use exact discretization of the KAZE fetaures preseneted in [55], which adopts a $semi-implicit$ scheme. The discretization can be expressed as

$$\frac{I^{i+1} - I^{i}}{\tau} = \sum_{l=1}^{m} A_l(I^i) I^{i+1} \tag{3.6}$$

where I^i and I^{i+1} are the filtered images at the current and next level respectively; τ is the time difference and A_l is a matrix that encodes the image conductivity for each dimension. The solution for I^{i+1} can be obtained as follows:

$$I^{i+1} = \left(\mathbf{I} - \sum_{l=1}^{m} A_l(I^i) I^{i+1} \right)^{-1} I^i \tag{3.7}$$

3.2.2 KAZE features

To detect interest keypoints, the nonlinear scale space is constructed using the scheme and variable conductance diffusion described in previous subsection. A similar implementation to SIFT without sub-sampling is adopted to build the scale space arranged in logarithmic steps in series of O octaves and S sub-levels. The octave and the sub-level indexes are mapped to their corresponding scale σ as

$$\begin{aligned} \sigma_i(o, s) = 2^{o+s/S}, \quad o \in [0 \dots O-1], \\ s \in [0 \dots S-1], \quad i \in [0 \dots K] \end{aligned} \tag{3.8}$$

where o and s are the discrete octave and sub-level indexes and K is the total number of filtered images. Since nonlinear diffusion filtering operates in time units, the scale level is mapped $\sigma_i \rightarrow t_i$ to convert pixel to time units as

$$t_i = \frac{1}{2}\sigma_i^2, \quad i = \{0 \dots K\} \tag{3.9}$$

Once the nonlinear scale space is constructed, the interest points are detected based on the response of *scale-normalized determinant of the Hessian Matrix* at mul-

tiple scale levels. The determinant at different σ scale levels can be computed as

$$I_{Hessian} = \sigma^2(I_{xx}I_{yy} - I_{xy}^2) \tag{3.10}$$

where I_{xx}, I_{yy} are the second order horizontal and vertical derivatives respectively and I_{xy} is the second order cross derivative. The derivatives are approximated by 3×3 Scharr filters and the maximum is computed at all scale levels expect $i = 0$ (first) and $i = N$ (last) scales.

The main orientation in a local neighborhood is obtained through the derivatives around each keypoint within a radius of $6\sigma_i$, where σ_i is the sampling step. Each derivative in the circular area is weighted with a Gaussian centered at the keypoint. For each point, the dominant orientation is calculated by summing the derivatives within a sliding circle window over an angle of $\frac{\pi}{3}$.

3.3 Feature Description

The descriptor is a finite vector which summarizes the properties such as location, orientation, and response strength of the detected keypoints. In this thesis, the SIFT descriptor [50] is used to describe the detected features. This is mainly due to the slight distinctiveness shown by SIFT [56] while describing the features. Moreover, the KAZE descriptor [55] is an extension of the SURF description method [51] to the nonlinear scale space. Therefore, using the KAZE descriptor may also result in good matching results.

To describe the detected keypoints, the gradient magnitude and orientation are computed at each detected keypoint. These are weighted by a Gaussian window to give less importance to the gradients that are farther from the keypoint center. Then the samples are accumulated into orientation histograms summarizing the content over 16×16 sample array for which 4×4 descriptors are computed. More details on implementation can be found in [50]. Figure 3.4 illustrates a simplified schematic diagram of SIFT description method to describe a keypoint.

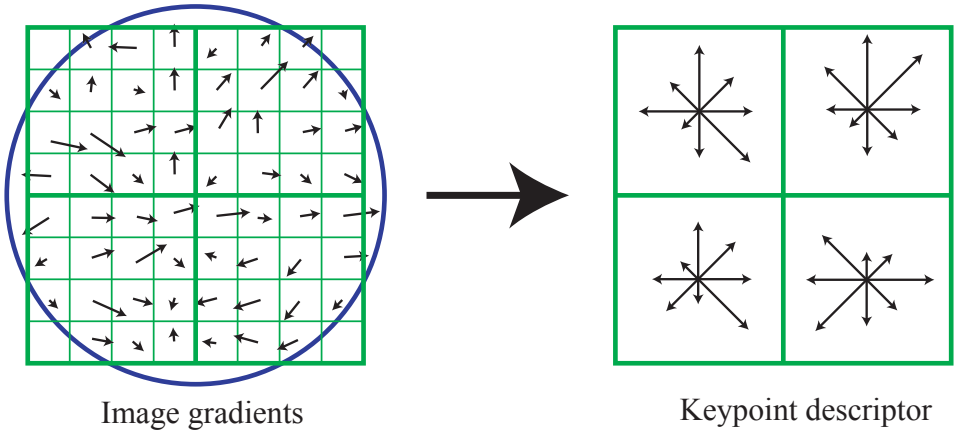


Figure 3.4: **2×2 SIFT feature description of** 8×8 **set of samples [50].** Left: computing the gradient magnitude and orientation. The overlaid circle indicate the Gaussian window. Right: 2×2 descriptor.

3.4 Feature Matching

Once the set of features are extracted, the feature matching stage sets up putative pairwise correspondence between the detected keypoints. The feature matching problem can be formulated as follows: given the feature descriptor $f_a^{1:n} = \{f_a^1, \ldots, f_a^n\}$, we seek the nearest neighbor from f^a for a query point from set $f_b^{1:m} = \{f_b^1, \ldots, f_b^m\}$. The matching process is usually exhaustive therefore, it is the most time consuming stage. The pairwise Euclidean distance d between the two feature vectors is used to find the nearest neighbor. The sum of squared differences (SSD) is used as the matching metric and the distance between two feature vectors f_i and f_j is calculated as

$$d(f^i, f^j) = \sum_u \sum_v \left(f^i(u,v) - f^j(u,v)\right)^2 \tag{3.11}$$

where ambiguous matches are rejected based on an effective ratio test [50]. The test accepts each match by comparing the closest neighbor with the second-closet neighbor. Therefore, only the unique matches below a certain matching distance are accepted as feature correspondence.

3.5 Outlier Elimination

The matching described in previous subsection can produce good matches (inliers). However, it may still be contaminated by inconsistent matches (outliers). Hence, for the next stage, a robust variant of RANSAC algorithm [48] known as M-estimator SAmple Consensus (MSAC) [58] algorithm is used to eliminate the outliers and extract geometrically consistent inliers.

RANSAC is a standard procedure to estimate the parameters of a certain mathematical model contaminated by outliers. The parameter model of interest here is the rigid transform in (2.14). The iterative RANSAC steps to estimate the transformation $\mathbf{T}$ for the largest set of *inliers* from the set of matched pairs $\mathcal{M}\{f_a^i, f_b^j\}$ is summarized in Algorithm 2.

Algorithm 2 $[\mathbf{T}, inliers] = \mathbf{RANSAC}(\mathcal{M}\{f_a^i, f_b^j\}, N_{trails})$

$inliers \Leftarrow 0$
$\mathbf{T} \Leftarrow 0$
$n \Leftarrow 0$
while $n < N_{trails}$ **do**
  $S \Leftarrow$ Randomly select subset of samples with minimum number of correspondence
  $\mathbf{T}_h \Leftarrow$ Hypothesize transformation for the minimal set
  $inliers_h \Leftarrow$ Test for number of consistent matches with $\mathbf{T_h}$
  if $inliers_h > inliers$ **then**
    $inliers \Leftarrow inliers_h$
    $\mathbf{T} \Leftarrow \mathbf{T}_h$
  end if
end while

RANSAC algorithm accepts feature pairs as inliers if it lies within a threshold τ. However for higher τ values the algorithm may produce poor estimate. This is mainly due to the cost function C which is used to score the inliers and outliers. To avoid this drawback, statistically robust MSAC is used. The MSAC algorithm introduces a new error term for the cost function into the RANSAC algorithm. The cost function C is defined as [58]:

$$C = \sum_i \rho(e_i^2) \tag{3.12}$$

where the error term ρ for the RANSAC and the MSAC algorithms are as follows:

$$\rho_{RANSAC}(e_i^2) = \begin{cases} 0 & e^2 < \tau \\ \text{constant} & e^2 \geq \tau^2 \end{cases} \tag{3.13}$$

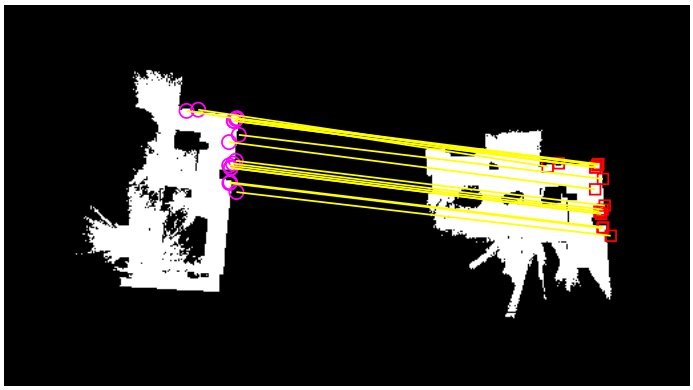
$$\rho_{MSAC}(e_i^2) = \begin{cases} e^2 & e^2 < \tau \\ \tau^2 & e^2 \geq \tau^2 \end{cases} \tag{3.14}$$

The main difference between (3.13) and (3.14) is that the ρ_{RANSAC} scores each outlier with a constant penalty whereas ρ_{MSAC} scores penalty for the outlier as well as the inliers. Thus, the MSAC algorithm yields an added benefit with no additional computation burden.

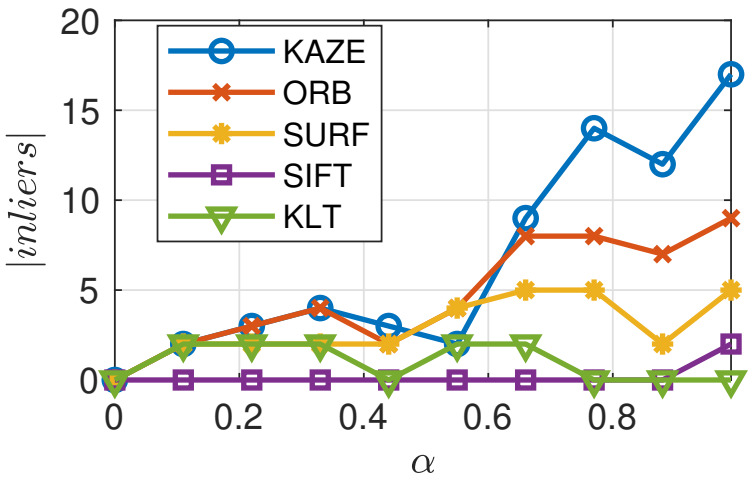
Figure 3.5a illustrates the location of geometrically consistent features in the common obstacle-free (described in subsection 3.1) region between two grid maps for one run of the MSAC algorithm. Figure 3.5b shows cardinality of the largest set of inliers returned by the MSAC algorithm for different feature detectors. The effect of thresholding cell probabilities based on α can be clearly observed, and the best result is obtained for KAZE feature with $\alpha = 0.99$ (i.e. by considering only the free gird cells whose probability is ≥ 0.99). It should be noted that when $\alpha = 0$, the original map itself is considered for feature matching. Thus, Figure 3.5b also highlights the resulting improvements by the processing stage described in subsection 3.1. Clearly, it is not just the KAZE feature which produce good results whereas it is the combination of thresholding the obstacle-free layer and the feature detector.

3.6 Grid Fusion

The final stage of the map fusion process is to verify the estimated transformation and then, fuse pairwise local probabilistic cell information of the individual maps based on the accepted transformation. Thereby, constructing a consistent global map of the environment.



(a)



(b)

Figure 3.5: **Geometrically consistent MSAC inliers across two occupancy maps.** (a) Location of KAZE inliers in the processed obstacle-free layer for threshold $\alpha = 0.99$, (b) cardinality of inlier set for KAZE, ORB, SURF, SIFT, and KLT methods.

3.6.1 Transformation Verification

We use the following two conditions to accept the estimated transformation:

- Even though only two valid feature correspondences are a sufficient fit for the rigid transform model to estimate the transformation, it is highly unlikely that the correspondences are true positives. Hence, only the transformation for a minimum inlier cardinality (well-over two feature correspondences) is accepted.

- Further, we use the acceptance index [15] based on pairwise cell agreement and

disagreement cells between map matrix $\mathbf{M}$ and transformed map matrix $\overline{\mathbf{M}}$ to check the quality of the transformation. The acceptance index ω is defined as

$$\omega(\mathbf{M},\overline{\mathbf{M}}) = \begin{cases} 0 & agr(\mathbf{M},\overline{\mathbf{M}}) = 0 \\ \frac{agr(\mathbf{M},\overline{\mathbf{M}})}{agr(\mathbf{M},\overline{\mathbf{M}})+dis(\mathbf{M},\overline{\mathbf{M}})} & agr(\mathbf{M},\overline{\mathbf{M}}) \neq 0 \end{cases} \tag{3.15}$$

where $agr(\mathbf{M},\overline{\mathbf{M}})$ indicates the number of cells in $\mathbf{M}$ and $\overline{\mathbf{M}}$ that agrees (either free or occupied). The disagreement $dis(\mathbf{M},\overline{\mathbf{M}})$ is the number of cells that disagrees (either $\mathbf{M}$ is free and $\overline{\mathbf{M}}$ is occupied or vice-versa).

3.6.2 Certainty Grid Fusion

Once the transformation is verified, the next stage is to combine the grid probabilities of the two local maps. Many existing methods focuses only on finding the transformation and aligning the maps whereas the fusion problem is given less consideration or not considered at all. Existing approaches include [44] in which the local grid probabilities were simply added by exploiting the additive property of log-odds. In [16,17], grid fusion rules were defined as lookup tables.

In this article, we use Bayesian inference to deal with the uncertainty and update the global grid probabilities. Using Bayes' rule, the probability of the global grid $p({}^{G}\{m_i\})$ given the grid probability $p({}^{a}\{m_i\})$ of map a and the transformed $(b \rightarrow a)$ grid probability $p({}^{a}\{\overline{m}_i\}_b)$ of map b can be calculated as:

$$p({}^{G}\{m_i\} \mid {}^{a}\{m_i\}, {}^{a}\{\overline{m}_i\}_b) = \frac{AB}{AB + (1-A)(1-B)} \tag{3.16}$$

where $A = p({}^{G}\{m_i\} \mid {}^{a}\{m_i\})$ and $B = p({}^{G}\{m\}_i \mid \{{}^{a}\{\overline{m}_i\}_b)$. The grid fusion in (3.16) can be extended to n robot maps as follows:

$$\frac{1}{p({}^{G}\{m_i\} \mid {}^{1}\{m_i\}, \ldots, {}^{n}\{m_i\})} - 1 = \prod_{k=1}^{n} \Big(\frac{1}{p({}^{G}\{m_i\} \mid {}^{n}\{m_i\})} - 1 \Big) \tag{3.17}$$

Since both (3.16) and (3.17) are associative and commutative, the order of operations is insignificant.

Chapter 4

FEATURE EVALUATION FOR REGISTERING HETEROGENEOUS MAPS

The computation complexity for constructing operable maps is crucial as it directly influence the performance of a mapping algorithms. Several factors such as type of sensors, SLAM algorithm, available computational resource, robot trajectory, or even the environment itself can determine the overall computation. Two common parameters which can be used to tune the properties of an occupancy map are the grid resolution and the scan rate. During collaborative scenarios, a robot equipped with a powerful CPU can construct maps using higher scanning frequency and grid resolution whereas low-cost processors have to settle for a restrictive range of resolutions and frequencies. In this chapter, dissimilar maps of the same environment generated with different values for r, T or both are referred to as heteregenous maps.

Evaluating results of collaborative mapping is challenging due to the difficulties involved in obtaining ground truth data to verify the matching quality. Many techniques evaluate the performance of map alignment based on manual annotation. The repeatability of SIFT, SURF, KLT, and Harris keypoint detectors were accessed in [30] based on manually verifying the correct pairings. In [27], the root mean square error (RMSE) and a success rate were used to compare different map alignment meth-

ods in which the successful alignments were manually labeled through visual inspection. In general, manual intervention is time-consuming, prone to errors, and may not be practical for real-time mapping applications. Recently in [19], different feature detectors were evaluated based on the time complexity, number of detected features, and distance traversed by the robots. However, this assessment lacks the measure of map alignment parameters (rotation and translation) which is critical for matching features in map fusion. Also, the comparison of keypoints for standard images in [59] is not precisely applicable for the case of grid maps.

Therefore, the motive of this chapter is to provide a set of evaluation techniques to compare feature-based alignment methods for matching heteregenous occupancy grid maps. Specifically, the keypoint detectors are assessed based on the effect of the following variables in grid maps:

Grid resolution, r: The number of grid cells used to represent a meter (cells per meter) of the environment. For example, 20 cells per meter gives a 5 cm precision.

Scan rate, T: The rate at which the sensor samples are fed to the SLAM algorithm. The rate is tuned by uniform down sampling of the actual sensor observation.

The rest of the chapter is divided into three sections. Section 4.1 presents the evaluation method which include the map data set of an indoor environment used for feature evaluation, and describes the methods used to access feature-based heterogeneous occupancy image matching. Section 4.2 details the different metrics used to evaluate the feature detectors and discusses the results. Finally, section 4.3 summarizes the evaluation findings.

4.1 Evaluation Method

4.1.1 Map Data Set

It is common to evaluate and validate the performance of a system based on available public data sets. There exist numerous evaluation data sets mainly for single-robot

SLAM. Even though various existing work on collaborative systems prefer them [16, 30, 40, 45], they are not necessarily fair as each author subdivides the map data conveniently [19]. Hence, a set of 2D laser maps of a real world indoor environment is generated using the QCar (Fig. 1.2). The set consist of 27 different grid maps of the same indoor environment. Each map was generated based on the following steps:

Data collection: First, the scans were captured approximately at 10 Hz using the 360° 2D laser scanner (RPLIDAR-A2) of the QCar.

SLAM algorithm: The graph SLAM [37, 38] algorithm is used to build the map based on the collected scans. The algorithm incrementally processes the scans to build a pose graph that links the scans. Whenever two regions of the map are found to be alike in the world irrespective of the robot pose history, the system calculates a relative transformation that aligns the two regions to *close the loop*. It utilizes the loop closure information to update the map and correct the estimated pose trajectory.

Heterogenity: To incorporate heterogeneity, the grid resolution and frequency of scans fed to the algorithm for each map were varied.

Figure 4.1 illustrates the values considered for r and T along with the time taken by the SLAM algorithm to complete the map. The factor τ here is defined as the ratio between the amount of time required to run the algorithm to the total duration to collect the original laser scans. The value for τ over 1 indicates that the algorithm lags behind the sensor's actual scanning frequency.

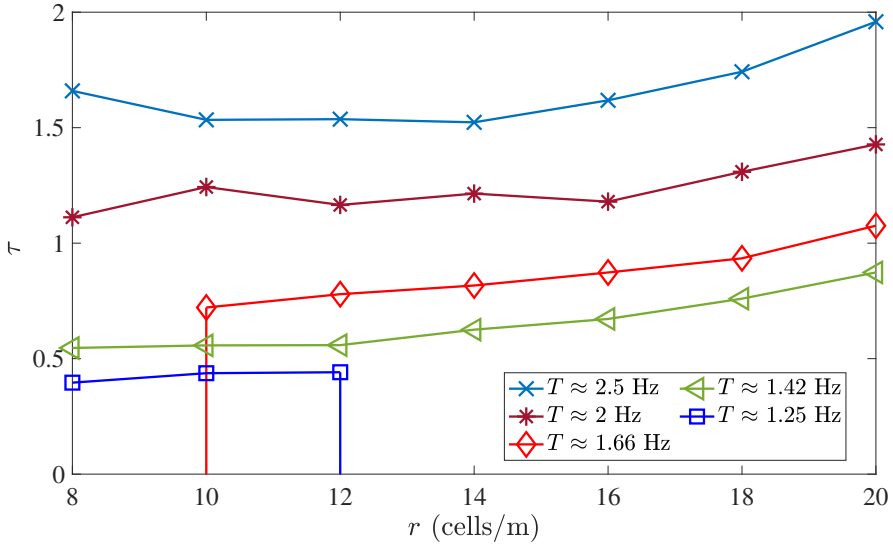


Figure 4.1: **Computation time τ for each map in the data set.** Solid line dropping to axis depicts scenarios when the mapping algorithm fails.

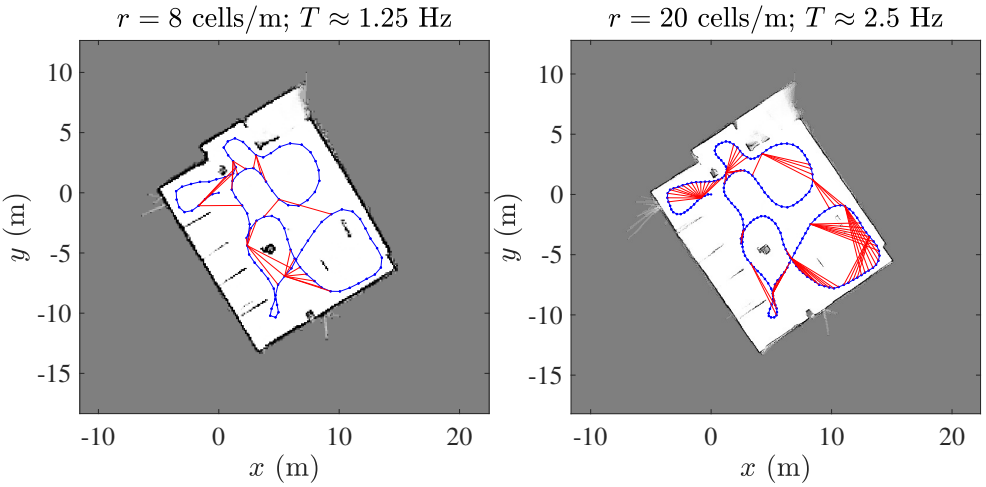
As seen from Figure 4.1, the values of r and T greatly influences the algorithm's performance and computation complexity. It can also be observed that the mapping process fails at low values for r and T. For example, the algorithm generates inaccurate map for $T \approx 1.25$ Hz and $r \geq 14$ cells/m. In general, the parameters r and T are set empirically and usually tuned during offline processing.

Table 4.1 summarizes the details such as the size of map $(m \times n)$ and the total number of scans (N_s) incorporated into each map.

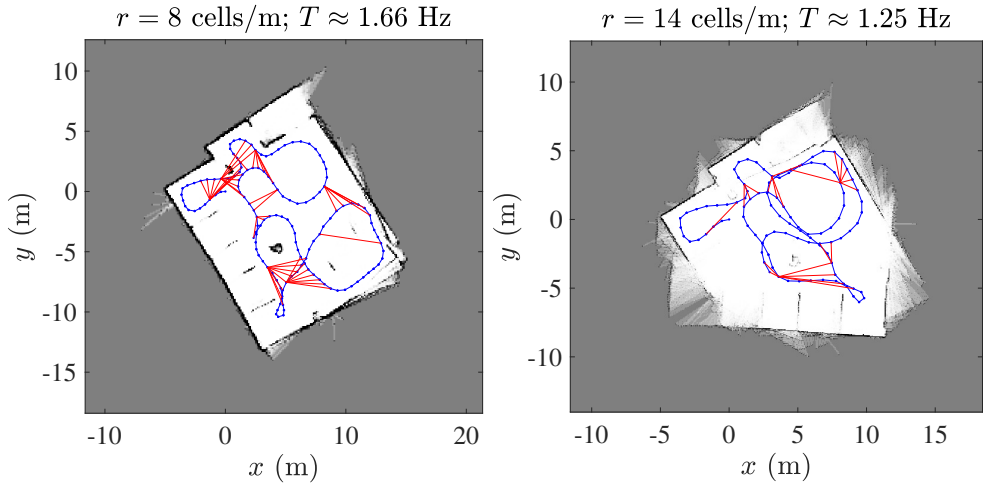
T **(Hz)**	≈ 2.5		≈ 2		≈ 1.66		≈ 1.42	
r **(cells/m)**	N_s	$m \times n$	N_s	$m \times n$	N_s	$m \times n$	N_s	$m \times n$
8	204	248×272	164	248×272	**NA**	**NA**	117	248×272
10	204	310×340	164	310×340	136	310×340	117	310×340
12	204	372×408	164	372×408	136	372×408	117	372×408
14	204	434×476	164	434×476	136	434×476	117	434×476
16	204	496×544	164	496×544	136	496×544	117	496×544
18	204	558×612	164	558×612	136	558×612	117	558×612
20	204	620×680	164	620×680	136	620×680	117	620×680

Table 4.1: **Details of individual maps used for evaluating different feature detection methods.**

Figure 4.2 shows four maps from the map set; two maps for which the values for r and T required least and most computation time, two map failing scenarios for low values of r and T.



(a)



(b)

Figure 4.2: **Occupancy maps along with the optimized pose graph for different values of r and T.** (a) maps with least ($r = 8; T \approx 1.25$) and most computation time ($r = 20; T \approx 2.5$), (b) two inaccurate maps computed for extreme values of r and T.

4.1.2 Evaluation Description

Common feature detection methods are compared to investigate their performance for matching heteregenous grid maps (from Table 4.1). The detectors are assessed

based on keypoint detectability, matching efficiency, computation time, and reliability of estimated map alignment parameters. It also address the choice of transformation model (similarity or rigid) to find the scaling factor between the maps. Table 4.2 presents the different feature detection methods studied in this thesis.

Keypoint Detection Method	**Descriptor**	**Properties**
Binary Robust Invariant Scalable Keypoints (BRISK) [60]	BRISK	corner detection; scale and rotation invariant
Harris-Stephens [30, 45, 53]	**NA**	corner detection; rotation invariant
Shi-Tomasi [52]	**NA**	corner detection rotation invariant
KAZE features [55]	KAZE	corner or blob detection; scale and rotation invariant
Oriented FAST and Rotated BRISK (ORB) [22, 54]	ORB	corner detection; rotation invariant
Scale Invariant Feature Transform (SIFT) [16, 29, 30, 40, 50]	SIFT	blob detection; scale and rotation invariant
Speeded-Up Robust Features (SURF) [30, 51]	SURF	blob detection; scale and rotation invariant

Table 4.2: **Summary of feature detectors used for evaluation.**

4.1.3 Occupancy Matrix

Each map from Table 4.1 is represented as a probability matrix $\mathbf{O}_{\{r,T\}}$ of m rows and n columns. The subscript $\{r, T\}$ is used to identify the maps based on the values of r and T. For example, $\mathbf{O}_{\{r=8,T\approx 2\}}$ refers to the occupancy matrix of the map with resolution 8 $cells/m$ and scan rate approximately as 2 Hz. The obstacle free layer of each matrix $\mathbf{O}_{\{r,T\}}$ is extracted after the thresholding procedure presented in section 3.1. The extracted matrix is denoted as $\mathbf{O}^{free}_{\{r,T\}}$. For simpler representation and better feature detection, both $\mathbf{O}_{\{r,T\}}$ and $\mathbf{O}^{free}_{\{r,T\}}$ are converted to a common matrix form (maximized) in which the occupied cells are set to 1, obstacle-free cells to 0, and the unknown cells to -1.

4.1.3.1 Geometric Transformations in Heteregenous Map Merging

An important problem in feature-based heteregenous map merging is scaling the map information. The scaling factor λ required to align heterogeneous map pair can be computed using the following two ways:

1. **Incorporating the knowledge about r before feature detection to find λ.** The required scaling to align the source matrix $\mathbf{O}_{\{r=s,T\approx n\}}$ to the size of target matrix $\mathbf{O}_{\{r=t,T\approx n\}}$ can be calculated as $\lambda = \frac{t}{s}$. Based on λ, the source image can be scaled using image scaling methods such as bicubic, bilinear or nearest-neighbor interpolation. Now, the problem reduces to finding a rigid transform which consists of only rotation and translation parameters. It must be noted that during interpolation the map information is altered hence it has to addressed during pair-wise integration of grid cells.

2. **Estimating λ based on similarity transform.** The inliers during the RANSAC stage are searched for a similarity transform model. Therefore, λ is considered as a parameter to be estimated. To estimate scaling using similarity transform, the keypoints must be scale-invariant.

4.2 Evaluation Results

This results section is divided into three subsections. Subsection 4.2.1 evaluates different feature detectors based on the number of detectable keypoints. It also highlights the stability of keypoint detectability incorporated by the occupancy map processing technique presented in 3.1. Subsection 4.2.2 compares the matching efficiency of the feature descriptors based on a precision metric. Finally, subsection 4.2.3 assess the reliability of different feature detectors based on inlier cardinality and RMSE.

4.2.1 Feature Detectability

There is always a tight bound on the number of delectable keypoints to avoid unnecessary computations even though it is not qualitative or quantitative performance

measure. For that reason, the total number of keypoints detected by different methods are shown in Figure 4.3. It must be noted that the occupancy matrix $\mathbf{O}_{\{r,T\}}$ is generated without any prepossessing. It is also common to order the detected keypoints based on the strength of response of each keypoint therefore the stronger points can be separated.

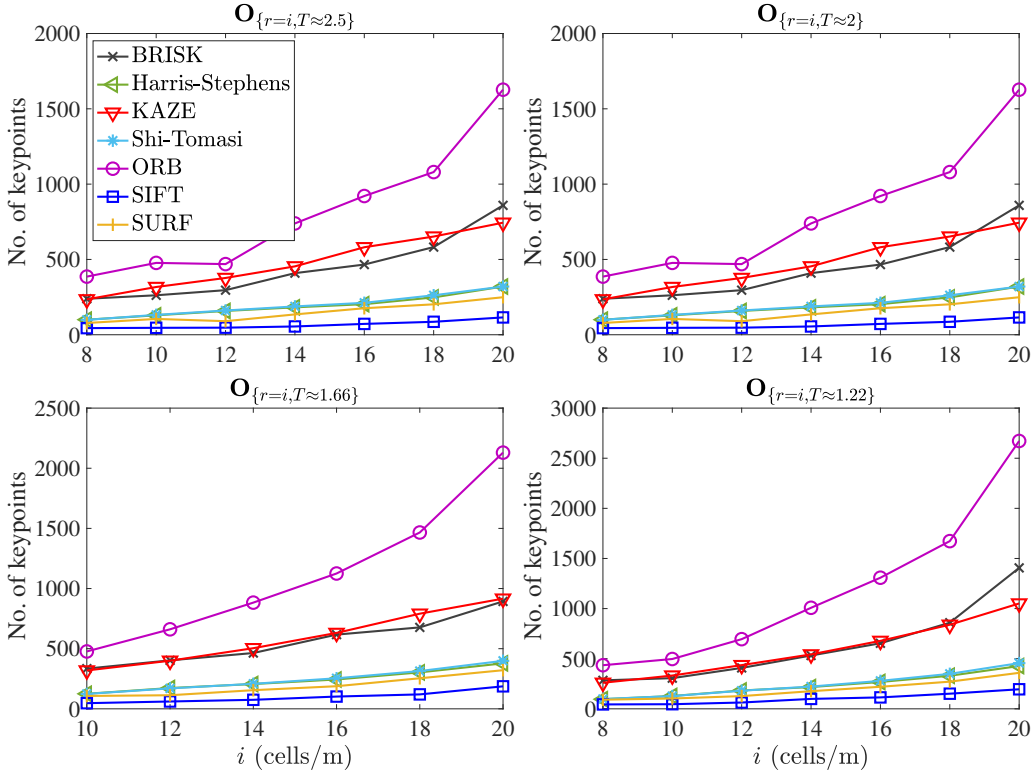


Figure 4.3: **Feature detectability testing.** Extracted number of keypoints by different detectors in map $\mathbf{O}_{\{r,T\}}$ for varying r and T. The keypoints are computed without processing and rescaling the occupancy information.

On the whole from Figure 4.3, a straightforward observation is that there is a substantial difference in the total features detected by each method. Especially, the ORB detector finds the maximum number of features in all maps whereas SIFT detector extracts the minimum number of points. The repeatability score of a keypoint can be only measured during pairwise matching of features. However, in a perfect world scenario (without noise), the change in resolution of an image can be viewed as a change in scale. Therefore, by intuition we expect the so-called scale-invariant

detectors to exhibit consistent number of features across varying resolutions. On that note, the total keypoints detected by SIFT, SURF, Shi-Tomasi, and Harris detectors "look" more consistent for resolution changes when compared to KAZE, BRISK, and ORB methods.

4.2.1.1 Effect of Thresholding Spatial Occupancy on Number of KAZE keypoints

Based on method 1, the prior information about grid resolution can be utilized to calculate the scaling factors between the map pairs. Therefore, the map images can be rescaled before the feature detection stage.

To test the effect of extreme scaling, the factor λ for case 1 is computed to upscale both $\mathbf{O}_{\{r,T\}}$ and $\mathbf{O}^{free}_{\{r,t\}}$ (from) to $r = 20$ $cells/m$ and also downscale to $r = 8$ $cells/m$. The output pixel values in the rescaled images are set to a weighted average of pixels in the nearest 4-by-4 neighborhood based on bicubic interpolation. Figure 4.4 portrays the total number of KAZE features detected after rescaling the occupancy matrices $\mathbf{O}^{free}_{\{r,T\}}$ and $\mathbf{O}_{\{r,T\}}$. It can be seen the number of features detected on the processed map $\mathbf{O}^{free}_{\{r,T\}}$ is more stable.

4.2.2 Matching Efficiency

The repeatability score of keypoints determine the efficiency of feature detectors. It measures the ability of a detection method to extract the same feature points across images irrespective of imaging conditions. A frequent metric used to evaluate the feature detection, extraction, and matching altogether is the inlier precision. The metric is based on the ratio of number of putative matches to the number of inliers obtained from an image pair. The precision of a keypoint detector be calculated as follows [55]:

$$precision = \frac{\#inliers}{\#correspondences} \tag{4.1}$$

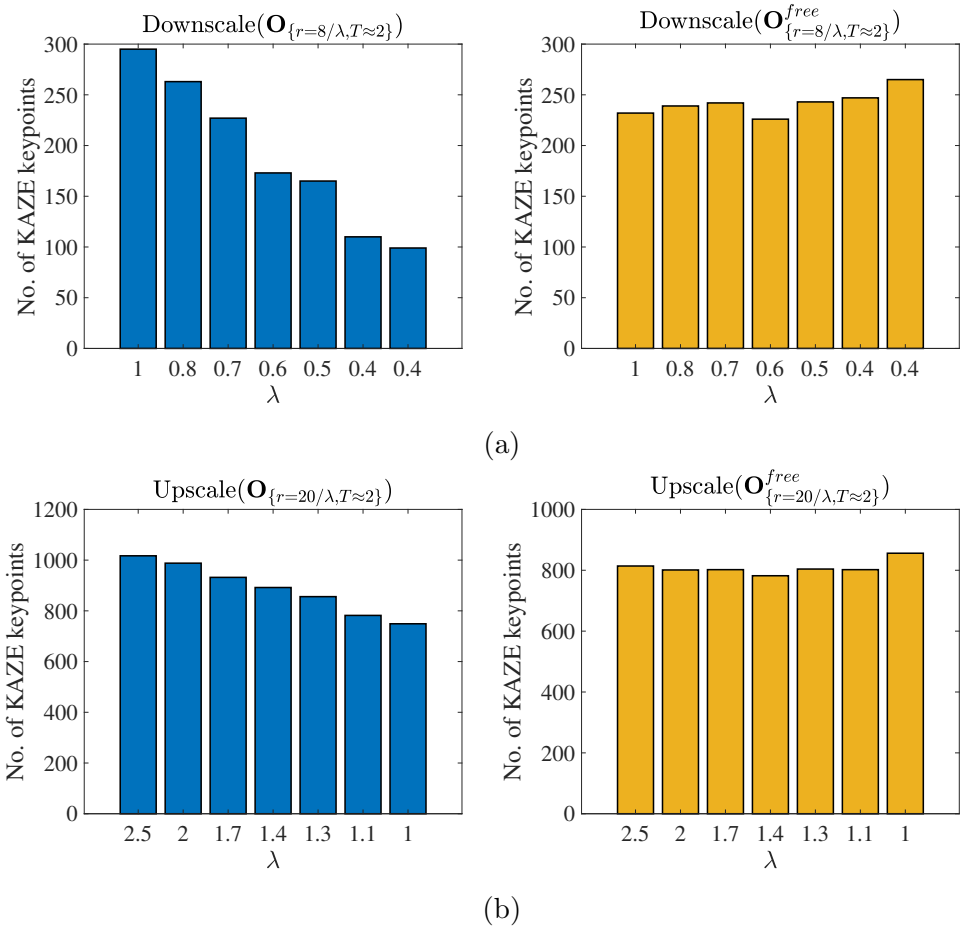


Figure 4.4: **Rescaling heteregenous maps.** Extracted number of KAZE keypoints after scaling $\mathbf{O}_{\{r,T\}}$ and $\mathbf{O}^{free}_{\{r,T\}}$ (a) when each r is down-scaled to 8 cells/m, (b) when each r is up-scaled to 20 cells/m

where the number of correspondences is the number of putative matches during the nearest neighbor matching from section 3.4.

The RANSAC algorithm at least require two correctly matched pairs of features in order to estimate the geometric transformation of interest (similarity or rigid). Higher the number of inlier pairs, better the the accuracy of estimated geometric alignment parameters. Hence, the inlier cardinality and precision from (4.1) during the RANSAC stage is considered as a joint measure to compare the efficiency as well as reliability of different feature detection and description methods.

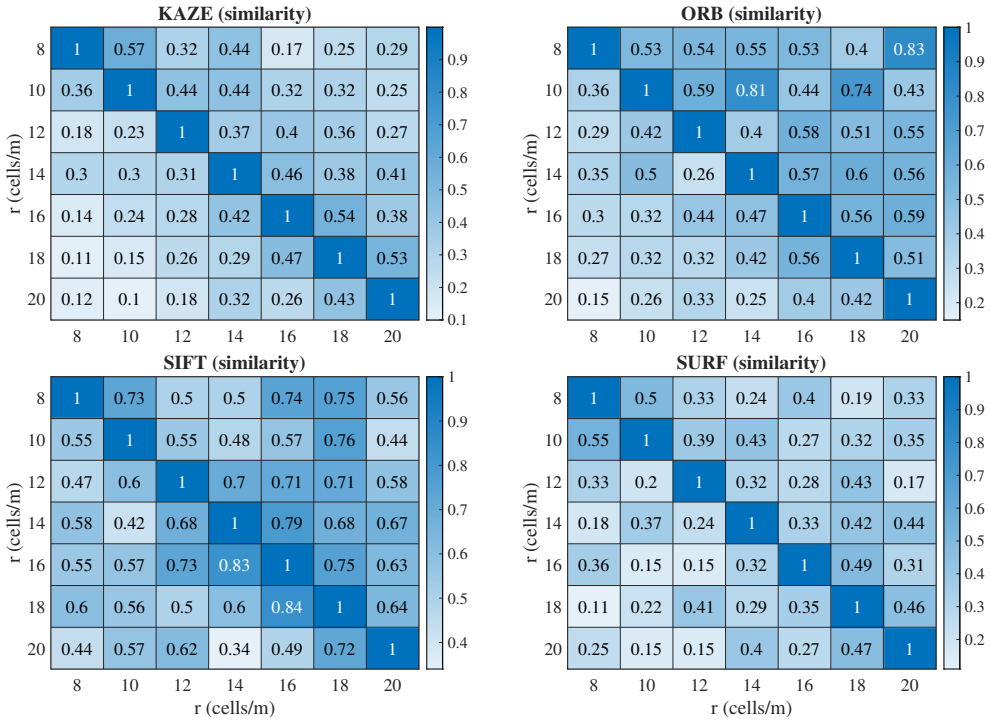
To conduct a fair evaluation, the inliers were computed by matching each map from Table 4.1 for $T \approx 2.5$ with other maps (42 combinations including 7 same resolutions is used). The maximum random trail for finding inliers by the MSAC

algorithm was set to 1000 and the confidence level for acceptance as inlier was set to 99% for all detection methods. The KAZE, SIFT, SURF, and ORB methods always met the minimum keypoints required to calculate the map transformation matrix. Whereas Harris, Shi-Tomasi, and BRISK methods failed to find the required inliers for various cases hence they are not ignored for the rest of the evaluation.

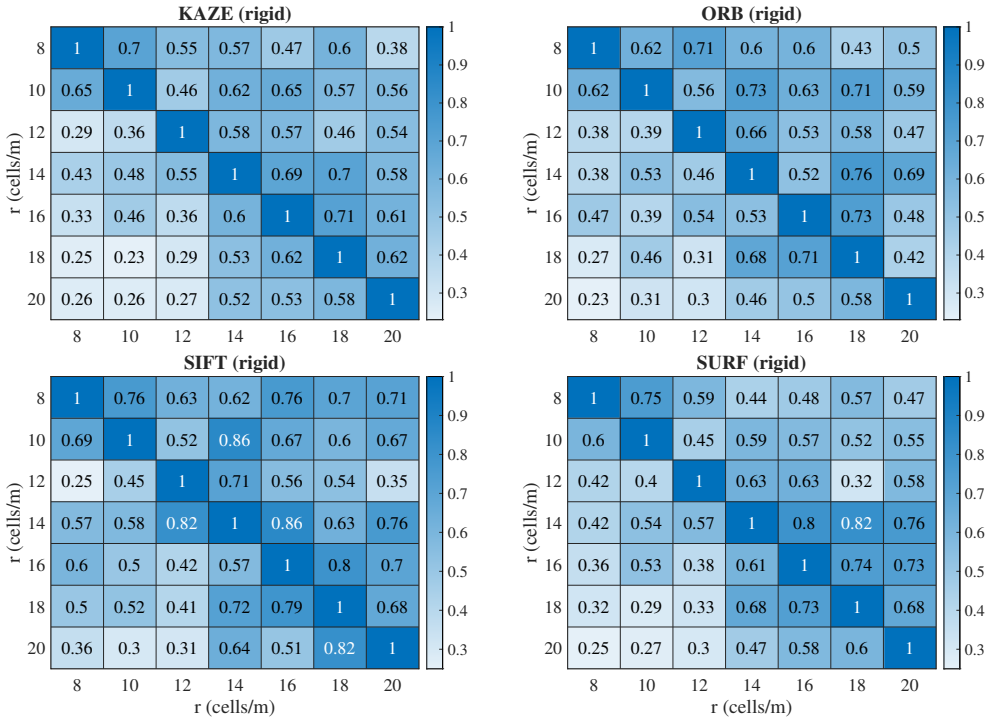
Figure 4.5 depicts the precision of inliers computed for one run of the MSAC algorithm. The precision was calculated for matching the maps at different resolutions using the rigid transform model (method 1) as well as the similarity transform model (method 2). Similar to Figure 4.5, the inlier cardinality were also calculated for both similarity and rigid transform model. Figure 4.6 illustrates the total inliers returned by the MSAC algorithm for KAZE, ORB, SIFT, and SURF methods.

As the resolution differences across two maps increases, the overall performance of the feature matching is drastically reduced. The SIFT detector and descriptor exhibited excellent precision (Fig.4.5a, 4.5b) for both method 1 and 2 when compared to KAZE, ORB, and SURF. In terms of inliers, KAZE method returned good number of inliers for rigid transform model (Fig. 4.6b). Whereas the overall inliers computed for the similarity model is poor for all detectors and descriptors (Fig. 4.6a).

On th whole from Figure 4.5 and 4.6 , it can be observed that addressing the heterogeneous map merging problem using the rigid transform model yield to better precision and returns more inliers when compared to the similarity transform model. However, the effect of interpolation required for the rigid transform has to be taken into account while fusing the grid information.

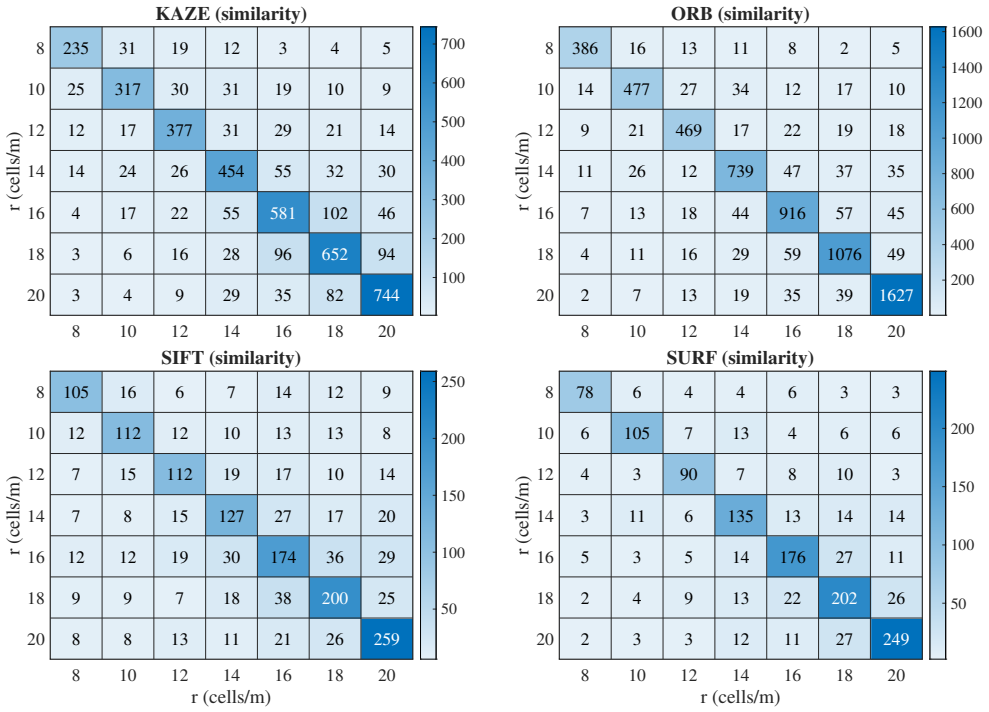


(a) Heteregenous resolution map merging as similarity transform problem.

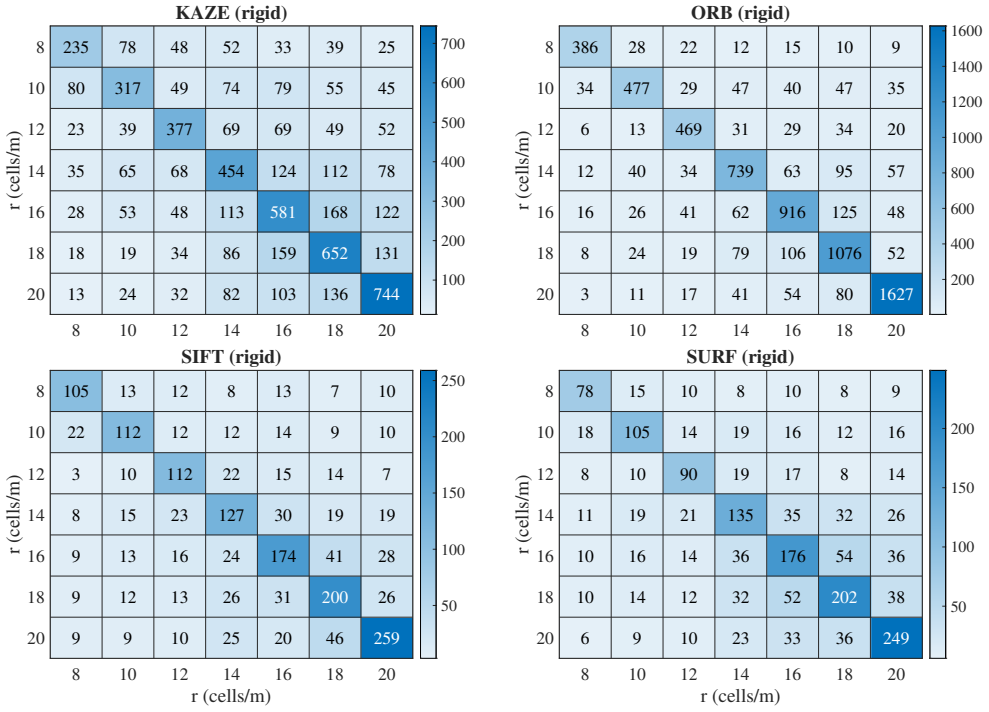


(b) Heteregenous resolution map merging as rigid transform problem.

Figure 4.5: **Inlier precision of KAZE, ORB, SIFT, and SURF for nearest neighbor matching.** The diagonal entries in each sub-figure indicates matching the map with itself (number of matched points = number of inliers).



(a) Heteregenous map merging as similarity transform problem.



(b) Heteregenous map merging as rigid transform problem.

Figure 4.6: **Inlier cardinality returned by the MSAC algorithm for KAZE, ORB, SIFT, and SURF methods.** The diagonal entries in each sub-figure indicates matching the map with itself (number of detected keypoints = number of inliers).

4.2.3 Transformation Reliability

The results may not be exactly identical between runs because of the randomized nature of the MSAC algorithm. Therefore, the RMSE [61] is used to compare the reliability of different feature detectors based on the alignment error. The RMSE is generally used to measure the difference between the estimated and the true parameter of interest. Here, it is used to different compare feature detectors based on their alignment error caused due to rotation. The RMSE of MSAC with respect to the rotation θ can be calculated as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(\theta - \hat{\theta})^2} \tag{4.2}$$

where $\hat{\theta}$ indicates the estimated rotation parameter and N denotes the number of Monte Carlo runs. The reason for only using rotation but not translation is that the rotation component of the transformation is the hardest to recover [15].

In order to compute the RMSE, knowledge about true rotation θ is required. Hence, the error is computed by applying random rotation to the occupancy maps and recovering the applied rotation. The estimated rotation $\hat{\theta}$ can be recovered as

$$\hat{\theta} = \text{atan2}(\lambda \sin\hat{\theta}, \lambda \cos\hat{\theta}) \tag{4.3}$$

where λ is the scaling factor.

Figure 4.7 and 4.8 illustrates the RMSE and the average computation time calculated over 100 runs for matching heteregenous maps with 100% overlap. The source image for each run was randomly rotated and interpolated using nearest neighbor interpolation technique. For each rotated source map, the $\hat{\theta}$ was recovered based on 4.3. The average time for each feature detector and descriptor was calculated as

$$Avg.time = t_{detection} + t_{description} + t_{mathcing} + t_{MSAC} \tag{4.4}$$

where $t_{detection}, t_{description}, t_{mathcing}, t_{MSAC}$ are the time taken for detecting the features,

describing the detected features, matching the descriptors using the nearest neighbor criterion and fitting the inliers using the MSAC algorithm.

On the whole, the KAZE method demonstrated minimal rotation error compared to other methods. However, it is the slowest in terms of computation time. The RMSE shown by SIFT method was high, and it is also less reliable at times as it failed to estimate the rotation parameter between runs. The SURF and ORB methods were fast (SURF was fastest by a small margin) out of which the SURF had better error performance.

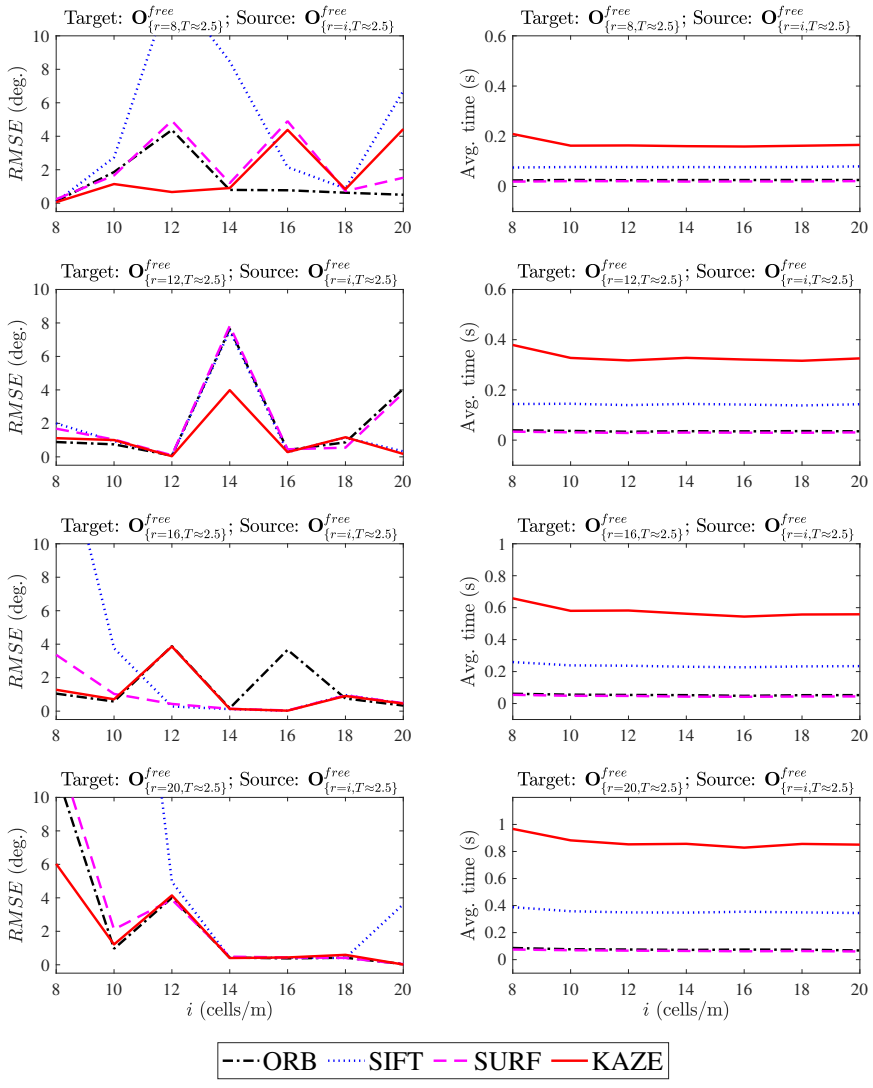


Figure 4.7: **RMSE and average feature matching time for change in resolution computed over 100 runs.**

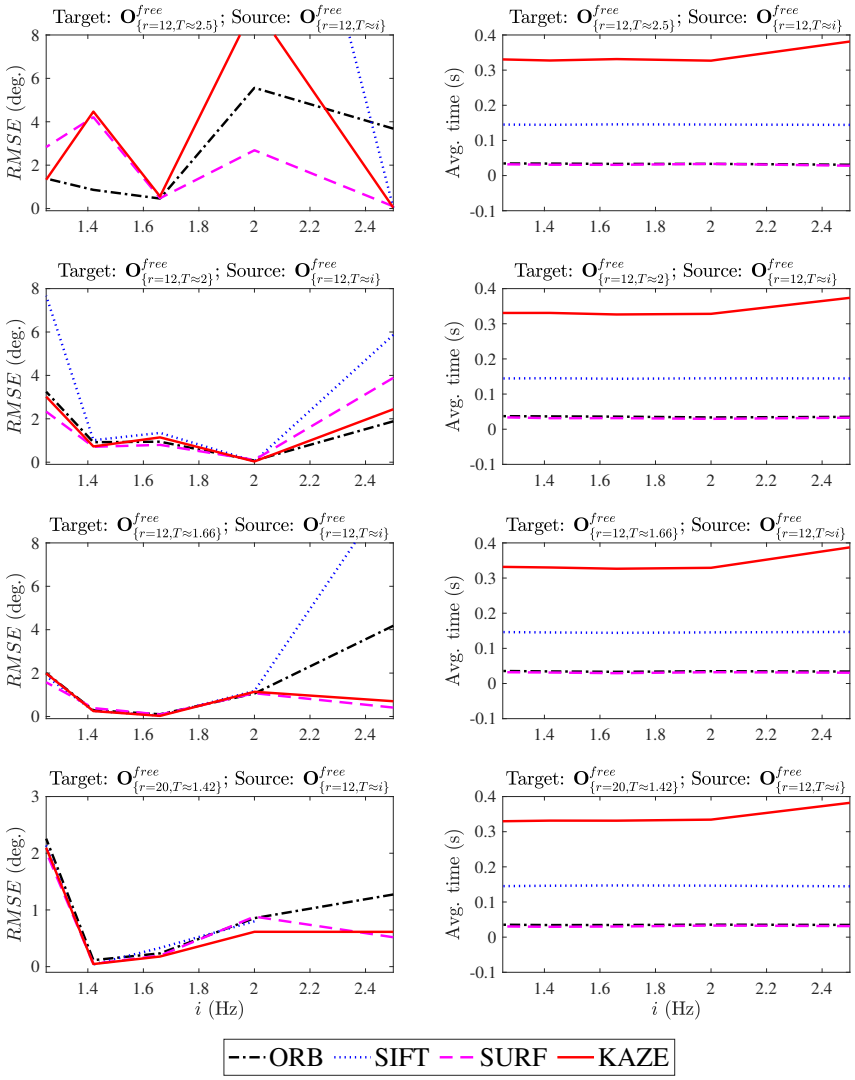


Figure 4.8: **RMSE and average feature matching time for change in sample rate computed over 100 runs.**

4.3 Summary

The chapter compared well-known feature detector such as Harris, Shi-Tomasi, BRISK, ORB, KAZE, SIFT and SURF based on their detection and matching performance for heteregenous map merging purpose. The findings of this chapter is summarized as follows:

Harris, Shi-Tomasi, BRISK: Even though these methods were able to detect sta-

ble number of corner features, the detected keypoints failed to establish consistent matches across heteregenous maps. However, they might still be a good choice for feature-matching across homogeneous maps.

ORB: feature detection and extraction method was the second fastest next to SURF with the best detectability. However, it showed poor matching efficiency and returned fewer number of inliers. The overall RMSE was also high when compared to KAZE and SURF.

KAZE: method was the slowest compared to all the other methods studied in this chapter. But, it has shown best performance with respect to error and number of inliers. Therefore, KAZE is highly reliable and comparable accuracy when compared to other methods at the cost of additional computation time.

SIFT: exhibited best matching efficiency based on inlier precision. However, the number of keypoints detected and the obtained number of inliers were less compared to other methods. The SIFT was also poor in terms for reliability as it did not compute the required alignment parameters in between the Monte Carlo runs.

SURF: was the fastest when compared to other methods. It also showed comparable accuracy during RMSE testing.

Chapter 5

EXPERIMENTAL RESULTS

In this chapter several real-world tests are conducted to evaluate the effectiveness of the proposed approach to integrate occupancy maps acquired during collaborative exploration. Section 5.1 describes the underlying experiment assumptions and the single-robot SLAM algorithm. Section 5.2 presents the result of a simple collaborative real-world mapping experiment using two homogeneous mobile robots. Section 5.3 illustrates the result heterogeneous map merging for different resolutions. Section 5.5 presents a hierarchical large-scale map fusion implementation to fuse six different maps of the same environment obtained from individual robots. Lastly, Section 5.6 and Section 5.7 shows merging using of maps obtained from heterogeneous sensors and a motivating application of path planning using the obtained global maps.

5.1 Experiment Setup

Real-world experiments were conducted using multiple QCars. The Qcars were remotely controlled and the environment was scanned using the laser scanner of the individual QCars. The graph SLAM [1, 37, 38] algorithm was utilized to build offline local maps. The graph SLAM incrementally processes the scans to build a pose graph that links the scans. Each node in the graph is connected by an edge constraint that relates the poses between the nodes and represent the uncertainty in that measurement. Whenever two regions of the map are found to be alike in the world irrespective

of the robot pose history, the system calculates a relative transformation that aligns the two regions to *close the loop*. The algorithm utilize the loop closure information to update the map and optimize the estimated pose trajectory.

The map fusion stages are summarized in Algorithm 3. The inputs are two local maps ${}^{a}\mathbf{m}$ and ${}^{b}\mathbf{m}$ and output is the global map ${}^{G}\mathbf{m}$. All the map fusion computations were performed using the MATLAB software on a central CPU core processor running at 3.60 GHz with 32 Gb of RAM.

Algorithm 3 $[{}^{G}\mathbf{m}] = \mathbf{mapFusion}({}^{a}\mathbf{m}, {}^{b}\mathbf{m})$

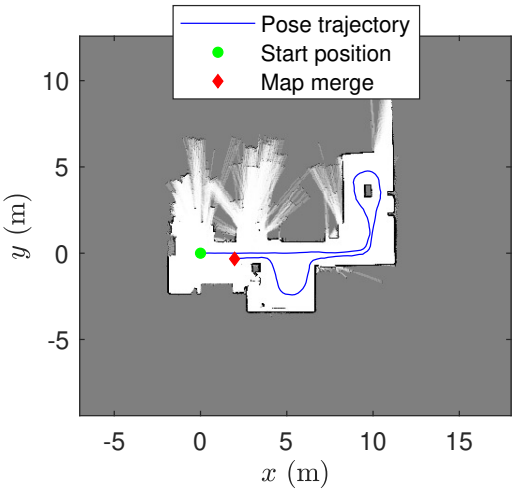
Process the maps to obtain occupancy images I_a^{free}, I_b^{free} based on section 3.1:
$[{}^{a}I, {}^{b}I] = \mathbf{processOccupancyMaps}({}^{a}\mathbf{m}, {}^{b}\mathbf{m})$
Detect KAZE keypoints k_a, k_b based on section 3.2:
$[k_a, k_b] = \mathbf{detectKAZEfeatures}({}^{a}I, {}^{b}I)$
Describe the detected features using the SIFT descriptor described in section 3.3:
$[f_a, f_b] = \mathbf{SIFTdescription}({}^{a}I, {}^{b}I, k_a, k_b)$
Find the nearest-neighbors based on section 3.4:
$[\mathcal{M}\{f_a^i, f_b^j\}] = \mathbf{featureMatching}(f_a, f_b)$
Compute the transformation $\mathbf{T}$ using the MSAC algorithm described in 3.5:
$[\mathbf{T}] = \mathbf{outlierElimination}(\mathcal{M}\{f_a^i, f_b^j\})$
Verify the transformation, and update the global map $\mathbf{m}_G$ based on grid fusion methodology presented in section 3.6:
$[{}^{G}\mathbf{m}] = \mathbf{gridFusion}({}^{a}\mathbf{m}, {}^{b}\mathbf{m}, \mathbf{T})$

In order to accommodate the integrated global occupancy information, a gird matrix is initialized with the cell probability values set to 0.5 (unknown). The dimensions of the global grid matrix is set as a function of the size of the individual maps.

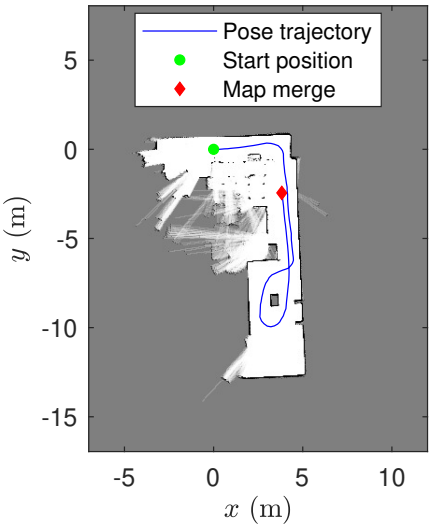
5.2 Homogeneous Map Fusion

The first experiment assumes a scenario of two robots performing collaborative mapping of a simple indoor environment. This demonstration aims to highlight the robustness of Algorithm 3 while merging maps with low overlapping area. Especially, the experiment points out the ability of the algorithm to find geometrically consistent feature-correspondences across homogeneous and heteregenous grid resolutions.

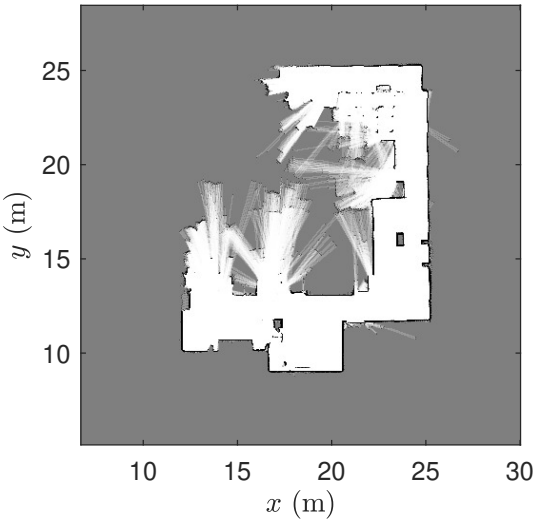
In this experiment, the instance of map merging is manually triggered i.e. we assume that the local maps are ready to be merged and received at the central computer. Figure 5.1a, 5.1b depicts the two overlapping local grid maps ${}^{a}\mathbf{m}$ and ${}^{a}\mathbf{m}$ of same resolutions obtained from the individual QCars (ready for map fusion). Figure 5.1c shows the resulting global map ${}^{G}\mathbf{m}$ obtained after transforming ${}^{b}\mathbf{m}$ to ${}^{a}\mathbf{m}$ and then to global coordinates using the Algorithm 3.



(a) Local map, ${}^{a}\mathbf{m}$



(b) Local map, ${}^{b}\mathbf{m}$



(c) Merged global map, ${}^{G}\mathbf{m}$

Figure 5.1: **Demonstration of merging two occupancy maps with same grid resolutions** ($r_a = r_b = 20$ cells/m).

5.3 Map Fusion for Heterogeneous Grid Resolutions

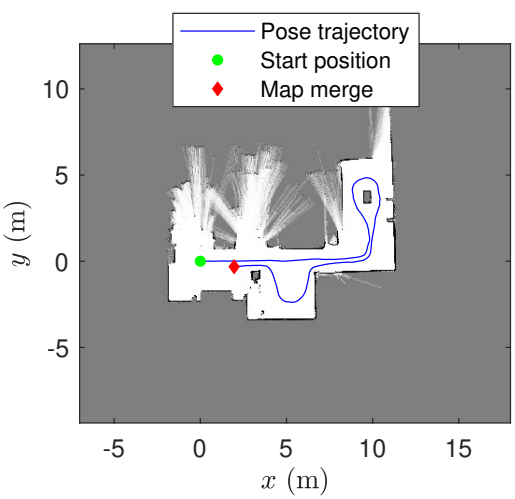
Next, we demonstrate merging maps of different grid resolutions. The main advantage of using heterogeneous maps is the flexibility. It helps to reduce the overall exploration and computation time. Further, it allows to share useful environment model information among different levels of robots.

To merge maps at different scale, we require the scaling factor in addition to the rotation and translation. Luckily, the gird resolution of the occupancy map is an user-defined variable. Hence, it can utilized to compute the scale change between the moving map (map to be merged) and the fixed map. Based on the scale change, the moving map can be rescaled to the same size as the fixed map using nearest-neighbor interpolation. It should be noted that interpolating map images may alter the information present at the pixel level. Therefore, to reduce the effect of interpolation, only the processed obstacle-free layer of the map is rescaled (i.e. the original map is not scaled). Now, using the Algorithm 3 we estimate the parameters for the rigid transform matrix. Then, we define the similarity matrix using the known scaling factor and the estimated rotation and translation. Finally, we apply the computed similarity transformation to the original moving map even though we fit the rigid model to estimate the parameters.

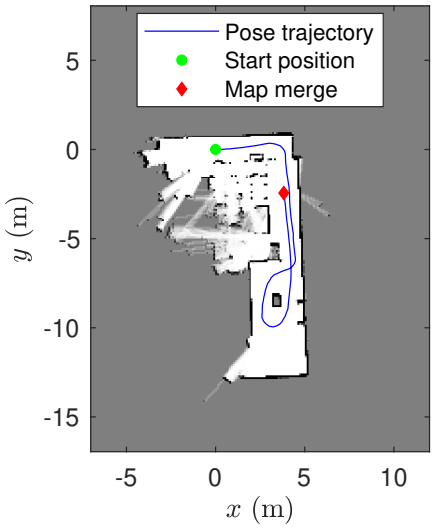
In our case, lets assume that ${}^{a}\mathbf{m}$ is the fixed map and ${}^{b}\mathbf{m}$ is the moving map. Similar to previous demonstration, Figure 5.2a, 5.2b shows the two local maps but with different grid resolutions. It must be noted that, the scaling difference considered here is large. Even then, we were able to obtain good number of consistent matches to estimate the transformation. Figure 5.2c shows the global map obtained as a result of transforming ${}^{b}\mathbf{m}$ to ${}^{a}\mathbf{m}$ and then to global coordinates.

In Figure 5.3, the pixel locations of the occupied cells are overlaid for three resulting global maps. For all three cases, the fixed map is ${}^{a}\mathbf{m}$ with resolution $r_a = 25$ cells/m. Whereas we consider three moving maps ${}^{b}\mathbf{m}$ of different resolutions $r_b = 10, 20, 25$ cell/m (i.e. one homogeneous and two heterogeneous merging). It can

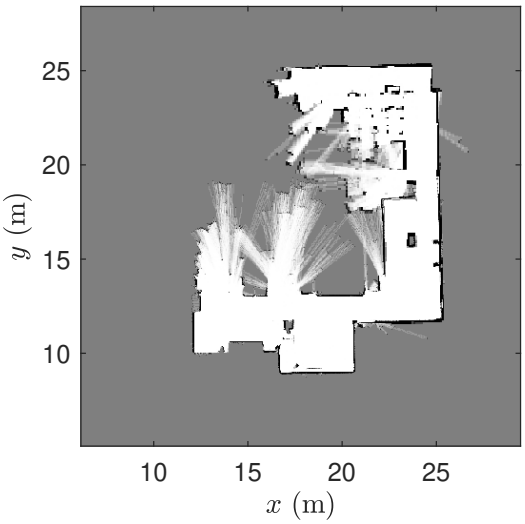
be clearly observed that for three cases we obtained a consistent transformation to transform ${}^{b}\mathbf{m} \rightarrow {}^{a}\mathbf{m}$.



(a) Local map, ${}^{a}\mathbf{m}$



(b) Local map, ${}^{b}\mathbf{m}$



(c) Merged global map, ${}^{G}\mathbf{m}$

Figure 5.2: **Demonstration of merging two occupancy maps with different grid resolutions** ($r_a = 25, r_b = 10$ cells/m).

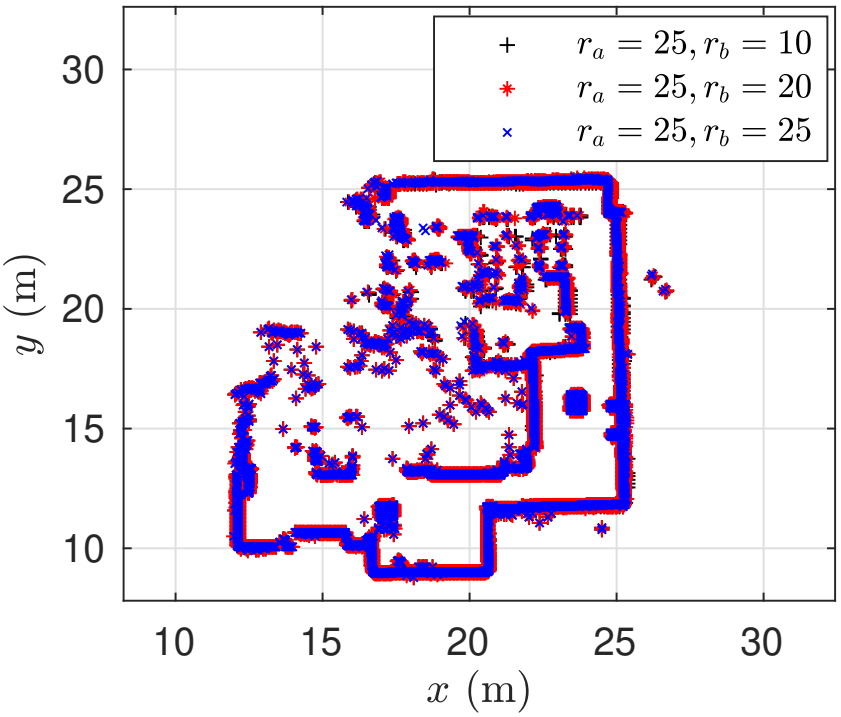


Figure 5.3: **Occupied pixel locations of three global maps obtained after merging maps with different grid resolution.** Each map is obtained as a merging result between a fixed map ${}^{a}\mathbf{m}$ of resolution $r_a = 25$ cell/m and a moving map ${}^{b}\mathbf{m}$ of resolutions $r_b = 10$, 20, 25 cells/m.

5.4 Performance of Map Merging

Due to the randomness involved in the MSAC algorithm, the estimated rotation and translation might vary for different runs. Similarly, parameters such as acceptance index ω, execution time and the cardinality of the largest inlier set also varies. Hence for a meaningful interpretation, we summarize the performance of map fusion in Table 5.1 by computing the average and deviations over 100 runs.

Table 5.1: **Mean and deviation of performance parameters while merging maps of same and different gird resolutions.** The results are calculated for the transformation ${}^{a}\mathbf{m}_b$ over 100 runs.

${}^{a}\mathbf{m}$	${}^{b}\mathbf{m}$	${}^{G}\mathbf{m}({}^{a}\mathbf{m}, {}^{a}\mathbf{T}_b({}^{b}\mathbf{m}))$				
r_a (cells/m)	r_b (cells/m)	$\omega({}^{a}\mathbf{m}, {}^{b}\overline{\mathbf{m}})$	Wall-clock time (sec)	$\mid inliers \mid$	Rotation (deg.)	Translation (m)
25	25	$0.98 \pm 2.3 \times 10^{-4}$	$0.49 \pm 7.7 \times 10^{-3}$	33	± 0.0472	± 0.9274
20	20	$0.97 \pm 5.1 \times 10^{-4}$	$0.32 \pm 4.9 \times 10^{-3}$	35	± 0.0935	± 1.4873
10	10	$0.95 \pm 2 \times 10^{-3}$	$0.10 \pm 1.6 \times 10^{-3}$	9	$\pm 1.1 \times 10^{-7}$	$\pm 3.06 \times 10^{-5}$
25	20	$0.97 \pm 7.8 \times 10^{-4}$	0.48 ± 0.0113	28	± 0.1046	± 1.5863
25	10	$0.93 \pm 9.8 \times 10^{-4}$	$0.47 \pm 7.2 \times 10^{-3}$	25	± 0.2161	± 1.6717
20	10	$0.94 \pm 3.2 \times 10^{-3}$	$0.31 \pm 5.6 \times 10^{-3}$	22	± 0.2535	± 1.9222

An important aspect of the map merging problem is when to merge the maps. To decide the merging instance, the cardinality of the largest feature correspondences and the acceptance is jointly utilized. In Figure 5.4, we assume that two occupancy maps are obtained for approximately every 1 sec as the map is being constructed by the single-robot SLAM algorithm. For each time index, the number of feature correspondences and the acceptance index based on the respective transformation is calculated. It can be clearly observed that, both inlier cardinality and accpetance index produces false positives (i.e. for time index < 40). However, when the true transformation is found the relative change between the successive time indexes are reduced. This, can be used to decide the instance of merging. It is important to note that for making such decisions, the maps has to be exchanged but not necessarily at fixed intervals.

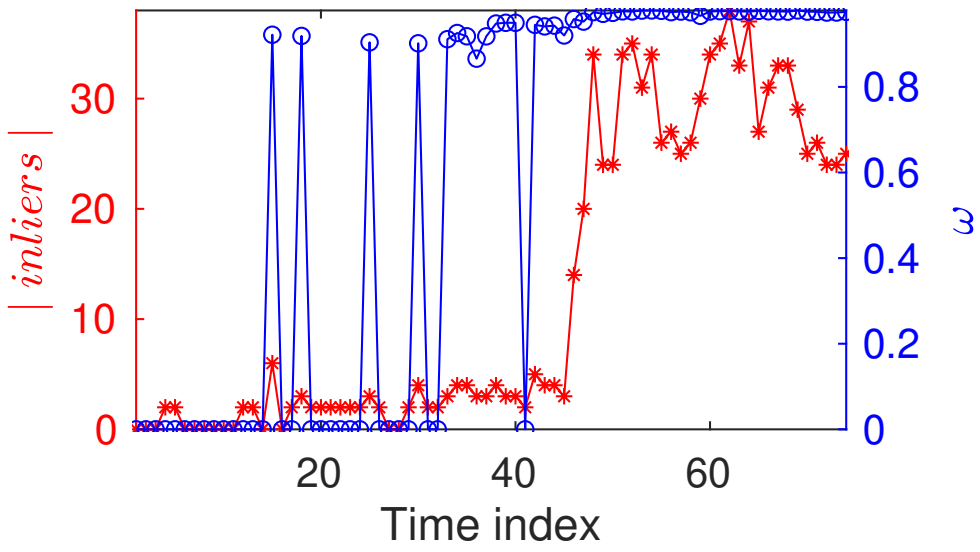


Figure 5.4: **Behavior of the cardinality of inlier set and the acceptance index for accepting a transformation.** The plot is shown for $r_a = 25$ and $r_b = 25$ cells/m.

5.5 Hierarchical Map Fusion

If the maps obtained from the robots a and b can be merged, similarly c and d can be merged, then find a transformation (if exists) between a, b, c, d for which all

four maps can be merged. This excellent property exhibited by the collaborative systems is known as transitivity [11]. To fully utilize the advantages of a multi-agents, the transitivity nature of the system should be exploited. Thus, the robots in an interaction mode can form exploration clusters in which they can coordinate their actions [11].

In this section, the map of large environment is reconstructed using six local maps provided by the individual robots. The motive of the experiment is to highlight the flexible transitive implementation of the presented method in the presence of heterogeneity. The individuals maps are overlaid on the building floor plan (test environment) in Figure 5.5.

Figure 5.5: **Six heteregenous local maps obtained from different robots overlaid on the CEI building floor plan.**

The individual robots explored the environment at different time interval. This assumption is common when deploying multiple robots in an environment as they are put into operation individually [19]. Each map differ from other maps in terms of grid resolution and the rate at which the scans are fed to the mapping algorithm. Thereby assuming a heterogeneous mapping scenario. In the central computer, all six maps are provided at once. The Algorithm 3 was modified to systematically

compare possible map pairings (out of 15 total combinations) to find the best candidate pairs based on their inlier strength and the acceptance index. Then the local maps are fused in a hierarchical fashion to construct the global map. The method implemented here to find the best candidate pairings is exhaustive. For efficient implementation, the exhaustive search model to find the best pair should be replaced with a better technique. Figure 5.6 shows the map fusion hierarchy to combine all six individual maps into one consistent global map. The executed order of merging is as follows: ${}^{a}\mathbf{m}, {}^{b}\mathbf{m}, {}^{c}\mathbf{m}, {}^{d}\mathbf{m}, {}^{e}\mathbf{m}, {}^{f}\mathbf{m} \rightarrow \{{}^{a}\mathbf{m}, {}^{b}\mathbf{m}\}, \{{}^{c}\mathbf{m}, {}^{e}\mathbf{m}\}, \{{}^{d}\mathbf{m}, {}^{f}\mathbf{m}\} \rightarrow \{{}^{a}\mathbf{m}, {}^{b}\mathbf{m}, {}^{c}\mathbf{m}, {}^{d}\mathbf{m}, {}^{e}\mathbf{m}, {}^{f}\mathbf{m}\}$

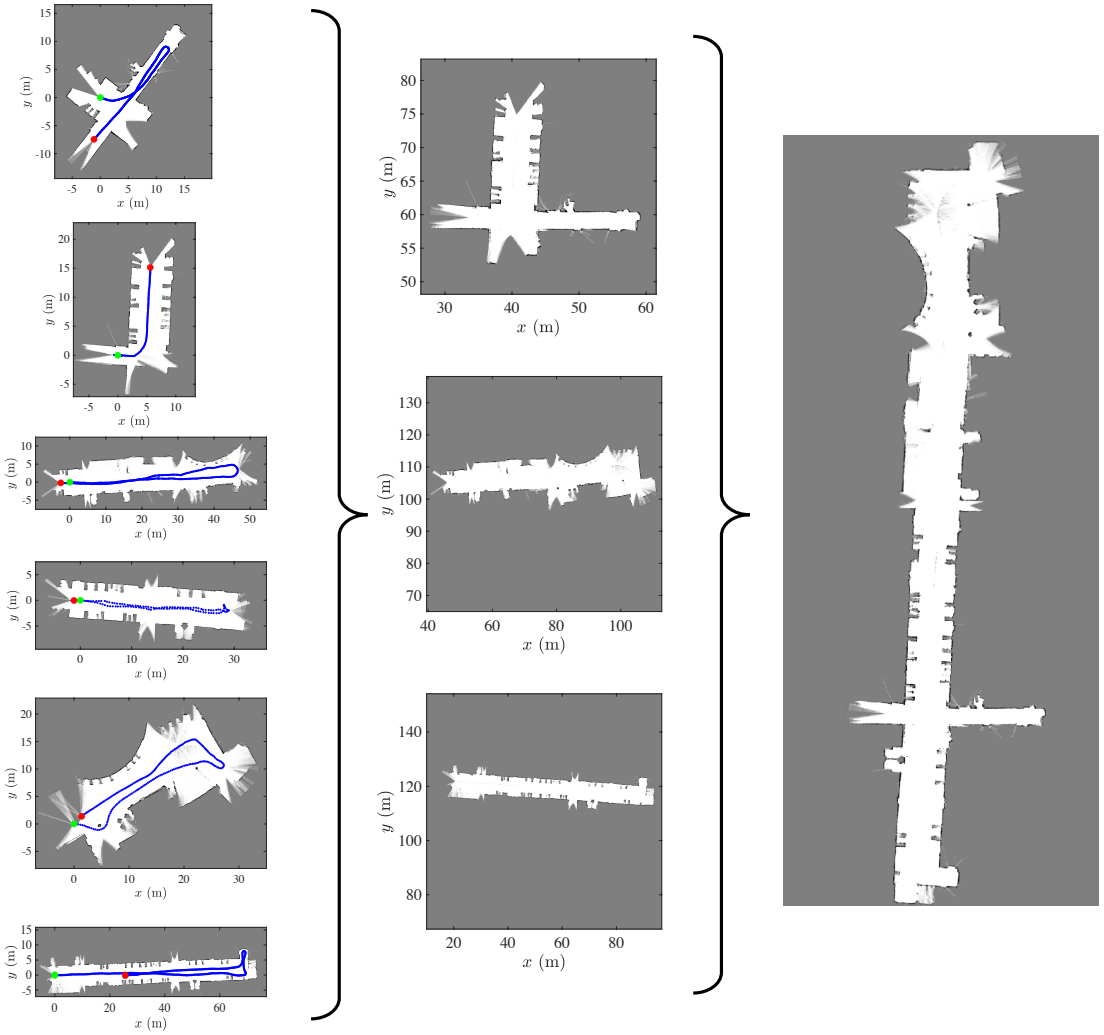


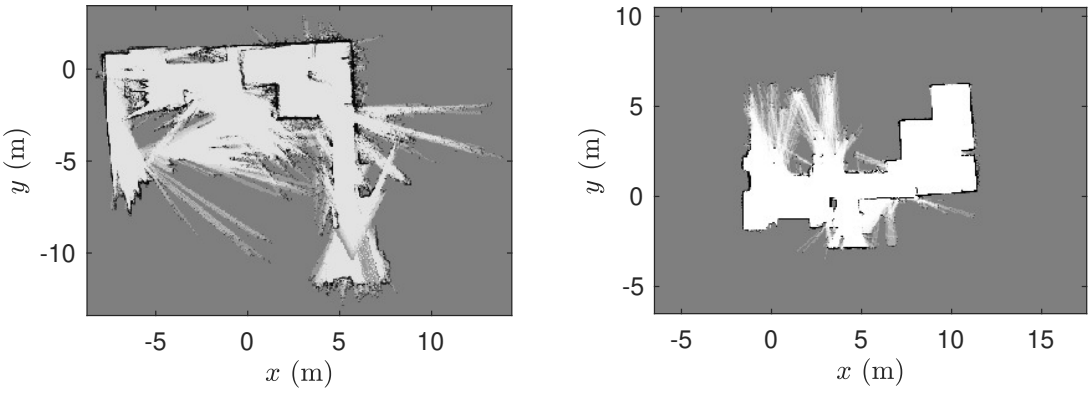
Figure 5.6: **Map fusion hierarchy.** The six heterogeneous local maps obtained from individual robots are hierarchically merged into a consistent global map.

Finally, the Figure 5.7 displays the obtained global map overlaid on the building blueprint to qualitatively interpret the alignment results. It must be emphasized that the single-robot SLAM was unable to construct the complete map due to accumulated errors in the system. if it all, it consumed huge amount of memory and time to finish the map.

Figure 5.7: **Global map obtained after hierarchical map fusion overlaid on the CEI building blueprint.**

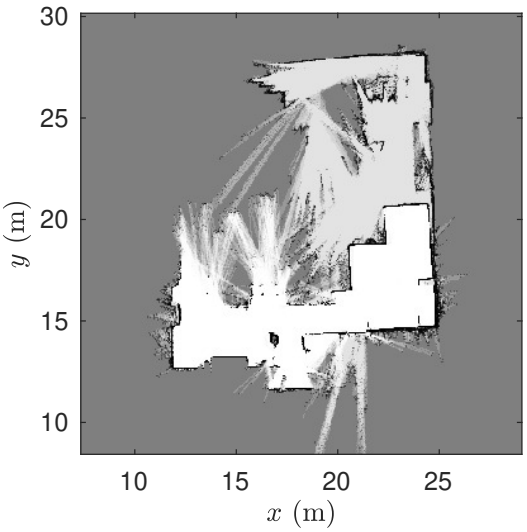
5.6 Map Fusion for Heterogeneous Sensors

In this experiment, we assume that the robots in collaboration uses heterogeneous sensors to map the environment. Figure 5.8a depicts the 2D occupancy map obtained from robot a which was constructed using depth measurements from RGB-D camera. Figure 5.8b depicts the map obtained from robot b which was constructed using LIDAR sensor. Figure 5.8c shows the merged global map obtained as result of heterogeneous map fusion.



(a) Camera map, ${}^{a}\mathbf{m}$

(b) LIDAR map, ${}^{b}\mathbf{m}$



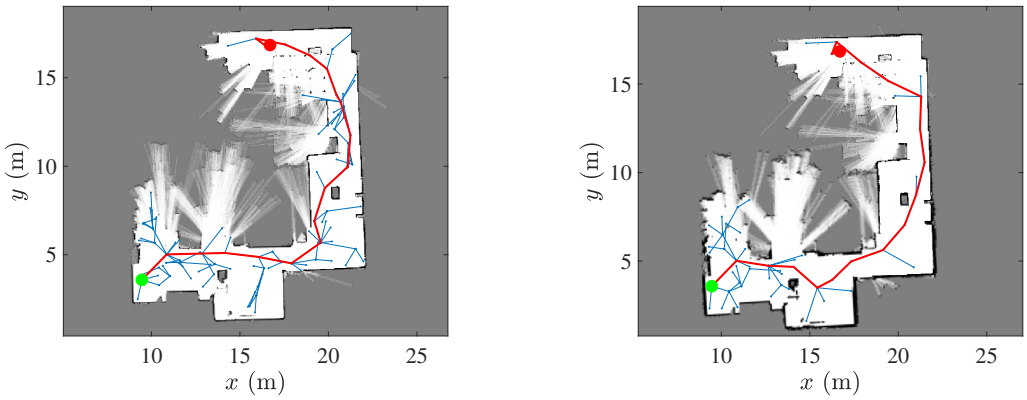
(c) Merged global map, ${}^{G}\mathbf{m}$

Figure 5.8: **Demonstration of merging two occupancy maps obtained from different sensors.**

5.7 Motion Planning

One of the important applications of robotic maps is robot motion planning. A motivating experiment is conducted to plan the motion for a robot using rapidly exploring random tree (RRT) algorithm [62]. The motive is to find a collision-free path for the provided start and end point location in the given map. The RRT

algorithm samples random states within the state space and attempts to connect a trajectory. These states and connections are validated or excluded based on the map constraints. Figure 5.9 illustrates the path planning results computed using the two global maps obtained as a result of map fusion.



(a) (b)

Figure 5.9: **Path planning using the RRT algorithm.** (a) Homogeneous global map, (b) Heterogeneous global map

Chapter 6

CONCLUSION AND FUTURE WORKS

This chapter presents the summary of the thesis. Then future research related to the problems in the area of collaborative mapping is discussed.

6.1 Summary

This thesis presented a novel feature-based occupancy map fusion approach for collaborative mapping. The unknown initial correspondence problem that occurs during map fusion is solved using the following stages: processing spatial probabilities to obtain refined maps, detecting features using KAZE feature detectors, describing the detected features using SIFT description, matching features using nearest neighbor matching, and estimating the map transformation parameters using the MSAC algorithm. The certainty grid fusion problem is solved using Bayesian inference.

The process of map fusion tends to be challenging due to the inabilities of the existing methods to find adequate valid correspondences across different mapping conditions. One of the advantages of the presented method is that it can always establish geometrically consistent feature correspondence, provided the single-robot SLAM solution is robust.

This thesis also provided a comparative study on well-known feature detection

methods such as BRISK, Harris, Shi-Tomasi, KAZE, ORB, SIFT, and SURF for the problem of merging maps at different scales. The provided results and evaluation procedures can be useful in the selection of the best feature detector while registering grid maps or similar applications.

Finally, the effectiveness of the proposed approach is validated based on real-world data for scenarios such as merging homogeneous maps, heterogeneous resolution maps, and maps generated using heterogeneous sensors. Further, we also illustrated the suitability of the presented approach to perform hierarchical map fusion.

6.2 Future Work

- **Multiple-robot pose graph optimization:** Recently, many researches focuses on multiple robot optimization methods (constraints between two multiple-robot pose graph). This thesis did not address this problem because here the primary motive was to merge heterogeneous data which means the single-robot can even use different type of SLAM solutions.

- **Multiple-robot active SLAM:** For an efficient implementation of collaborative SLAM using robot swarms [63], the active SLAM problem for the single-robot SLAM should be solved [64]. This could enable many real-time collaborative applications such as coordinated exploration,

- **3D occupancy maps:** Most of the existing work focuses on 2D occupancy maps (including this thesis). The 3D map fusion is a emerging field of research however it involves handling and processing large volumes of data in real-time. Therefore, the scalability and network bandwidth of the system could be challenging.

- **Heterogeneous vehicles:** The use of different type of mobile platforms (aerial and ground vehicles) in a multi-robot system has many applications. For instance, a ground robot may see features of the environment that a aerial cannot, and at the same time, a aerial may have access to different areas that a ground

robot does not. However, for such type of map fusion the overlapping assumption might not be feasible.

- **Decentralized approach:** Centralized decision-making methods have the latent problem that if the central administrator goes out of operation, the cooperative task cannot be carried out. Decentralized methods solve this limitation as each agent acts individually, interacts with the other agents when it is possible and decides on its own how to contribute to the collaborative reconstruction task. However, here we face a new main problem which is real-time decision-making.

Bibliography

[1] Cyrill Stachniss, John J Leonard, and Sebastian Thrun. Simultaneous localization and mapping. In *Springer Handbook of Robotics*, pages 1153–1176. Springer, 2016.

[2] Angelos Mallios, Pere Ridao, David Ribas, and Emili Hernández. Scan matching slam in underwater environments. *Autonomous Robots*, 36(3):181–198, 2014.

[3] Jie Shao, Wuming Zhang, Nicolas Mellado, Nan Wang, Shuangna Jin, Shangshu Cai, Lei Luo, Thibault Lejemble, and Guangjian Yan. Slam-aided forest plot mapping combining terrestrial and mobile laser scanning. *ISPRS Journal of Photogrammetry and Remote Sensing*, 163:214–230, 2020.

[4] Sajad Saeedi, Michael Trentini, Mae Seto, and Howard Li. Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics*, 33(1):3–46, 2016.

[5] Pierre-Yves Lajoie, Benjamin Ramtoula, Fang Wu, and Giovanni Beltrame. Towards collaborative simultaneous localization and mapping: a survey of the current research landscape. *arXiv preprint arXiv:2108.08325*, 2021.

[6] Nathan Michael, Shaojie Shen, Kartik Mohta, Vijay Kumar, Keiji Nagatani, Yoshito Okada, Seiga Kiribayashi, Kazuki Otake, Kazuya Yoshida, Kazunori Ohno, et al. Collaborative mapping of an earthquake damaged building via ground and aerial robots. In *Field and service robotics*, pages 33–47. Springer, 2014.

[7] Liam Paull, Guoquan Huang, Mae Seto, and John J Leonard. Communication-constrained multi-auv cooperative slam. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 509–516. IEEE, 2015.

[8] Michael T Ohradzansky, Eugene R Rush, Danny G Riley, Andrew B Mills, Shakeeb Ahmad, Steve McGuire, Harel Biggie, Kyle Harlow, Michael J Miles, Eric W Frew, et al. Multi-agent autonomy: Advancements and challenges in subterranean exploration. *arXiv preprint arXiv:2110.04390*, 2021.

[9] Yulun Tian, Katherine Liu, Kyel Ok, Loc Tran, Danette Allen, Nicholas Roy, and Jonathan P How. Search and rescue under the forest canopy using multiple uavs. *The International Journal of Robotics Research*, 39(10-11):1201–1221, 2020.

[10] Jorge Pena Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. *IEEE Access*, 8:191617–191643, 2020.

[11] Kurt Konolige, Dieter Fox, Benson Limketkai, Jonathan Ko, and Benjamin Stewart. Map merging for distributed robot navigation. In *Proceedings 2003 IEEE/RSJ international conference on intelligent robots and systems (IROS 2003)(Cat. No. 03CH37453)*, volume 1, pages 212–217. IEEE, 2003.

[12] Luca Carlone, Miguel Kaouk Ng, Jingjing Du, Basilio Bona, and Marina Indri. Simultaneous localization and mapping using rao-blackwellized particle filters in multi robot systems. *Journal of Intelligent & Robotic Systems*, 63(2):283–307, 2011.

[13] Andrew Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12):1243–1256, 2006.

[14] Jose-Luis Blanco, Javier Gonzalez, and Juan-Antonio Fernandez-Madrigal. A new method for robust and efficient occupancy grid-map matching. In *Iberian*

Conference on Pattern Recognition and Image Analysis, pages 194–201. Springer, 2007.

[15] Stefano Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous robots*, 25(3):305–316, 2008.

[16] Liang Ma, Jihua Zhu, Li Zhu, Shaoyi Du, and Jingru Cui. Merging grid maps of different resolutions by scaling registration. *Robotica*, 34(11):2516–2531, 2016.

[17] Yufeng Yue and Danwei Wang. *Collaborative Perception, Localization and Mapping for Autonomous Systems*, volume 2. Springer Nature, 2020.

[18] Xun S Zhou and Stergios I Roumeliotis. Multi-robot slam with unknown initial correspondence: The robot rendezvous case. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pages 1785–1792. IEEE, 2006.

[19] Carlos Alberto Velásquez Hernández and Flavio Augusto Prieto Ortiz. A real-time map merging strategy for robust collaborative reconstruction of unknown environments. *Expert Systems with Applications*, 145:113109, 2020.

[20] Sajad Saeedi, Liam Paull, Michael Trentini, Mae Seto, and Howard Li. Map merging for multiple robots using hough peak matching. *Robotics and Autonomous Systems*, 62(10):1408–1424, 2014.

[21] Sajad Saeedi, Liam Paull, Michael Trentini, and Howard Li. Occupancy grid map merging for multiple robot simultaneous localization and mapping. *International Journal of Robotics and Automation*, 30(2):149–157, 2015.

[22] Jiří Hörner. Map-merging for multi-robot system. 2016.

[23] Elizabeth R Boroson and Nora Ayanian. 3d keypoint repeatability for heterogeneous multi-robot slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6337–6343. IEEE, 2019.

[24] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[25] Daisuke Kakuma, Satoki Tsuichihara, Gustavo Alfonso Garcia Ricardez, Jun Takamatsu, and Tsukasa Ogasawara. Alignment of occupancy grid and floor maps using graph matching. In *2017 IEEE 11th international conference on semantic computing (ICSC)*, pages 57–60. IEEE, 2017.

[26] Christina Georgiou, Sean Anderson, and Tony Dodd. Constructing informative bayesian map priors: A multi-objective optimisation approach applied to indoor occupancy grid mapping. *The International Journal of Robotics Research*, 36(3):274–291, 2017.

[27] Saeed Gholami Shahbandi, Martin Magnusson, and Karl Iagnemma. Nonlinear optimization of multimodal two-dimensional map alignment with application to prior knowledge transfer. *IEEE Robotics and Automation Letters*, 3(3):2040–2047, 2018.

[28] Rema Daher, Theodor Chakhachiro, and Daniel Asmar. A novel method for map alignment assessment using synthetic displacement fields. In *2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 148–155. IEEE, 2021.

[29] Victor Terra Ferrão, Cássio Dener Noronha Vinhal, and Gelson da Cruz. An occupancy grid map merging algorithm invariant to scale, rotation and translation. In *2017 Brazilian Conference on Intelligent Systems (BRACIS)*, pages 246–251. IEEE, 2017.

[30] Jose-Luis Blanco, Javier González-Jiménez, and Juan-Antonio Fernández-Madrigal. A robust, multi-hypothesis approach to matching occupancy grid maps. *Robotica*, 31(5):687–701, 2013.

[31] Qcar by quanser. https://www.quanser.com/products/qcar/.

[32] P Cheeseman, R Smith, and M Self. A stochastic map for uncertain spatial relationships. In *4th international symposium on robotic research*, pages 467–474. MIT Press Cambridge, 1987.

[33] Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *The international journal of robotics research*, 23(7-8):693–716, 2004.

[34] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. *Aaai/iaai*, 593598, 2002.

[35] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI*, volume 3, pages 1151–1156, 2003.

[36] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46, 2007.

[37] Giorgio Grisetti, Rainer Kümmerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[38] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278. IEEE, 2016.

[39] Mathieu Labbé and François 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*, 36(2):416–446, 2019.

[40] Zutao Jiang, Jihua Zhu, Congcong Jin, Siyu Xu, Yiqiong Zhou, and Shanmin Pang. Simultaneously merging multi-robot grid maps at different resolutions. *Multimedia Tools and Applications*, 79(21):14553–14572, 2020.

[41] Andrea Censi, Luca Iocchi, and Giorgio Grisetti. Scan matching in the hough domain. In *Proceedings of the 2005 IEEE international conference on robotics and automation*, pages 2739–2744. IEEE, 2005.

[42] Johann Radon. On the determination of functions from their integral values along certain manifolds. *IEEE transactions on medical imaging*, 5(4):170–176, 1986.

[43] Heon-Cheol Lee and Beom-Hee Lee. Enhanced-spectrum-based map merging for multi-robot systems. *Advanced Robotics*, 27(16):1285–1300, 2013.

[44] Sajad Saeedi, Liam Paull, Michael Trentini, Mae Seto, and Howard Li. Efficient map merging using a probabilistic generalized voronoi diagram. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 4419–4424. IEEE, 2012.

[45] Zhiyang Lin, Jihua Zhu, Zutao Jiang, Yujie Li, Yaochen Li, and Zhongyu Li. Merging grid maps in diverse resolutions by the context-based descriptor. *ACM Transactions on Internet Technology (TOIT)*, 21(4):1–21, 2021.

[46] Heoncheol Lee and Seunghwan Lee. Extended spectra-based grid map merging with unilateral observations for multi-robot slam. *IEEE Access*, 2021.

[47] David Forsyth and Jean Ponce. *Computer vision: A modern approach.* Prentice hall, 2011.

[48] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[49] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

[50] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[51] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[52] Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[53] Chris Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[54] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[55] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *European conference on computer vision*, pages 214–227. Springer, 2012.

[56] Emmanouil G Tsardoulias, Marios Protopapas, Andreas L Symeonidis, and Loukas Petrou. A comparative analysis of pattern matching techniques towards ogm evaluation. *Journal of Intelligent & Robotic Systems*, 98(3):733–758, 2020.

[57] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[58] Philip HS Torr and Andrew Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer vision and image understanding*, 78(1):138–156, 2000.

[59] Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*, pages 1–10. IEEE, 2018.

[60] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. Ieee, 2011.

[61] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[62] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE, 2000.

[63] Miquel Kegeleirs, Giorgio Grisetti, and Mauro Birattari. Swarm slam: Challenges and perspectives. *Frontiers in Robotics and AI*, 8:23, 2021.

[64] Ashutosh Singandhupe and Hung Manh La. A review of slam techniques and security in autonomous driving. In *2019 third IEEE international conference on robotic computing (IRC)*, pages 602–607. IEEE, 2019.

Vita Auctoris

NAME:	Sooraj Sunil
PLACE OF BIRTH:	Kerala, India
YEAR OF BIRTH:	1996
EDUCATION:	Rajalakshmi Engineering College, Tamil Nadu, India 2014 - 2018 Bachelor of Engineering. University of Windsor, Windsor, Ontario 2019 - 2021 Master of Applied Science.