Titre: Title:	Machine Learning for Multi-Robot Semantic Simultaneous Localization and Mapping
Auteur: Author:	Benjamin Ramtoula
Date:	2020
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Ramtoula, B. (2020). Machine Learning for Multi-Robot Semantic Simultaneous Localization and Mapping [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/5205/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Machine Learning for Multi-Robot Semantic Simultaneous Localization and Mapping

BENJAMIN RAMTOULA
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie Aérospatial

AVRIL 2020

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Machine Learning for Multi-Robot Semantic Simultaneous Localization and Mapping

présenté par **Benjamin RAMTOULA**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Foutse KHOMH, président
Giovanni BELTRAME, membre et directeur de recherche
Guchuan ZHU, membre et codirecteur de recherche
Guillaume-Alexandre BILODEAU, membre

ACKNOWLEDGEMENTS

I would like to thank all the great people that have made the work presented in this thesis possible. First and foremost, I am very grateful to my supervisor, Prof. Giovanni Beltrame, for giving me the freedom and the means to explore my ideas. He provided me with numerous great opportunities to acquire experience and learn, and constant support. The environment he created in MISTLab has greatly contributed to my motivation and well-being during my time in Montreal.

I would also like to thank all the members of MISTLab that I have interacted with, and with whom I enjoyed entertaining lunch and coffee breaks, lab meetings, journal clubs, and interesting discussions.

I am especially thankful to my close collaborators that I had a great deal of fun working with, and learned a lot from: Adam, Pierre-Yves and Ricardo.

I had the great chance and pleasure to spend 4 months in the Multi-robot Systems Lab (MSL) of Stanford University and had an incredible experience thanks to the friendliness and warm welcome of all of MSL's members. For this, I would like to thank Prof. Mac Schwager for hosting me, and for his precious guidance on the project that resulted from this stay, as well as all MSL members.

Finally, I would like to thank my family, and especially my partner, Marion, who have all been incredibly supportive at all times.

RÉSUMÉ

L'automatisation et la robotique prennent une place de plus en plus importante dans notre vie quotidienne, avec de nombreuses utilisations possibles. Les robots pourraient nous épargner des tâches dangereuses et pénibles, ou rendre des choses impossibles jusqu'à maintenant possibles. Pour que les robots s'intègrent en toute sécurité dans notre monde et dans de nouveaux environnements inconnus, il est clef qu'ils soient équipés d'une capacité de perception, et en particulier qu'ils puissent se localiser par rapport à leur entourage. Afin d'être réellement indépendants, les robots doivent pouvoir le faire en se basant uniquement sur leurs propres capteurs, les plus couramment utilisés étant les caméras. Une solution pour obtenir de telles estimations est d'utiliser un algorithme de cartographie et localisation simultanée (SLAM), dans lequel le robot va simultanément construire une carte de son environnement et estimer son propre état. Le SLAM avec un seul robot a fait l'objet de nombreux travaux scientifiques, et est désormais considéré comme un domaine de recherche mature. Cependant, l'utilisation d'une équipe de robots peut offrir plusieurs avantages en termes de robustesse, d'efficacité et de performances pour de nombreuses tâches. Dans ce cas, des algorithmes de SLAM multi-robots sont nécessaires pour permettre à chaque robot de bénéficier de l'expérience de toute l'équipe. Le SLAM multi-robot peut s'appuyer sur des solutions SLAM classiques, mais nécessite des adaptations et fait face à des contraintes de calculs et de communications supplémentaires. Un défi particulier dans le SLAM multi-robots est la nécessité pour les robots de trouver des fermetures de boucles inter-robots: des liens entre les trajectoires de différents robots qui peuvent être trouvés lorsqu'ils visitent le même endroit. Deux catégories d'approches sont possibles pour détecter les fermetures de boucles inter-robots. Dans les méthodes indirectes, les robots communiquent pour vérifier s'ils ont cartographié un espace commun, puis tentent de trouver des fermetures de boucles à partir des données recueillies par chacun des robots dans cet espace. Dans les méthodes directes, les robots s'appuient directement sur les données de leurs capteurs pour estimer les fermetures de boucles. Chaque approche a des avantages et des inconvénients, mais les méthodes indirectes ont été plus étudiées récemment. Ce mémoire s'appuie sur les avancées récentes de la vision par ordinateur pour présenter des contributions à chaque catégorie d'approches pour la détection de fermetures de boucles inter-robots. Une première contribution est présentée pour la détection de fermetures de boucles indirecte dans une équipe de robots entièrement en communication. Elle utilise des constellations, une représentation sémantique compacte de l'environnement basée sur les objets qui le compose. Des descripteurs ainsi que des méthodes de comparaisons des constellations sont présentés pour reconnaître de manière robuste les

lieux en fonction de leur constellation avec peu de d'échanges de données. Ces derniers sont utilisés dans un mécanisme de reconnaissance de lieu décentralisé qui s'adapte efficacement à une taille d'équipe grandissante. La méthode proposée rivalise avec l'état de l'art en termes de performances et d'échanges de données, tout en étant plus transparente et interprétable. La deuxième contribution présentée dans ce mémoire est une méthode d'estimation de position et d'orientation relative basée sur des images qui peut être appliquée à de nombreux objets, y compris des robots pour la détection directe de fermetures de boucles inter-robots. La solution est composée d'un "*front-end*" reposant sur l'apprentissage profond pour détecter et suivre l'entité dont la position et orientation doivent être estimées, et d'un "*back-end*" classique de filtrage pour obtenir les estimations finales. Cette approche est testée à bord d'un drone estimant la position et l'orientation relative d'un autre drone, et surpasse l'état de l'art en termes de performances et de vitesse. Les deux solutions présentées sont complémentaires, et facilement adaptables à de nouveaux scénarios de déploiements de robots.

ABSTRACT

Automation and robotics are becoming more and more common in our daily lives, with many possible applications. Deploying robots in the world can extend what humans are capable of doing, and can save us from dangerous and strenuous tasks. For robots to be safely sent out in our real world, and in new unknown environments, one key capability they need is to perceive their environment, and particularly to localize themselves with respect to their surroundings. To truly be able to be deployed anywhere, robots should be able to do so relying only on their sensors, the most commonly used being cameras. One way to generate such an estimate is by using a simultaneous localization and mapping (SLAM) algorithm, in which the robot will concurrently build a map of its environment and estimate its state within it. Single-robot SLAM has been extensively researched and is now considered a mature field. However, using a team of robots can provide several benefits in terms of robustness, efficiency, and performance for many tasks. In this case, multi-robot SLAM algorithms are required to allow each robot to benefit from the whole team's experience. Multi-robot SLAM can build on top of single-robot SLAM solutions, but requires adaptations and faces computation and communication constraints. One particular challenge that arises in multi-robot SLAM is the need for robots to find inter-robot loop closures: relationships between trajectories of different robots that can be found when they visit the same place. Two categories of approaches are possible to detect inter-robot loop closures. In indirect methods, robots communicate to find if they have mapped the same area, and then attempt to find loop closures using data gathered by each robot in the place that was jointly visited. In direct methods, robots directly rely on data they gather from their sensors to estimate the loop closures. Each approach has its own benefits and challenges, with indirect methods being more popular in recent works. This thesis builds on recent computer vision advancements to present contributions to each category of approaches for inter-robot loop closure detection. A first approach is presented for indirect loop closure detection in a team of fully connected robots. It relies on constellations, a compact semantic representation of the environment based on objects that are in it. Descriptors and comparison methods for constellations are designed to robustly recognize places based on their constellation with minimal data exchange. These are used in a decentralized place recognition mechanism that is scalable as the size of the team increases. The proposed method performs comparably to state-of-the-art solutions in terms of performance and data exchanges require, while being more meaningful and interpretable. The second approach presented in this thesis is a vision-based relative pose estimation method that can be applied to many objects, including robots for direct

inter-robot loop closure detection. It relies on a deep learning-based front-end to detect and track the rigid body whose pose is to be estimated, and on a classical filter-based back-end to estimate its full relative pose. This method is tested onboard a flying drone estimating the relative pose of another flying drone, and outperforms state-of-the-art solutions in terms of performance and speed. The two contributed solutions are complementary and easily adaptable to new robotics scenarios.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
AUC	Area Under Curve
BRIEF	Binary Robust Independent Elementary Features
BoW	Bag-of-Words
CAPRICORN	Communication Aware Place Recognition using Interpretable Constellations of Objects in Robot Networks
CNN	Convolutional Neural Networks
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
GPS	Global Positioning System
GPU	Graphics Processing Unit
HOG	Histogram of Oriented Gradients
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
IOU	Intersection Over Union
MSL-RAPTOR	Monocular Sequential Lightweight Rotation And Position Tracking On-Robots
PCA	Principal Component Analysis
PF	Particle Filter
PnP	Perspective-n-Point
QR	Querying Robot
RANSAC	Random Sample Consensus
SE	Special Euclidean group
SIFT	Scale-Invariant Feature Transforms
SLAM	Simultaneous Localization and Mapping
SO	Special Orthogonal group
SURF	Speeded-Up Robust Features
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter
VLAD	Vector of Locally Aggregated Descriptors

CHAPTER 1 INTRODUCTION

Due to the Fourth Industrial Revolution [1], robots and automation are entering our daily lives at a fast pace [2]. In fact, a recent report by the World Economic Forum [3] predicts that more than half of today's tasks will be performed by machines by 2025. That's only 5 years from now! Some of these technologies are already visible with self-driving cars [4] or agricultural drones [5] for example.

One of the key aspects that has made this available is the advancements in perception technologies, allowing robots to make sense of their environment, and their place within it.

This thesis presents works creating tools to assist with perception in multi-robot systems. These contributions result from work within the Making Innovative Space Technologies Laboratory (MISTLab) of Polytechnique Montreal between January 2018 and March 2020 and in part from a collaboration with the Multi-Robot Systems Lab (MSL) of Stanford University, USA, between September 2019 and March 2020.

It is presented in the format of a thesis by articles, with the core content being composed of two scientific articles in Chapters 4 and 5.

The first article is in-press after being peer-reviewed and accepted for presentation at a conference:

- Benjamin Ramtoula, Ricardo de Azambuja and Giovanni Beltrame, "CAPRICORN: Communication Aware Place Recognition using Interpretable Constellations of Objects in Robot Networks," *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. [in-press]

This work was supervised by Prof. Beltrame, and the whole project benefited from Ricardo de Azambuja's high-level guidance.

The second article is currently under review for publication in a scientific journal:

- Benjamin Ramtoula, Adam Caccavale, Giovanni Beltrame, Mac Schwager, "MSL-RAPTOR: A Fast Onboard Monocular Multi-Object Pose Tracker," *IEEE Robotics and Automation Letters (RA-L)*, 2020.[under review]

This work was co-supervised by Prof. Beltrame and Prof. Schwager. Adam Caccavale was responsible for the back-end of the solution and had a key role in every step of the project.

1.1 Context and Basic Concepts

For the deployment of robots and automation in our lives and a wide variety of environment, equipping them with efficient perception capabilities is critical.

Useful tasks that robots can perform usually consists of interactions with their environment and require several steps. First, the robot will have to consider its current state as well as the state of its environment. Using those, it must make decisions on which actions to perform to carry out the task successfully. Finally, it must plan how to complete these actions and execute them. A robot's perception module is responsible for the first step, making sense of the environment and the robot's place within it. It is critical as it serves as the basis for every subsequent decision and action.

One particular task of the perception module is to provide a localization estimate of the robot. Some applications rely on external sources of information such as a Global Positioning System (GPS) to provide this information, but these are not always available or accurate enough. Hence, a perception system should be able to provide localization information on its own, solely relying on sensory input.

Due to the wide availability of cameras and their practicality for embedded systems, a common sensory input for perception is vision. In those cases, perception modules rely on the field of Computer Vision and process images of the environment to extract high-level information. This thesis focuses on vision-based perception.

1.1.1 Simultaneous Localization and Mapping

One popular group of perception algorithms are Simultaneous Localization and Mapping (SLAM) algorithms. They offer solutions to concurrently build maps of the environment and localize a robot within it. These algorithms have been widely developed by the robotics research community, and now offer solutions for accurate localization using visual sensors for a single robot [6].

Visual SLAM algorithms are typically separated into two parts: a front-end, and a back-end. The front-end is responsible for processing images into representations of the environment, estimates of motion, and relationships between parts of the stored map and a currently seen place. The back-end takes into account all estimates provided by the front-end, which are noisy and not necessarily consistent with each other, and generates the maximum likelihood estimate of the map and the robot's state.

Pose graph formulation

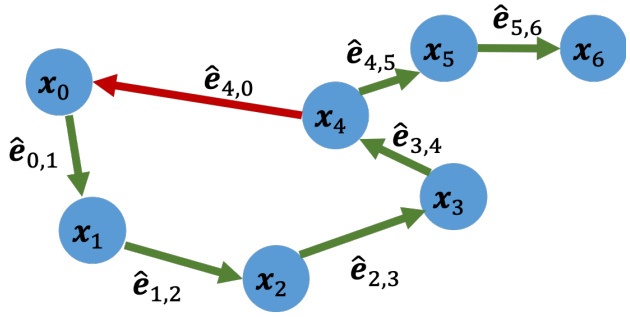

Figure 1.1: Example of a pose graph formulation of a SLAM problem. Nodes represent poses of the robot, and edges represent relative pose (transformation) estimates. Edges in green represent relations created by the odometry module and the arrow in red represents a relation created by a loop closure.

The state of a robot in the environment is usually represented by its general pose in the world, which contains information about its position and its orientation. In a space of dimension d, the pose at time i is defined as $\mathbf{x}_i \doteq [\mathbf{R}_i, \mathbf{t}_i]$, where $\mathbf{R}_i \in \mathrm{SO}(d)$ is a rotation matrix and $\mathbf{t}_i \in \mathbb{R}^d$ is a translation vector. The actual poses of a robot in an unknown environment are unknown and must be estimated by the SLAM solution. However, only noisy estimates of relationships between poses are estimated by SLAM algorithms. The estimated transformation between poses $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by a transformation matrix $\hat{\mathbf{e}}_{i,j} \doteq [\hat{\mathbf{R}}_{i,j}, \hat{\mathbf{t}}_{i,j}]$, where $\hat{\mathbf{R}}_{i,j} \in \mathrm{SO}(d)$ and $\hat{\mathbf{t}}_{i,j} \in \mathbb{R}^d$. Modern SLAM solutions typically formulate the SLAM problem as a graph in which poses $\mathbf{x}_i$ of the robot correspond to nodes, and edges $\hat{\mathbf{e}}_{i,j}$ represent estimates of constraints relating the poses $\mathbf{x}_i$ and $\mathbf{x}_j$, as illustrated in Fig. 1.1.

Front-end

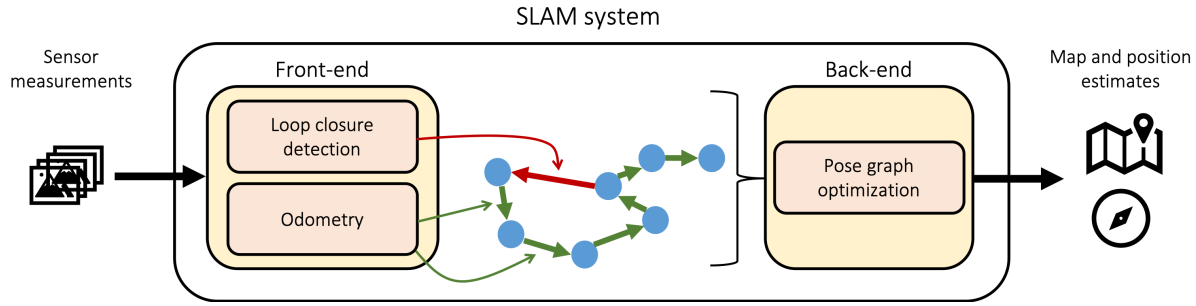

Figure 1.2: Architecture of a SLAM system.

The front-end itself is composed of two modules: the odometry module and the loop closure detection module, as illustrated in Fig. 1.2. The odometry module takes consecutive frames

to estimate the motion of the robot. The loop closure detection module detects when a robot revisits a place and helps reduce errors due to the accumulation of drift. In the graph formulation of SLAM, the front-end generates the nodes in the pose graph as well as the edges connecting them.

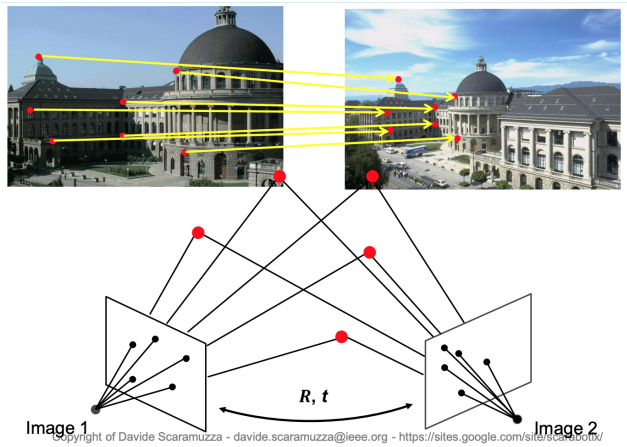

Figure 1.3: Working principle of visual odometry ©Davide Scaramuzza.

Odometry In creating the graph, visual odometry provides edges between consecutive poses $\mathbf{e}_{i,i+1}$. These are shown in green in Fig. 1.1. The working principle of visual odometry is to extract visual features from consecutive frames, match them, and use computer vision algorithms to estimate the transformation explaining best the change in feature positions between the frames (Fig. 1.3). A final optimization step of bundle adjustment can be used to improve odometry considering several consecutive estimates. Used alone, odometry will tend to drift and is unable to generate accurate global maps.

Loop closure detection To correct for drift accumulated using odometry, it is required to find loop closures in the pose graph. These are relations between poses generated far from each other in time. They appear when a robot revisits the same area later during its task, as illustrated in red in the simple example of Fig. 1.1. As a robot performs its task, its map and number of poses will grow, making loop closure detection a hard problem that must scale with a large number of poses. In visual SLAM, this is done by separating the process into two parts: visual place recognition, and relative pose estimation.

Visual place recognition finds places that have been revisited during the whole duration of the task. Since the re-encounter of a place may happen with a different point-of-view and at a different time, this module must be robust to viewpoint and condition changes. Moreover, it must have a mechanism to search efficiently for matches in previously seen places, as

comparing directly every place seen would require exponential computation as the robot gathers more data.

Once a place has been recognized, the transformation between poses from which the place has been seen must be estimated by the relative pose estimation module. This can be done using a similar working principle as the visual odometry (Sec. 1.1.1).

Back-end

Using the noisy relations created by the front-end, the back-end produces the final best estimate of all poses (Fig. 1.2). In the pose graph formulation, this is done by pose graph optimization. Pose graph optimization makes assumptions about the distribution of noise of the relative pose estimates $\hat{\mathbf{e}}_{i,j}$ to formulate a minimization problem:

$$\min_{\mathbf{x}} F(\hat{\mathbf{e}},\mathbf{x})$$
$$\text{where}$$
$$\mathbf{x} \doteq \{\mathbf{x}_i \forall i\}$$
$$\hat{\mathbf{e}} \doteq \{\hat{\mathbf{e}}_{i,j} \forall i,j\}$$

The solution to this problem results in the maximum likelihood estimate of all poses, and is found using classical optimization techniques.

1.1.2 Multi-robot teams

While research in robotics, including in SLAM, has initially focused on single robots, having a team of robots can be a better solution for some tasks. As such, multi-robot systems have been the topic of more and more interest recently. In terms of productivity, a team of robots can usually perform a task faster and more efficiently than a single robot. For example, in a mapping task, robots can split the areas to explore and combine their data to create a global map faster. Multi-robot systems can also offer more capabilities, for example, a team can be heterogeneous with each robot having specialties in sensors and locomotion. Robots can then collaborate to perform a task that would be impossible for a single robot.

Multi-robot systems can be centralized or decentralized. In a centralized system, all the robots rely on one entity to perform computations for a task and coordinate the team. On the other hand, decentralized systems rely on each robot of the team being able to make decisions and contribute to the computations for a task. These can offer more robustness to single-agent failure, and better scalability as the team grows.

1.2 Problem Statement

1.2.1 Challenges with multi-robot SLAM

SLAM has been well researched in the single-robot case, but in a team, robots must incorporate data gathered by all members of the team to generate a consistent global map and localize themselves within it. This makes multi-robot SLAM solutions subject to more requirements than single-robot SLAM.

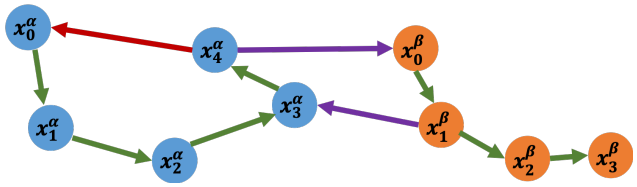

Figure 1.4: Example of a multi-robot pose graph formulation of a SLAM problem. Nodes represent poses of the robots α in blue and β in orange, and edges represent relative pose estimates. Edges in green are generated from odometry, the arrow in red represents an intra-robot loop closure, and arrows in purple represent inter-robot loop closures.

One of the critical requirements for multi-robot SLAM is to align the maps of each robot, which are usually built in each robot's individual reference frame. This is done by relying on inter-robot loop closures: estimates of transformations connecting the poses of different robots. These are very similar to loop closures found in single robot SLAM, referred to as intra-robot loop closures, except that they correspond to poses of different robots that have visited the same place. An illustration is shown in Fig. 1.4.

Inter-robot loop closures can be estimated directly or indirectly. In a direct manner, a robot can obtain relative pose estimates to other robots without needing to communicate. This can be the case when specific sensors provide such information, or when a robot is in the field of view of another, which allows estimating its relative pose visually. In an indirect manner, robots communicate and share data to find inter-robot loop closures.

Detecting inter-robot loop closures indirectly is made very hard by constraints associated with multi-robot systems. The same problems of intra-robot loop closure detection are still present, with the added difficulty that the search must now be performed over a full team of robots, and that data exchanges are limited by communication constraints. It is thus necessary to develop solutions that can find inter-robot loop closures with efficient computation and low bandwidth requirements, especially as the number of robots scales up.

1.2.2 Advances in computer vision

In the past few years, the use of deep learning has allowed great improvements for many computer vision tasks. One particular advantage brought by deep learning has been the ability to learn high-level representations and features that can serve as a base for multiple tasks. These have been at the center of methods extracting high-level understanding from images in tasks such as classification, object detection, or image segmentation.

Hence, computer vision can provide robust high-level features from images as well as semantic information while reaching great performance on many challenging tasks.

1.2.3 Potential for improving the multi-robot SLAM front-end

High-level features and semantics can be very descriptive while remaining compact and providing robustness to changes in conditions such as illumination. As such, they have an interesting potential to be used for efficient inter-robot loop closure detection methods. The problem studied in this thesis is how these recent advances in computer vision can improve inter-robot loop closures while considering all the constraints existing in a multi-robot system.

In particular, how recent computer vision advances can serve for both direct and indirect inter-robot loop closure detection.

This is done through two approaches we present:

- CAPRICORN, a mechanism for decentralized place recognition in a robot team that relies on a compact semantic representation of the environment.
- MSL-RAPTOR, a framework for relative pose estimation that can serve for direct inter-robot loop closure detection.

During the time spent at MISTLab, in addition to the two presented works, we also contributed to DOOR-SLAM [7] (not presented in this thesis). DOOR-SLAM is a fully distributed SLAM system that is robust to front-end outliers. Both CAPRICORN and MSL-RAPTOR are potential additions to DOOR-SLAM for improved place recognition and estimation of inter-robot loop closures. The robust back-end mechanism of DOOR-SLAM would provide robustness to the weaknesses of CAPRICORN and MSL-RAPTOR.

1.3 Research Objectives

Through the articles presented in Chapters 4 and 5, this thesis pursues the following research objectives:

- Identify from which advances in computer vision the inter-robot loop closure detection problem can benefit.
- Develop an indirect loop closure detection solution requiring low bandwidth and scalable to a large robotic team.
- Develop a direct loop closure detection solution that can easily be adapted to a wide range of robots.
- Evaluate the developed solutions using common datasets, baselines, and against state-of-the-art results.

1.4 Thesis Outline

The rest of this thesis is organized in 6 chapters. The following one, Chapter 2 contains a literature review of relevant works related to the problems considered here. It is followed by Chapter 3 which details the approach taken in this work, as well as how the articles presented fit within the problem tackled. Chapter 4 presents an article proposing a method for indirect loop closure detection. Chapter 5 contains an article proposing a relative pose estimation method which can serve for direct inter-robot loop closure detections. Chapter 6 discusses the results and approaches presented in both articles. Finally, Chapter 7 summarizes the thesis and provides an overview of limitations and ideas for future works.

The references can be found at the end of this thesis.

CHAPTER 2 LITERATURE REVIEW

The chapter will present past works that have focused on the loop closure detection problem. First, works relying on indirect methods and visual place recognition are presented. These have considered mainly the single robot case, and usually do not consider communication constraints and team-wide search mechanisms. Recently, some multi-robot visual place recognition solutions have been proposed, which are also presented. The second part of this chapter contains an overview of works on object relative pose estimation. This is a more general problem that can be applied to direct inter-robot loop closure detection when estimating the relative pose of another robot. These rely on line-of-sight observations of other robots in the team.

2.1 Indirect inter-robot loop closure detection

2.1.1 Traditional approach

Indirect loop closure detections usually rely on two steps. The first one is visual place recognition and is responsible for detecting if a place has been already seen, either previously in the single-robot case, or by another robot in the multi-robot case. The second step is to estimate the relative motion between poses from which the place was seen.

Visual place recognition relies on a description of places generated from the gathered images. These descriptions can either be local or global.

Local features and descriptors Local feature descriptors extract salient parts of an image and generate descriptions of it. These are found in two steps: first, *detection* finds keypoints, points in the image that are exhibit interesting repeatability properties, and then *description* generates a descriptor associated with those keypoints. Some approaches are able to do both steps such as scale-invariant feature transforms (SIFT) [8], speeded-up robust features (SURF) [9]. Others focus on one of these, such as features from accelerated segment test (FAST) [10] which detects features, or binary robust independent elementary features (BRIEF) [11] which provides a keypoint descriptor.

For place recognition, the descriptors of these features can be compared using a distance measure, allowing to detect if multiple similar features appear on two images. However, each image can contain hundreds of visual features, each with its own descriptor.

The bag-of-words (BoW) model [12, 13] is a solution to quantize local keypoints into a single

global description of an image. It does so by partitioning the feature descriptor space into a set number of words, usually by using clustering on a training set. The BoW descriptor of an image is then generated by counting the number of features of the image associated with each partition of the space (word). In doing so, it loses information about features' locations, but provides a compact descriptor of an image, and has shown to be efficient and accurate for place recognition.

Not only can local features and descriptors serve for place recognition using a BoW approach, but they are usually required for the relative pose estimation between two images. This step is performed by matching specific keypoints in two images and using classical multi-view geometry approaches to estimate the relative pose [14].

Global descriptors Global descriptors for place recognition do not rely on a detection step, and directly make use of a single descriptor taking into account all the content of an image.

Gist [15] is such a descriptor, and is obtained by applying a Gabor filter with different scales and orientation parameters over an image, and averaging the resulting feature maps into a single descriptor.

Histograms of oriented gradients (HOG) [16] are another type of global descriptor. They are obtained by first computing the gradients on a pre-processed input image by applying a filter. The image is then split in a grid of patches, and a histogram of the gradients orientations is computed over each patch. After a normalization step, the histograms are combined to make a global descriptor.

The BoW presented in the previous paragraph relies on local features but is made a global descriptor through the quantization.

These can serve to match places but do not contain enough information to estimate relative poses from matched images. This step still requires the use of keypoints and their descriptors.

Learned descriptors A key requirement for a place recognition method is its robustness to changing environment, conditions, and viewpoints. In traditional computer vision approaches, filters and extraction methods are hand-crafted to meet these requirements, but it is hard to ensure they generalize well to many scenarios.

Recently, advances in deep learning have allowed discovering robust representations to support computer vision tasks. In particular, Convolutional Neural Networks (CNNs) [17] can use backpropagation to learn a high number of parameters of convolutional layers to perform

tasks such as image classification. These are trained using large and diverse datasets, such as ImageNet [18]. As a result, parameters learned in convolutional layers produce high-level feature extraction [19,20] which have served as a base to improve performance on many tasks compared to traditional approaches.

In place recognition, using convolutional layers from pre-trained neural networks on a different task, such as image classification or object detection, has allowed making use of robust features as descriptors to compare and recognize places, even in radically different conditions [21, 22].

More than making use of high-level learned features directly as descriptors, recent state-of-the-art place recognition systems use an adaptation of the Vector of Locally Aggregated Descriptors (VLAD) process into a trainable NetVLAD layer, which is applied on the features extracted from a CNN. This allows end-to-end training to learn the best parameters for the NetVLAD layer, and generate a robust global image descriptor tailored to the place recognition task [23].

2.1.2 Multi-robot decentralized adaptations

In a centralized context, the central entity can usually access data gathered by each robot and relies on similar methods as single-robot systems. In a decentralized multi-robot system context, similar descriptors can be used for place recognition, but require advanced mechanisms to perform team-wide searches efficiently.

Tardioli *et al.* avoid communicating full descriptors of keypoints between robots by relying on a pre-defined descriptor vocabulary and sending only indexes in the vocabulary [24]. This allows matching keypoints without between different robots by sending only their position and associated descriptor index.

Cieslewski *et al.* instead consider the case where the team of robots is fully connected, and all robots can communicate with one another. They first propose a mechanism relying on a BoW descriptor, in which each robot is given responsibility for specific words of the vocabulary in such a way that a place recognition query will split computation and communication over the whole team [25]. A second approach relies on the NetVLAD global descriptor and assigns a partition of the descriptor space to each robot in the team. This assignment allows a generated descriptor to be sent to only the robot responsible for the partition that contains it, and to find matches over robots of the whole team that have also found descriptors in that partition [26].

Tian *et al.* [27, 28] and Giamou *et al.* [29] proposed complementary frameworks and algo-

rithms to plan and coordinate data exchanges according to available communication and computation resources.

2.2 Direct inter-robot loop closures

Instead of sharing data between robots, which can be impractical in a multi-robot setting, loop closures can be detected when a robot appears in the line-of-sight of another using a direct method.

2.2.1 Using hardware adaptations

Most works that have used direct inter-robot loop closure in a multi-robot SLAM setting have relied on specific hardware other than the vision module that is usually available. They can use specific sensors to provide the relative pose [30, 31] such as time-of-flight distance measurements [32].

However, additional hardware can affect the cost, payload, and energy consumption of a robot. Instead, vision is usually available and used for other tasks, and it would be preferable to use it for estimating the relative pose of another robot visible in its field-of-view. This is done by a previous work [33] in which a robot is equipped with a visual cue to facilitate relative pose estimation from the other robot. This still required to modify the physical construction of a robot to equip it with an easily recognizable pattern.

2.2.2 Using vision

Yet, the problem of relative pose estimation of a rigid body using vision finds many applications in robotics, such as manipulation, collision avoidance, and planning. When estimating the 3 degrees of translation as well as the 3 degrees of orientation of the entity, it is commonly referred to as 6D pose estimation and is framed as estimating the pose of an object in the frame of the camera [34].

This general problem can be applied to estimate the relative pose of another robot that is visible to provide a direct estimate of an inter-robot loop closure.

Historically, 6D pose estimation methods have relied on matching the current view of an object to pre-defined templates or sets of features.

Template-based approaches Template-based approaches rely on templates of the object of interest to estimate its pose based on its projection in the image. Templates consist of

samples of the object model as seen from different viewpoints. Comparing a template against the current view in the image can produce a distance measure that should be minimized to find the pose associated with the best matching template.

Several distance measures exist, one of the first being used is the Hausdorff distance. It measures how well two subsets of a metric space match each other by measuring the largest distance that appears between any two points of the sets. However, the mathematical definition of this distance applies to closed continuous sets, whereas templates available for matching in images are discrete sets of points. Early works [35] have developed several algorithms to rely on the Hausdorff distance for template matching evaluation. They first present a method to obtain a Hausdorff distance based on discrete points which can be applied for comparison between a template and an image. Another proposed algorithm deals with the problem that occlusions can occur and cause high distance measures even when portions of the patterns match. To handle these scenarios, a definition of the Hausdorff distance on portions of shapes is defined that allows to match objects that may be partially hidden. Finally, techniques are presented to avoid considering all poses of a model and their associated template when comparing to an image, and to efficiently search for the best match.

However, the Hausdorff distance does not take the direction of edges into account during comparisons, which can lead to false positive decisions when matching a template to an image [36]. To make use of these directions, Steger [37, 38] relies on representing the image and template as a set of points with associated gradient direction vectors. These added gradient direction vectors allow the direction of edges to contribute to the distance score. The similarity measure they propose consists of using the sum of dot products between the template and image gradients, which results in better performance for object pose estimation.

While taking into account the direction of edges, Steger's similarity measure can be heavily affected by cluttered background, which could modify independently gradients of the template and the current view, leading to a low similarity score. To face these problems, Hinterstoisser *et al.* [39] adapt Steger's similarity measure to use it with new model and image representations that consider all gradients in a local neighborhood instead of only the local gradient. Thanks to this representation, the effect of the edge is more consistent in the representation independently on the background, leading to a more robust similarity measure.

LINEMOD [40] considers the increased availability of RGB-D cameras and builds on the previous work to present a framework that incorporates the use of an additional depth modality to help against errors due to cluttered backgrounds. This is done by the addition of depth cues to the template representations which complement the computation of the similarity measure.

Feature-based approaches Similarly to databases of templates used in template-based approaches, the first feature-based methods relied on databases of keypoints and descriptors associated with viewpoints of an object's model that are matched to the current view of an object.

Lepetit *et al.* proposed to frame this as a classification task [41], in which each patch of an image around a keypoint must be classified as one pre-computed patch associated with a specific keypoint from the training set. Using the consistent matches found over several keypoints and patches allows estimating the pose. To prepare for classification using the training set, view sets of patches are created by using images of the object to which homographies are applied with a regular sampling of parameters such that the resulting warped images simulate a wide range of viewpoints. For each viewpoint, a clustering algorithm computes a set number of means over the patches in an eigen space found by a Principle Component Analysis (PCA). This allows having a computationally efficient method during inference, in which patches of the input image are converted into eigen space, and matched to known patches and viewpoints using a Nearest Neighbor search.

The same group extends this work by using Randomized Trees [42] instead of the Nearest Neighbor search used previously [43]. The overall approach and methods, such as for building the view-set, remain very similar, with a slight variation when building view sets. In the previous approach, keypoints are detected in original images and homographies are applied to the patches around them. In this work, the whole image is warped and new keypoints are detected on the resulting image instead. This leads to the use of more robust keypoints. With this solution, a tree is built in which nodes perform a test to split the space of image patches. Each leaf contains an estimate of class distributions for the input image patch based on training data. Similarly to before, the final matches over several keypoints allow estimating the pose with a P3P algorithm and a RANSAC scheme. This proposed approach is still very effective and allows for faster runtime than the previous one.

With the advances allowing to create 3D scans and model objects accurately, several works have made use of such models for pose estimation. Choi and Christensen presented a work combining approaches relying on keypoints and the use of CAD models. They rely on SURF keypoints matched from a pre-computed set of keyframes of the object to the new frame, and back-project the 3D coordinate of each point keypoint to the CAD model to find the pose from the 2D-3D correspondences matched. This keypoint-based approach serves as the "Global Pose Estimation". Once an initial pose estimate is obtained, the pose is tracked using the following frames with the faster "Local Pose Estimation" module, in which the 3D CAD module is used with a template-based approach. This work combines the strengths of both

feature-based methods for initialization and template-based methods for faster pose tracking.

Both template-based and feature-based approaches have been very performant, however, they are always instance-based. This means that for the object for which pose is to be estimated, they require specific knowledge of models or images from many viewpoints. They also rely heavily on textures and salient features, with descriptors that are required to be recognizable under many conditions.

Learning approaches Following computer vision advancements, deep learning has contributed to new pose estimation solutions with better robustness and performance. One approach has been to learn to regress poses directly from images.

SSD-6D [44] does so by taking inspiration in 2D single shot object detectors. From an image, it extracts six feature maps using pre-trained weights for another task. Each map is then convolved with trained kernels that are responsible for predicting objects' 2D bounding boxes, object classes, and scores for viewpoints and in-plane rotations that serve to determine the 6D pose. Viewpoints and in-plane rotations are treated as a classification that is learned based on a pre-defined fixed number of viewpoints and in-plane rotations, similarly to feature-based methods. From the 2D bounding box, the classified viewpoint and in-plane rotation, and knowledge of the object's size, the full 6D pose can directly be estimated. Due to the discrete sampling of viewpoints and in-plane rotations needed to treat them as a classification task, additional pose refinement steps are proposed depending on sensor modalities.

PoseCNN [45] also learns to regress the pose in an end-to-end training manner by decoupling the pose estimation in several components. It first obtains a semantic segmentation of the image. It then predicts the 2D coordinates of the object in the image based on pixels' semantic labels, and a learned pixel-wise orientation indicator. In parallel, the network directly predicts the depth of the object center, which can be combined with camera intrinsics and the predicted 2D projection of the object in the image to estimate translation. Finally, the orientation is estimated by a direct regression of convolutional features extracted from the bounding box which contained the object's projection.

LieNet [46] takes a similar approach to PoseCNN, in which a semantic segmentation of the object is found by the network, as well as a depth estimate. As in PoseCNN, projective geometry is used to infer the position component of the pose, and orientation is found by direct regression, except that it uses a different orientation representation. In fact, PoseCNN uses a quaternion representation which is over-parametrized given that only 3 parameters are necessary to encode an orientation. LieNets instead use Lie algebra to regress with a 3-dimensional vector.

Better accuracies have been reached by methods that, instead of attempting to directly regress the pose, use deep learning to extract and associate 2D features from images with 3D keypoints associated with the object. Once associations are found, a Perspective-n-Point (PnP) algorithm [14] is used to estimate the object's pose, with a random sample consensus (RANSAC) component to be robust to erroneous associations.

BB8 [47] is such a method. As a first step, it segments the object in the image. Once a window surrounding the object's projection has been found, another CNN is applied on it to estimate the pixel positions of the corners of the 3D bounding box that encompasses the object. Finally, using PnP with matches from the detected 2D corner positions and the 3D known bounding box corner positions provides the pose estimate.

However, the multi-stage approach of BB8 makes predicting poses slow. Tekin *et al.* [48] follow the same idea of predicting 2D corner projections, combined with PnP, but instead take inspiration from single-shot detection architectures, such as YOLO [49]. This allows to directly predict corner positions without requiring several stages and leads to a much faster solution.

All these vision-based approaches have focused on the general object pose estimation problem, and have not considered its use for inter-robot loop closure detections. As such, they are often not suited for deployment on a mobile robot. For example, in manipulation tasks that often serve as motivations for the proposed solutions, the objects are usually close to the cameras and a robotic arm is used with access to a GPU-equipped desktop for computation. In a multi-robot SLAM setting, robots will only have access to limited computation capabilities and can be order of magnitudes further to the entity whose pose is to be estimated. As such, these approaches provide a solution for inter-robot loop closure detection, but may not be suited to its practical use in a practical multi-robot SLAM scenario.

CHAPTER 3 RESEARCH APPROACH AND THESIS ORGANIZATION

This chapter presents the approach taken to reach the research objectives presented in Sec. 1.3. It details how the contributions presented in Chapters 4 and 5 are consistent with those research objectives, and how they are connected to each other.

The overall objective of this work is to leverage advances in computer vision to support new inter-robot loop closure methods suited to the constraints of multi-robot systems. Two key aspects guided the approaches taken during this work:

- Two main complementary ideas can lead to efficient solutions, and both should be explored.
- Latest advancements in computer vision provide desirable properties for multi-robot systems, and can be used as the basis for inter-robot loop closure detections.

3.1 Exploring solutions for efficient inter-robot loop closure detection

The main challenges reside in the communication and computation constraints as a team of robots grows. These challenges can be handled in two ways. The first solution is to rely on indirect loop closure methods that are designed to find loop closure with minimal, but representative, data and using efficient distribution mechanisms. Such an approach is designed and presented in Chapter 4, in which a new very compact but descriptive representation for the environment is proposed. Descriptors and matching mechanisms are designed for this representation, as well as a decentralized mechanism to perform place recognition using this representation over a fully connected robotic team.

The second approach to handle these challenges is to limit the need to communicate data for loop closure detection by having each robot detecting inter-robot loop closures independently. This is done using a direct inter-robot loop closure method, in which a robot relies on its sensors to estimate the relative pose of another robot. A relative pose estimation approach is designed and presented in Chapter 5. The approach takes advantage of the fact that sequences of images are available in a real scenario, which is often not used in 6D pose estimation methods. It is evaluated by running onboard in a scenario of a flying drone tracking the relative pose of another flying drone.

In both cases, sensors and computation capabilities available during the deployment of robots with embedded computing devices were kept in mind while designing the solutions. Moreover,

these solutions are adaptable and generalizable to a large range of robots and scenarios, making them reusable further than the experiments and conditions considered in experiments of Chapters 4 and 5.

3.2 Building on recent computer vision advancements

In both solutions presented in this thesis, we consider the use of vision sensors which are usually available on robots to perform inter-robot loop closure detections. Hence, it is natural to make use of recent computer vision advances in the design of our methods.

In particular, extracting compact information about the position and identity of rigid bodies visible in images, in the form of a bounding box with a class label for example, is extremely relevant for both direct and indirect inter-robot loop closures. Indirect methods need to rely on descriptions of the environment with minimal data and can make use of this geometric and semantic information to generate such descriptions. Direct methods need to both identify and localize the robot whose relative pose is to be estimated based on images. Computer vision techniques are particularly suited for this task and are reasonably fast on embedded computing devices.

Both methods presented in Chapters 4 and 5 are thus built on top of semantic region extraction systems, and rely on the semantic labels to either build a representation, or to adjust the pose estimation. The solution in Chapter 5 makes additional use of computer vision advancements by relying on visual tracking as a key module of the method.

The solutions presented are applicable in conditions and environments in which the computer vision techniques that are built upon are functional and reliable. This can be done by using pre-trained methods adapted to the situation considered or re-training them for the task at hand.

3.3 Organization of the document

The document is presented in the format of a thesis by articles and follows the recommended structure.

- Chapter 1 introduces the context and basic concepts for this research, and motivates the works that are presented.
- Chapter 2 reviews relevant works for the task at hand in the past literature.

- Chapter 3 situates the contributions within the research objectives and justifies their consistency.
- Chapter 4 is composed of an article accepted for presentation at the *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. The article presents an approach to perform decentralized place recognition using a compact semantic description of places. It proposes a solution to the inter-robot loop closure detection problem with an indirect approach.
- Chapter 5 is composed of an article under review at the *IEEE Robotics and Automation Letters (RA-L)*, 2020. This article presents an approach to visually track the relative pose of objects, including other robots. It can serve as a solution to the inter-robot loop closure detection problem using a direct approach.
- Chapter 6 summarizes and discusses the presented works, and explores their limitations and possible extensions.
- Chapter 7 concludes and summarizes this thesis.
- References for all cited works are presented at the end of this document.

CHAPTER 4 ARTICLE 1: CAPRICORN: Communication Aware Place Recognition using Interpretable Constellations of Objects in Robot Networks

Preface:
Full Citation: Benjamin Ramtoula[1,2], Ricardo de Azambuja[1] and Giovanni Beltrame[1], "CAPRICORN: Communication Aware Place Recognition using Interpretable Constellations of Objects in Robot Networks," *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

Abstract - Using multiple robots for exploring and mapping environments can provide improved robustness and performance, but it can be difficult to implement. In particular, limited communication bandwidth is a considerable constraint when a robot needs to determine if it has visited a location that was previously explored by another robot, as it requires for robots to share descriptions of places they have visited. One way to compress this description is to use *constellations*, groups of 3D points that correspond to the estimate of a set of relative object positions. Constellations maintain the same pattern from different viewpoints and can be robust to illumination changes or dynamic elements. We present a method to extract from these constellations compact spatial and semantic descriptors of the objects in a scene. We use this representation in a 2-step decentralized loop closure verification: first, we distribute the compact semantic descriptors to determine which other robots might have seen scenes with similar objects; then we query matching robots with the full constellation to validate the match using geometric information. The proposed method requires less memory, is more interpretable than global image descriptors, and could be useful for other tasks and interactions with the environment. We validate our system's performance on a TUM RGB-D SLAM sequence and show its benefits in terms of bandwidth requirements.

4.1 Introduction

Simultaneous Localization and Mapping (SLAM) has applications in a variety of scenarios: from aerial robots in GPS-denied environments to vacuum cleaners and self-driving cars, supporting many different types of missions. In some of these missions, a multi-robot SLAM system can improve performance, robustness to individual robot failures, and allow the use of heterogeneous robots with different means of movements and sensors [50].

[1]Department of Computer and Software Engineering, École Polytechnique de Montréal, Canada
[2]School of Engineering, École Polytechnique Fédérale de Lausanne, Switzerland
contact: benjamin.ramtoula@polymtl.ca

Multi-robot SLAM generally follows the same process as single-robot SLAM: each robot uses sensor measurements to perform odometry and has a place recognition system to close loops and correct odometry drift. The robots estimate global trajectories and generate maps solving an optimization problem matching measurements and loop closures.

Robots perform odometry (the first step) locally and they are not affected by any other robot collaborating for the SLAM task. Odometry can, therefore, follow a method from the mature field of single-robot SLAM [51]. However, during place recognition and optimization, the robots need to consider information gathered from other robots working in parallel to take advantage of the multi-robot system and build a global map. For these steps, major adaptations of single robot SLAM techniques are necessary to cope with challenges and constraints inherent to multi-robot systems [50].

There are two extremes when considering sharing information among robots: a centralized system, where a single entity communicates with every robot performing operations as well as giving feedback [52–54]; and a decentralized system, in which robots communicate directly. Centralized systems rely on the central entity to always be functional and scale in computation and bandwidth with the number of robots. On the other hand, decentralized systems do not rely on a single entity and avoid the bottlenecks of centralized systems [55].

In this paper, we present a method for Communication Aware Place Recognition using Interpretable Constellations of Objects in Robot Networks (CAPRICORN). We introduce compact spatial and semantic descriptors which can be compared for place recognition using constellations, a representation of the environment which is particularly well suited for the constraints of multi-robot systems. We also present a mechanism to decentralize it efficiently (Fig. 4.1).

Based on previous ideas [56, 57], and considering the recent advances in real-time object detection methods, RGB-D sensors and embedded AI systems, we build 3D *constellations* of points to represent scenes, where each point approximates the position of an object. The constellation is human understandable and capable of supporting other tasks requiring interaction with the environment. Its shape and scale are robust to viewpoint changes. Since a constellation builds on an object detector module, it can inherit robustness properties which can be associated with the detector, such as invariance to illumination changes. Our method benefits from the fact that object representations require significantly less memory and bandwidth than other representations based on visual features or point clouds [58].

Constellations allow us to perform a decentralized loop closure check in 2 steps. To find loop closures, we first use a semantic descriptor of a constellation which is extremely compact and contains only information concerning object labels. Inspired by the method in [25], each robot shares the descriptor with all other robots, each of which potentially finds similar

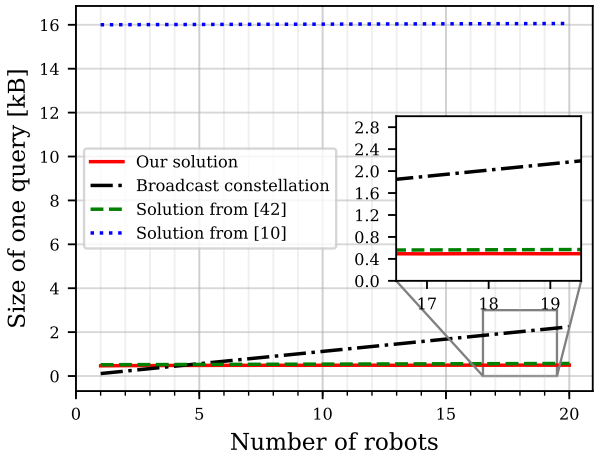

Figure 4.1: Size of a place recognition query over the full team depending on number of robots. "Broadcast constellation" corresponds to sending the full constellation to all robots. 1-hop communication between all robots is considered here.

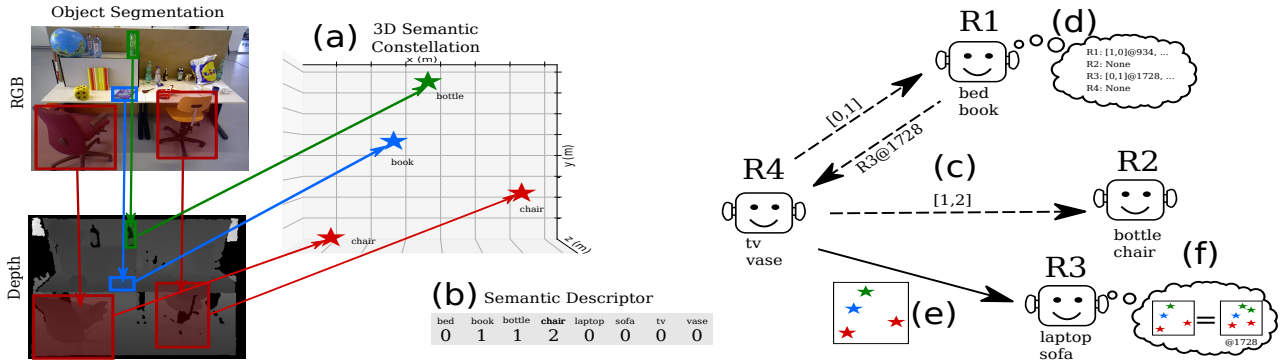

Figure 4.2: Illustration of our full system, in a simplified case with 4 robots. In this example, the detector can predict 8 classes, and R4 is making a query. (a) R4 creates a constellation. (b) R4 generates an associated semantic descriptor. (c) R4 splits and distributes the descriptor among robots responsible for classes seen: R1 (book), and R2 (bottle and chair). (d) R1 and R2 save the sub-descriptors, compare them against all previously received sub-descriptors, and respond with match candidates: R1 responds that R3 at frame 1728 had sent a similar sub-descriptor. (e) R4 sends its full constellation to R3 since it matches best semantically. (f) R3 performs data association between constellations using our geometric surroundings descriptors and produces a final loop closure score. Details of the process including descriptors and comparisons are presented in Section 4.3.

scenes based on specific object labels, and returns candidate scene matches. From these results, the querying robot sends its constellation to the potential matching robots. These, in turn, compute an overall score considering both "which" semantic elements are two frames

(labels), and "how" they are organized in space (relative distances), allowing accurate loop closure detection. The process is illustrated in Fig. 4.2 and detailed in Section 4.3, while experiments and final results are presented in Sections 4.4 and 4.5, respectively. The source code is provided on our repository: `https://github.com/MISTLab/CAPRICORN`, and a brief summary of this work is presented in [59].

4.2 Related work

4.2.1 Visual Place Recognition

A major part of a visual place recognition module is the description of a place [60]. Techniques using solely visual features, either global [15, 61] or local [9, 62], that can be quantized with bag-of-words [12, 13] are most commonly used. However, an important challenge is for the descriptors to exhibit robustness to extreme viewpoint changes as well as dynamic changes in the environment. These changes can severely affect the visual appearance of an image, making methods relying only on visual features fail. Advances in deep learning, specifically convolutional neural networks (CNNs) [17], allow to extract more robust and high-level features from images [63–66]. Recent works propose various methods to make use of CNNs to improve the place recognition performance. Some of these [67–71] make use of learned feature maps of CNNs trained for other tasks such as object proposals, which can be robust to viewpoint and condition changes [21]. NetVLAD [23] is a CNN architecture aimed at creating a global descriptor based on an image, trained end-to-end for place recognition. While these methods improve robustness, they lack interpretability and do not consider computation and communication constraints, which can be significant in a multi-robot system.

4.2.2 Semantic entities for mapping and place recognition

Using semantic entities in the environment for mapping has been gaining popularity in recent years. SLAM++ is an early work which proposed a full SLAM solution based on a set of predefined objects whose 3D models are available [72].

Following works avoided the need for object models by using discovery from point cloud data [73], 2D object detections to create 3D models on the go [74], or by fitting 3D quadrics to objects [75–77]. These SLAM solutions require precise estimates of the objects' poses for loop closures.

Several solutions consider loop closures using graphs or constellations of entities. Finman et al. detect objects using primitive kernels to build a graph [78], from which they perform

place recognition. Gawel et al. propose to build a graph from semantically segmented images for multi-view localization [79]. Frey et al. consider constellations of landmarks which can be matched geometrically to perform data association and map merging in semantic SLAM [57].

4.2.3 Decentralized place recognition

The problem of decentralized place recognition requires a mechanism to share descriptions efficiently. When full connectivity cannot be assumed, some methods rely on sharing information only to neighboring robots [80–82], to deduce global associations. For fully connected teams, Cieslewski and Scaramuzza proposed a way to share a bag-of-words descriptor that is split among every robot such that the amount of data exchanged for a place recognition query is independent of the number of robots [25]. They extended their work using NetVLAD [23] as a global descriptor, with a mechanism allowing each descriptor to be sent only to one specific robot [26]. These solutions usually make use of descriptors which are hardly interpretable and not reusable for other tasks.

4.3 Methodology

Our decentralized place recognition method (Fig. 4.2) relies on the creation of a spatially and semantically meaningful descriptor (a *3D constellation of objects*) associated with each frame coming from an on-board RGB-D camera. Similarly to the work that inspired us using visual descriptors [25], the sharing process requires to pre-assign all possible classes of detectable objects among the robots in the system.

Overall, for each RGB-D frame, we execute the following steps, as illustrated in Fig. 4.2:

1. A querying robot QR creates a constellation (4.3.1)
2. QR generates an associated semantic descriptor (4.3.2)
3. QR distributes the descriptors among a subset of the other robots, based on the objects in the scene (4.3.3)
4. The receiving robots save the descriptors, compute their scores against all previously received descriptors, and respond with match candidates (4.3.3)
5. QR sends its full constellation to the robots with the closest-matching global semantic descriptors (4.3.4)

6. The receiving robots selected in step (e) perform data association between the constellations (4.3.5) and combine the results with global semantic descriptors to make a place recognition decision (4.3.6)

4.3.1 3D Constellations of Objects

Our system consists of a robot set $\Omega = \{\alpha, \beta, \gamma \ldots\}$, where each robot ω has registered a set of frames I_ω. Each frame i contains 3 color channels (RGB) and one depth channel: $i \doteq \{i_R, i_G, i_B, i_D\}$.

From each frame $i \in \{I_\alpha, I_\beta, I_\gamma, \ldots\}$, an object-to-point detection system detects visible objects and associates them to a 3D point expressed in the camera frame. The implementation of this system is left as a design choice depending on types of environments, computing power, and algorithms available.

Each point k generated from a detected object in frame i of robot α is associated with a label $l^k_{\alpha_i}$ taken within L, the set of all detectable indexes which depends on the object-to-point semantic segmentation system. A point also has a 3D position $\mathbf{p}^k_{\alpha_i}$. We define an object as $o^k_{\alpha_i} \doteq \{l^k_{\alpha_i}, \mathbf{p}^k_{\alpha_i}\}$. Therefore, for a frame i seen by robot α we obtain the constellation $\mathcal{C}_{\alpha_i} \doteq \{o^0_{\alpha_i}, o^1_{\alpha_i}, o^2_{\alpha_i}, \ldots\}$.

Since the object-to-point detection system is exchangeable, the proposed approach is straightforward to adapt to new environments with different classes of objects, or simply to improve as state-of-the-art detection tools advance.

4.3.2 Semantic descriptor

We design a semantic descriptor that does not use the 3D information of the constellations to check for semantic consistency. It allows to compress information and to simplify how information is split among robots. This simplified descriptor can be seen as a histogram in which each bin corresponds to a possible class in L. A bin's value is the number of instances of the class in the scene. For constellation $\mathcal{C}_{\alpha_i}$, the semantic descriptor $\mathcal{S}_{\alpha_i}$ has a value in the bin corresponding to class index l given by $\mathcal{S}_{\alpha_i}[l] = |\{o^k_{\alpha_i} | l^k_{\alpha_i} = l \ \forall k\}|$ such that $\mathcal{S}_{\alpha_i} \in \mathbb{N}^{|L|}$. This semantic descriptor $\mathcal{S}_{\alpha_i}$ is very sparse since most classes which can be detected in standard object detectors usually do not appear in the same scene. It differs from the similar semantic words used in SemanticSIFT [83] in that it counts the number of independent entities of a class rather than use a binary indicator of the presence of pixels from a semantic class.

4.3.3 Distributed semantics

To check for similar places seen by other robots in an efficient, scalable, and decentralized way, we use a technique from previous work [25] which we adapt to our descriptors. The idea is to split the global descriptor into partial descriptors which only consider specific classes, and assign them to different robots.

During a query of frame i from robot α, each robot ω, with preassigned label indexes $\text{ł}^\omega \subset L$, is responsible for the partial descriptor $\mathcal{S}^\omega_{\alpha_i} = \mathcal{S}_{\alpha_i}[\text{ł}^\omega]$. Robot ω only receives non-zero values of the partial descriptors to reduce communication bandwidth, benefiting from the sparsity of the semantic descriptor. The receiving robots store the received partial descriptor and compare it to all previously received ones. The comparison of two partial descriptors of $\mathcal{C}_{\alpha_i}$ and $\mathcal{C}_{\beta_j}$ assigned to robot ω is performed using:

$$s^\omega_{\alpha_i,\beta_j} = \frac{\sum_{l\in\text{ł}^\omega} \min(\mathcal{S}^\omega_{\alpha_i}[l], \mathcal{S}^\omega_{\beta_j}[l])}{\sum_{l\in\text{ł}^\omega} \max(\mathcal{S}^\omega_{\alpha_i}[l], \mathcal{S}^\omega_{\beta_j}[l])} \tag{4.1}$$

where $s^\omega_{\alpha_i,\beta_j}$ corresponds to the well-known Jaccard index of the descriptors, also known as Intersection over Union (IOU), commonly used in object detection [84].

Each robot ω then finds frames producing the highest scores and returns the corresponding frame and robot ID to the querying robot α: $(\gamma, m) = \arg\max_{\beta,j} s^\omega_{\alpha_i,\beta_j}$. The number of highest scoring frames returned n_{ret} can be adjusted to the available bandwidth and number of robots. Note that the querying robot is also assigned a subset of labels, and has registered previous queries related to those labels. Hence, it also needs to compare these to its own current query, which requires no external communication.

4.3.4 Deciding on the best global semantic matches

The querying robot α receives robot and frame IDs from all robots detecting partial semantic matches. α must decide which robots to further query for a semantic and geometric comparison of the constellations. Similarly to the geometric check step used in previous works [25,26], we choose to send the full constellation to the robots which provide the highest number of frames returned from the highest scoring partial semantic vectors, as these are more likely to have observed similar scenes. The robots receiving the full constellation compare it to their own recorded constellations.

Again, the number of robots that should be further queried n_{fq} can be controlled and adjusted to the available bandwidth and the number of robots.

4.3.5 Data association

Once a robot β receives a querying constellation $\mathcal{C}_{\alpha_i}$ from another robot α, it compares it to every constellation it has previously recorded. The first step when comparing constellations $\mathcal{C}_{\alpha_i}$ and $\mathcal{C}_{\beta_j}$ is to perform data association: the process of matching points of one constellation to points in another. In other terms, matching specific objects detected in one frame to objects detected in another.

Labels appearing in both constellations are defined as $l_{\alpha_i,\beta_j} \subset L$. For each object k with labels in l_{α_i,β_j}, we define a geometric surroundings descriptor $\mathbf{v}^{\mathbf{k}}_{\omega_\mathbf{m}} \in \mathbb{R}^{|L|}$. Elements in this vector are given the value 0.0 if they correspond to the index of a label not appearing in both constellations. Otherwise, elements are given by the Euclidean distance to the closest instance of the corresponding label in the constellation:

$$\mathbf{v}^{\mathbf{k}}_{\alpha_\mathbf{i}}[l] = \begin{cases} 0.0 & \text{if } l \notin l_{\alpha_i,\beta_j} \\ \min_{n|l^n_{\alpha_i}=l} ||\mathbf{p}^k_{\alpha_i} - \mathbf{p}^n_{\alpha_i}|| & \text{if } l \in l_{\alpha_i,\beta_j} \end{cases} \forall l \in L \tag{4.2}$$

This vector represents how far the closest elements of each appearing label are from a given object. We use this vector to find the most likely matches between $\mathcal{C}_{\alpha_i}$ and $\mathcal{C}_{\beta_j}$, that is finding points with the same label and the closest vectors:

$$\text{match}^k_{\alpha_i,\beta_j} = \underset{n|l^n_{\beta_j}=l^k_{\alpha_i}}{\arg\min} ||\mathbf{v}^{\mathbf{k}}_{\alpha_\mathbf{i}} - \mathbf{v}^{\mathbf{n}}_{\beta_\mathbf{j}}|| \tag{4.3}$$

We then keep the match between the object of index k in $\mathcal{C}_{\alpha_i}$ and the object of index n of $\mathcal{C}_{\beta_j}$ only if they both match each other and their vectors are sufficiently close according to a threshold d:

$$\text{matches}^m_{\alpha_i,\beta_j} = (k,n) \text{ if } \begin{cases} n = \text{match}^k_{\alpha_i,\beta_j} \\ k = \text{match}^n_{\beta_j,\alpha_i} \\ ||\mathbf{v}^{\mathbf{k}}_{\alpha_\mathbf{i}} - \mathbf{v}^{\mathbf{n}}_{\beta_\mathbf{j}}|| < d \end{cases} \tag{4.4}$$

The value of d depends on the estimation noise when reducing an object to a point. d must be high enough so that the noise does not prevent matches, but small enough to reject incorrect matches. It can be tuned with prior experiments by measuring the variance of the estimated point position associated with an object when the viewpoint changes.

Our proposed method combines semantic information and geometric information while being independent of the reference frame from which the 3D positions of points are expressed. It can

detect valid matches and filter out incorrect ones while not requiring additional descriptors associated with each element of the constellations.

4.3.6 Loop closure detection

The matching process (Section 4.3.5) acts as a geometric filter on valid semantic matches: only items of the same class and with similar geometries of surrounding objects are matched. We use this to evaluate a score between two constellations. To obtain a full comparison of two constellations, we compute the Jaccard index over the full semantic descriptors of both constellations (Eq. 4.1, but with s_{α_i,β_j} calculated using $\mathcal{S}_{\alpha_i}$ and $\mathcal{S}_{\beta_j}$). This provides a semantic comparison to which we multiply g_{α_i,β_j}, the fraction of objects successfully matched over the number of common objects in the two constellations. Using this measure, for a place to be recognized we need two constellations to demonstrate similar semantic entities resulting in a high s_{α_i,β_j}, but also for those entities to have similar geometric relationships to produce a high g_{α_i,β_j}.

$$s_{\alpha_i,\beta_j} = \frac{\sum_{l\in L} \min(\mathcal{S}_{\alpha_i}[l], \mathcal{S}_{\beta_j}[l])}{\sum_{l\in L} \max(\mathcal{S}_{\alpha_i}[l], \mathcal{S}_{\beta_j}[l])} \tag{4.5}$$

$$g_{\alpha_i,\beta_j} = \frac{|\text{matches}_{\alpha_i,\beta_j}|}{\sum_{l\in L} \min(\mathcal{S}_{\alpha_i}[l], \mathcal{S}_{\beta_j}[l])} \tag{4.6}$$

$$\text{score}_{\alpha_i,\beta_j} = s_{\alpha_i,\beta_j} . g_{\alpha_i,\beta_j} \tag{4.7}$$

4.4 Experiments

4.4.1 Dataset used and ground-truth

To ensure that our method is able to perform place recognition while also being data-efficient, we test with it using the *freiburg3_long_office_household* sequence of the TUM RGB-D SLAM dataset [85]. To the best of our knowledge, no other datasets provide RGB-D data of environments containing objects and loop closures. As a proof-of-concept, our experiments are based on one sequence since loop closures had to be manually annotated.

The *freiburg3_long_office_household* sequence consists of 2585 frames recorded with a hand-held camera circling around two back-to-back desks with a variety of different objects on them. The alignment of the reference frame with the desks allowed us to combine ground-truth pose of the camera to manual labeling in order to obtain a ground-truth of recognized places. We use the pose of the camera and the depth values to estimate the position of the scene observed in each frame. We can then use these scene positions to estimate distances

between scenes observed in each pair of frames, and we obtain a ground-truth loop closure score based on these distances. We transform the distances to obtain scores between 0.0, when scenes are very far, and 1.0, when the scenes' positions overlap. We verify qualitatively the ground-truth scores by visualizing scores from samples of image pairs. As consecutive frames can easily be detected as loop closures, we ignore closures from frames within 12 seconds (200 frames).

4.4.2 Choices specific to our experiment

Object-to-point detection system

Given a frame i, the channels $\{i_R, i_G, i_B\}$ are given to the real-time object detector YOLOv3 [84] with a zero threshold to detect as many objects as possible. We use the weights supplied by the authors obtained from training on the COCO dataset [86], which can detect 80 objects: $L = \{0, 1, 2, ..., 78, 79\}$. Using the depth channel i_D, we compute the median 3D positions of points within the detected bounding boxes of objects to estimate a point associated to each object.

Preassigning labels

In this work we use a straightforward approach to assign labels to different robots. Each robot is responsible for consecutive label indexes so that all robots have the same number of labels. We leave the study of more complex assignment strategies for future works.

Parameter choice

Based on our experiments, we allow each robot to return up to the $n_{ret} = 4$ highest matching frames when performing the partial semantic comparison (Section 4.3.3), as well as sending its full constellation to up to $n_{fq} = 4$ robots for the geometric and semantic check (Section 4.3.6). This presented a good balance between performance and communication. We used a value of 0.25m for d to approve a match while performing data association.

4.4.3 Evaluation

We simulate our method in a computer program processing all the frames of the *freiburg3_long_office_household* sequence. The program simulates all the communications which would appear between robots as well as the computations happening on each robot. It outputs the final loop closure scores for each pair of frames. We perform the evaluation of

our system in centralized and decentralized cases to verify that our decentralized mechanism does not compromise performance.

In the centralized system, every constellation obtained can be compared to every other constellation registered during the previous frames of the sequence, and only the full comparison of two constellations is performed (4.3.6).

For the decentralized case, we use a similar approach as [26] by splitting the sequence into sub-trajectories, each assigned to a different robot. An example of a split sequence is shown in Fig. 4.3. Moreover, instead of using the method on multiple processes, we simulate the data each robot would have access to and would communicate with. We evaluate both performance and data exchanged between robots. The data used by our representation is shown in Table 4.1.

Table 4.1: Data required for our representations in bytes. Messages which are communicated are shown in bold font.

Value in the semantic descriptor	0.5
Label index	1
Distributed semantic query	number of classes seen $\cdot(1 + 0.5)$
Robot index	1
Frame index	2
Returned robots and frame id	$\leq n_{ret} \cdot (1 + 2)$
Position value (x,y or z)	2
Full check query (constellation)	$\leq n_q \cdot$ number of objects $\cdot(1 + 3 \cdot 2)$

4.5 Results

The similarity matrices obtained in both centralized and decentralized cases are shown in Fig. 4.4. Overall, we can confirm that similar regions of frame pairs produce detected loop closures in both cases. The areas match what we expect from the sequence: the first desk is seen at the beginning with various points of views until frame 450, and then later after frame 1550. The second desk is visible, again with various points of view, between frames 450 and 1550, which also corresponds to a region with consistent matches. Most of the ground truth scores (Section 4.4.1) associated with the detected loop closures are close to 1.0, and concentrated in particular regions, which confirms their validity. Yet, some smaller regions of detected closures are associated to a lower ground truth score. Even though a high ground truth score is a good confirmation of the match, a lower score doesn't necessarily mean that

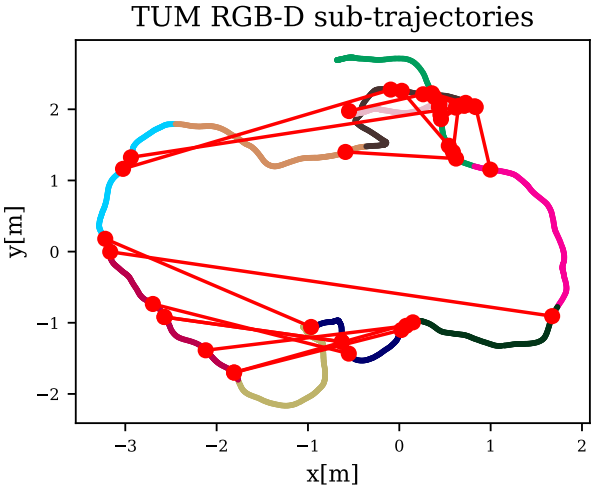

Figure 4.3: Example of the *freiburg3_long_office_household* trajectory split between 10 robots. Red lines indicate detected loop closures when the camera was facing the same area.

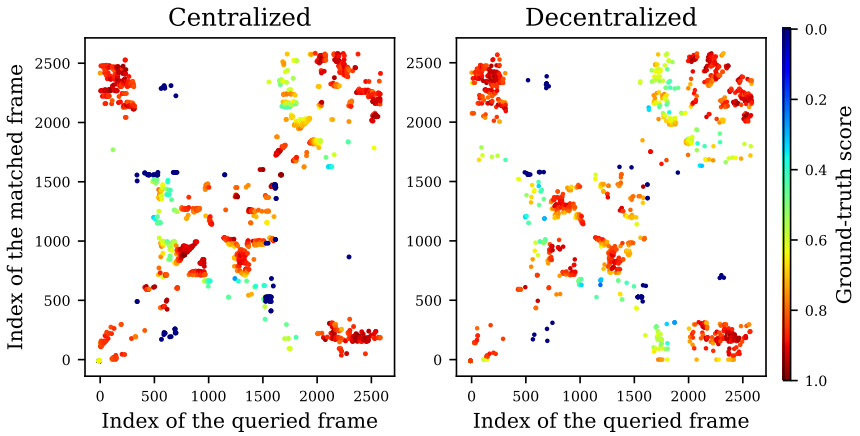

Figure 4.4: Similarity matrices obtained for the centralized solution and decentralized solution with 10 robots. The diagonals are empty due to the removal of neighboring frames during the search. Points show the pairs producing the best loop closure score. Loop closure scores below 0.25 were removed.

the matches are incorrect. A lower score means that estimated regions where the camera was facing in the two frames were considered to be far. However, we have no way of automatically and accurately checking if parts of two frames correspond to the same scene. It is possible that the camera was aimed at different areas which we have calculated to be relatively far, but that part of a same scene was visible. This could lead our system to recognize parts of constellations which would be visible in both frames, even though the ground truth score would be low.

The performance of our system is confirmed by looking at the precision-recall curves, shown in Fig. 4.5. Our centralized system performs very well, obtaining an area under curve (AUC) of 0.9282. The decentralized mechanism slightly affects the performance, reducing the AUC by 8.6%. This drop is close to the drop of 6.7% due to the decentralized mechanism in the solution based on NetVLAD [26]. Additionally, we evaluate a centralized solution using the NetVLAD implementation from DSLAM [55], which performs slightly better than our decentralized solution but not as well as our centralized one. The best recall obtained from the decentralized solution is not as high as the one from the centralized solution. This can be explained as a cost of the decentralization mechanism. In fact, the mechanism will only perform full checks on pairs of frames which were selected on a series of partial semantic comparisons, whereas the centralized version will check every pair. As a result, it is possible that some potentially good matches are ignored in the decentralized case because they don't stand out using the series of partial semantic comparisons. This situation can happen, for example, if two scenes contain common combinations of objects. The objects would be seen often and so the two scenes won't produce particularly good partial semantic scores, and won't be explored further. In the centralized case, those scenes are still checked completely and produce high loop closure scores when considering data association and the geometric score.

We also produce a comparison of data used for our decentralized method and previous works [25, 26], as well as the case where each robot broadcasts its constellation to every other robot in Fig. 4.1. For previous works, the query size is independent of the sequence used, hence we can estimate the amount of data based on the mechanisms proposed. It is clear that all methods making use of a decentralization mechanism do not consume more data as robots are added. Moreover, we achieve query sizes of around 490B, which is comparable to previous work using NetVLAD [26] requiring 512B. However, note that the close score between our solution and the one based on NetVLAD is due to the fact that in the latter, the descriptor is sent to only one robot. Considering cases in which robots may not be able to communicate directly to every other robot, multi-hop communication may be necessary. In that case, sending the NetVLAD descriptor of one frame using 512 bytes through several

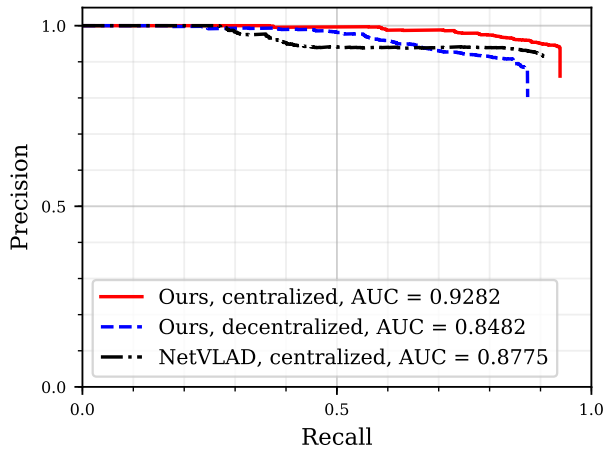

Figure 4.5: Precision recall obtained in both centralized and decentralized cases (10 robots), compared with NetVLAD.

robots will quickly increase necessary bandwidth. In our solution, only very small messages are sent to every robot in the distributed semantic comparison and when performing the full comparison (Table 4.1). In the cases where multi-hop communication is required, our solution will scale better with the number of hops required.

Fig. 4.6 shows that the classes observed in the sequence are unbalanced, and generate higher load for specific robots with our solution. For example, bottles (index 39) are more frequent in our sequence. However, this distribution is meaningful and interpretable: specific relations between objects are common in the environment. For example, keyboards are usually close to screens, and chairs to tables. This may make our method fragile against semantic perceptual aliasing, but recent robust optimization techniques help in that regard [87]. Moreover, certain areas may be associated with specific objects, as acknowledged by research on place categorization [88]. We believe the use of knowledge concerning the distribution of objects in the environment can improve load balancing as future work. We would also like to underline that the use of constellations can, in theory, allow for relative pose estimation between two frames. This step is currently performed through visual features, which turns out to be the most bandwidth-consuming module of DSLAM [55]. Constellations have the potential to greatly improve this step. However, in our experiments, the accuracy and consistency of the object-to-point detector were not sufficient to produce relative pose estimates of satisfying precision.

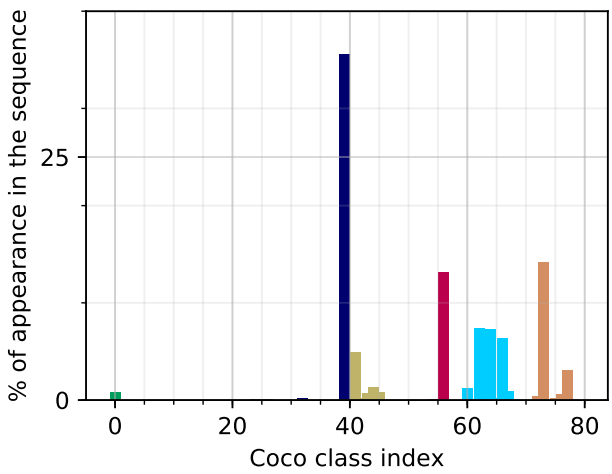

Figure 4.6: Distribution of classes of detected objects during the full sequence with 10 robots. Bars of the same colors represent labels assigned to the same robot.

4.6 Conclusions

We present a new method to perform decentralized place recognition using 3D constellations of objects. These constellations have the advantages of being compact, meaningful, and inheriting robustness and invariance associated with object detectors. Our solution maintains performance when used with a decentralized mechanism in a sequence of the TUM RGB-D SLAM dataset and matches state-of-the-art results in terms of required bandwidth.

CHAPTER 5 ARTICLE 2: MSL-RAPTOR: A Multi-Object Pose Tracker For Onboard Robotic Perception

Preface:

Full Citation: Benjamin Ramtoula, Adam Caccavale, Giovanni Beltrame, Mac Schwager, "MSL-RAPTOR: A Fast Onboard Monocular Multi-Object Pose Tracker," *IEEE Robotics and Automation Letters (RA-L)*, [under review] 2020.

Abstract - Determining the relative position and orientation of objects in an environment is a fundamental building block for a wide range of robotics applications. To accomplish this task efficiently in practical settings, a method must be fast, use common sensors, track multiple dynamic rigid-bodies, and generalize easily to new objects and environments. We present MSL-RAPTOR, a two-stage algorithm for tracking multiple rigid bodies with a monocular camera. The image is first processed by an efficient neural network-based front-end to detect new objects and track 2D bounding boxes between frames. The set of class labels and bounding boxes are passed to the back-end that updates each object's pose using an unscented Kalman filter (UKF). The bounding box uncertainties are fed back to the 2D tracker to improve robustness. Each object's class is identified so a class-specific UKF can be used if custom dynamics and constraints are known. Adapting to track the pose of new classes only requires providing a trained 2D object detector or labeled 2D bounding box data, as well as the approximate size of the objects. The performance on MSL-RAPTOR is first verified on the NOCS-REAL275 dataset, achieving results comparable to RGB-D approaches despite not using depth measurements. When tracking a flying drone from onboard another drone, it outperforms the fastest comparable method in speed by a factor of 3, while giving lower translation and rotation median errors by 66% and 23% respectively.

This research was supported in part by ONR grant number N00014-18-1-2830, the Ford-Stanford Alliance program, a Mitacs Globalink research award, and the Natural Sciences and Engineering Research Council of Canada (NSERC). We are grateful for this support.

B. Ramtoula is with the School of Engineering, École Polytechnique Fédérale de Lausanne, Switzerland and with the Department of Computer and Software Engineering, Polytechnique Montréal, Montreal, Canada benjamin.ramtoula@epfl.ch

A. Caccavale is with the Department of Mechanical Engineering, Stanford University, Stanford, USA awc11@stanford.edu

G. Beltrame is with the Department of Computer and Software Engineering, Polytechnique Montréal, Montreal, Canada giovanni.beltrame@polymtl.ca

M. Schwager is with the Department of Aeronautics & Astronautics, Stanford University, Stanford, USA schwager@stanford.edu

5.1 Introduction

For a robot to reason about its environment and make informed decisions, it is critical to be able to determine the position and orientation of neighboring rigid bodies. From drone racing [89] to object manipulation [90] or autonomous lane changing [91], tasks relying on such estimates are numerous and are important to create truly autonomous robots.

This fundamental capability underpinning a wide array of robotics tasks is called *relative pose estimation*. It consists in estimating the 3D position and the 3D orientation (6D pose) relative to the ego robot using onboard sensors, typically vision. Successful algorithms are accurate, robust, fast, and applicable to a range of situations (e.g. do not require expensive sensors, complex training, or detailed object models). The proposed algorithm meets all of these standards.

Classically, approaches for estimating relative poses from images rely on matching the view of an object to a database of templates or visual features. These methods are limited in the face of scenes with low textures or changing conditions.

Modern methods use neural networks to help overcome these challenges, sometimes by directly estimating the relative pose from each image, or by a hybrid approach in which the network performs the feature extraction and then calculates the relative pose. The design of these methods allows impressive pose estimation performance, but often lacks several of the other desirable properties that would be required in a tool for real-world deployment on robots.

Contribution. With the requirements for onboard robotic perception in mind, we propose MSL-RAPTOR, a method for **M**onocular **S**equential **L**ightweight **R**otation **A**nd **P**osition **T**racking **O**n-**R**obots. It is a modular coupled two-stage algorithm for lightweight multi-object pose tracking relying on sequences of monocular images (Fig. 5.1). The algorithm runs 3X faster than the fastest comparable state-of-the-art method on a Jetson TX2, a common embedded GPU platform from Nvidia. Our approach combines a visual 2D object detection and tracking front-end with a classical filtering algorithm, the unscented Kalman filter (UKF), as the back-end. To track a pose, MSL-RAPTOR only requires approximate knowledge of the size of the tracked objects (i.e. the dimensions of the 3D bounding box), and the ability to detect their 2D bounding boxes from monocular images.

The front-end of the algorithm has two modes. *Detection* finds 2D axis-aligned bounding boxes of objects with their class type from an image. *Tracking* uses the previous image cropped to the object's bounding box and the current image to output the new bounding box of that object. This step is quicker than *detection* and can provide more informative

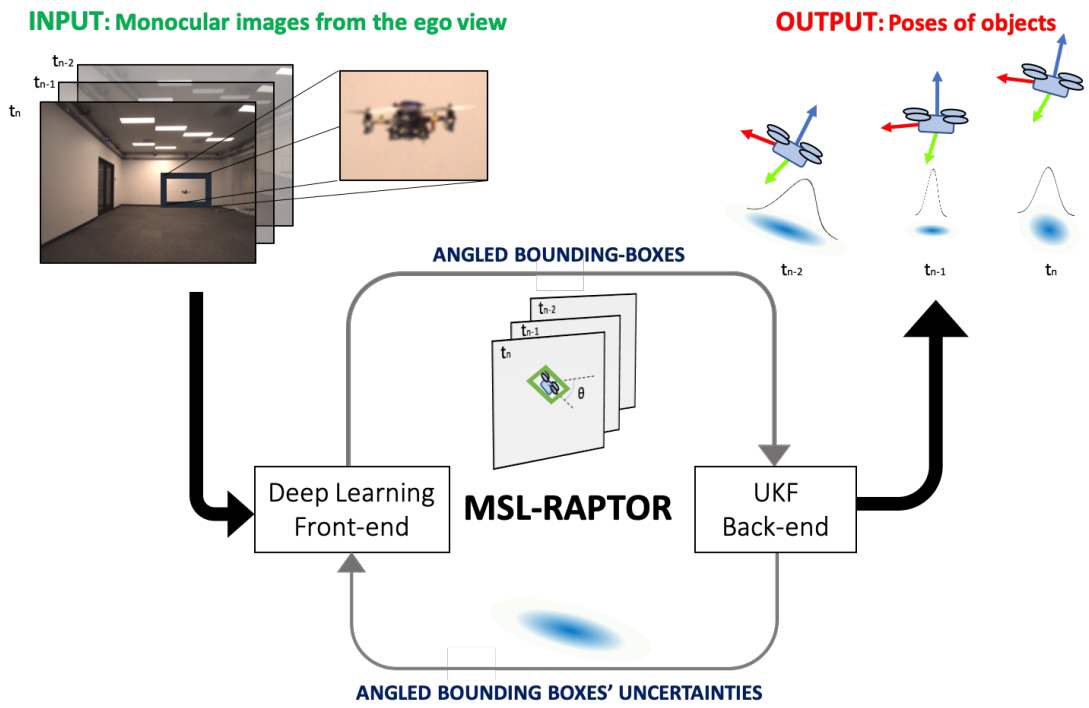

Figure 5.1: MSL-RAPTOR relies on sequences of monocular images to estimate distributions over poses of tracked objects. Its front-end produces angled bounding boxes which are used by the back-end to generate pose estimates, as well as measurement uncertainties that are returned to the front-end to improve bounding box tracking.

angled bounding boxes. In general, the pose that produces a given 2D bounding box cannot be determined uniquely (Fig. 5.2), so a filter analyzing a sequence of these measurements is used to determine the likely pose of the object.

These measurements are then passed to the algorithm's back-end, which runs a class-specific unscented Kalman filter (UKFs) [92,93] to estimate the posterior of the pose for each tracked object. The measurement's uncertainty for each object is fed back to the front-end for use in probabilistically determining when re-detecting objects is necessary due to uncertain tracking (often caused by occlusions), or to associate detections to previously-tracked objects.

MSL-RAPTOR attains competitive performance on the NOCS-REAL275 dataset [94] with considerably fewer requirements than state-of-the-art methods in sensor modality and object-specific training. Furthermore, we also show its adequacy for onboard perception in a practical experiment in which a flying drone is required to estimate the pose of another drone. In our perception setup, the speed of the fastest state-of-the-art method [48] (referred to here as SSP) drops from 30 Hz to 3 Hz when switching from a desktop GPU to an Nvidia Jetson TX2 embedded computer. We show that MSL-RAPTOR achieves 66% lower translation and

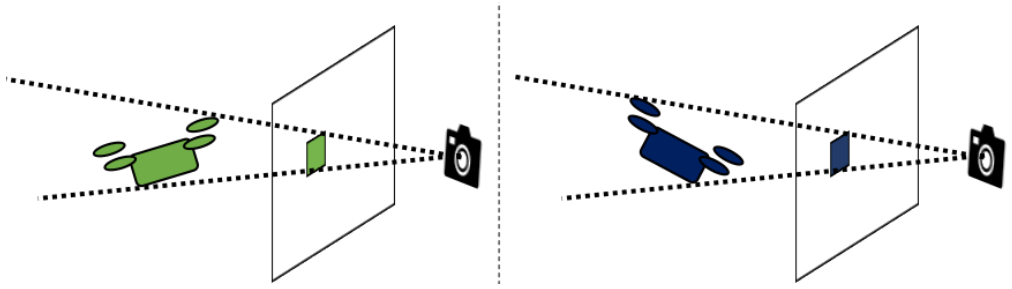

Figure 5.2: Illustration of the pose ambiguity from bounding box measurements. Several poses can lead to the same projected box on the image.

23% lower rotation error while running onboard at ∼9 Hz.

5.2 Related Work

Our method combines and builds upon related work in the areas of object detection, visual tracking, pose estimation, and probabilistic filtering.

5.2.1 Visual data extraction

Our system relies on monocular images as its sensory inputs, and as such we make use of recent advances in computer vision powered by deep learning to extract semantic and geometric information. Two of the most relevant areas of visual data extraction are detection and tracking. Detection involves identifying an object in an image and locating areas of pixels that represent it. Tracking assumes the object has already been detected in a previous frame and uses that prior to propagate pixels corresponding to projections of the same object in the next frame.

Detection

YOLOv3 [49] and SSD [95] are two well-known detection methods that take advantage of the fact that object-classification networks are fast by applying this technique at different positions and scales to quickly detect objects. The latest version of YOLO incorporates the ResNet network architecture [96] that is designed to improve training on deep learning networks and has been shown to perform well on image processing tasks. MAVNet [97] is a network designed specifically for object detection by micro-unmanned aerial vehicles (UAVs), with a focus on fast computation for onboard robotic perception in scenarios that are often more challenging than less-realistic datasets. While previous methods do not make use of

the continuity of images in the real-world, some recent works take advantage of sequences of images for detection, for example by using shifting views to differentiate the foreground from the background [98].

Tracking

Tracking methods leverage the correlation between time and object position for a sequence of images. Given a previous segmentation of an object, an updated region can be found with less computation since it is likely that the object has not moved far between frames. Most of these methods are class agnostic; they can track an object frame-to-frame without knowing anything a priori about the object other than its first location. Siamese networks [99] have been at the origin of recent state-of-the-art results [100, 101]. SiamMask [100] in particular uses a binary segmentation mask that improves training and provides pixel-wise segmentation and angled bounding boxes of the object. THOR [102] builds a framework to augment tracking methods by aggregating a set of different templates (i.e. views).

5.2.2 Pose prediction

There are many approaches in the computer vision literature to estimate the pose of an object from a single monocular image, though many require depth-sensing in addition to an RGB image. As this task is more challenging than generating a 2D bounding box, most of these methods do not run in real-time.

PoseCNN [103] and LieNets [46] use convolutional neural networks (CNN) to regress a pose estimate. A downside of large CNN-based methods is they can be slow, though LieNets is relatively fast at 10 Hz on a desktop GPU. Approaches such as BB8 [47], SSD-6D [44], DOPE [90] and SingleShotPose [48] estimate the 2D projections of the 3D bounding box encompassing the object and solve the Perspective-n-Point (PnP) problem to obtain a pose estimate [104]. NOCS [94] instead presents a category-level canonical representation in which pixel correspondences are found and combined with depth data to estimate the pose. These methods are the most efficient of the existing deep pose estimation techniques, but even the fastest cannot achieve real-time performance on embedded systems. MSL-RAPTOR is able to achieve these speeds on much more limited hardware by leveraging the relationship between sequential images for speed and memory efficiency.

5.2.3 Pose tracking

For onboard robotic perception it is common to have access to sequences of images. These provide more information that can help pose estimation. Recent works have sought to blend deep learning advances, particularly for perception, with those of probabilistic filtering. PoseRBPF [105] decouples the translation and orientation estimation problems. The translation is estimated and fed into a network that extracts a latent representation of the cropped view of the object. Along with a pre-computed codebook for each object, a Rao-Blackwellized particle filter (PF) is used to estimate orientation. Limiting the PF to just orientation allows this method to reach between 20 Hz for 50 particles to 3 Hz for 400 particles on a desktop GPU. The nature of this filter does not admit knowledge of the object dynamics, which lessens the multi-hypothesis tracking capability offered by a PF.

6-PACK [106] is another pose estimation method that leverages temporal consistency by learning to represent objects by keypoints that are matched between frames. It obtains impressive state-of-the-art category-level pose estimation results, but contrary to MSL-RAPTOR, it relies on RGB-D data and is unable to run real-time on an embedded device.

The authors of DART [107] eschew neural networks and instead perform iterative optimizations given a depth map at each time step and knowing the signed distance function of the tracked object. This method can reach 30 Hz on a powerful GPU, but the assumptions required for sensing and knowledge of the object make it ill-suited to general pose estimation in robotics.

To our knowledge, no method offers MSL-RAPTOR's speed and performance for onboard robot perception.

5.3 Problem Formulation

We consider the scenario where a computationally-limited mobile robotic agent with a calibrated monocular camera is performing a task requiring the identification and tracking of the relative poses of other dynamic objects in the environment. Let $\bar{\mathbf{O}}$ be the set of all objects, and $\mathbf{O}^t \subseteq \bar{\mathbf{O}}$ be the subset of visible objects at time t when an RBG image $I^t \in \mathbb{R}^{W \times H}$ is captured. The state of the i^{th} object $O_i^t \in \mathbf{O}^t$ is represented as $\mathbf{x}_i^t = [\mathbf{p}_i^t, \mathbf{v}_i^t, \mathbf{q}_i^t, \boldsymbol{\omega}_i^t] \in \mathbb{R}^{13}$ where $\mathbf{p}_i^t \in \mathbb{R}^3$ is 3D position, $\mathbf{v}_i^t \in \mathbb{R}^3$ is linear velocity, $\mathbf{q}_i^t \in \mathbb{R} \times \mathbb{R}^3$ is a quaternion representation of orientation, $\boldsymbol{\omega}_i^t \in \mathbb{R}^3$ is angular velocity. The position and orientation component of this state can be represented as the objects pose $\mathbf{T}_i^t \in SE(3)$. Henceforth the time superscripts will be dropped for notational convenience unless they are needed for clarity.

Each object O_i has a span in the 3D space that can be approximated with a 3D bounding

Figure 5.3: The bottle's axis-aligned (red) and angled (green) bounding boxes. The angle allows for a tighter, more informative, fit of the bottle.

box $\mathbf{B}_i \in \mathbb{R}^{8\times 3}$ and a class label (e.g. `drone`, `bottle`, etc) denoted c_i. Additionally, each class of object will have an associated simple model used to approximate the dynamics of the object and update the components of the state: $\mathbf{x}_i^{t_2} = \mathbf{F}(\mathbf{x}_i^{t_1})$. By default, a simple constant velocity model is used,

$$\mathbf{x}_i^{t_2} = \mathbf{F}(\mathbf{x}_i^{t_1}) = \begin{cases} \mathbf{p}_i^{t_2} = \mathbf{p}_i^{t_1} + \mathbf{v}_i^{t_1}(t_2 - t_1) \\ \mathbf{v}_i^{t_2} = \mathbf{v}_i^{t_1} \\ \theta \doteq |\boldsymbol{\omega}_i^{t_1}|(t_2 - t_1) \\ \Delta\mathbf{q} \doteq \left[\cos\left(\frac{\theta}{2}\right), \frac{\boldsymbol{\omega}_i^{t_1}}{|\boldsymbol{\omega}_i^{t_1}|}\sin\left(\frac{\theta}{2}\right)\right] \\ \mathbf{q}_i^{t_2} = \mathbf{q}_i^{t_1} \otimes \Delta\mathbf{q} \\ \boldsymbol{\omega}_i^{t_2} = \boldsymbol{\omega}_i^{t_1} \end{cases} \tag{5.1}$$

but more sophisticated models may be provided. In general, no knowledge of the tracked object's control input is assumed. Note that $\otimes$ refers to quaternion multiplication [108].

When processing an image, 2D projections of the objects O_i onto the image I are enclosed in minimum-area bounding boxes: $\mathbf{z}_i = [x_i, y_i, w_i, h_i, \alpha_i]$ where x_i and y_i are the column and row of the box center, w_i and h_i are the box width and height, and α_i is the box angle from the horizontal image axis. If instead, an axis-aligned bounding box is produced (i.e. $\alpha = 0$), it is denoted $\tilde{\mathbf{z}}_i$ (Fig. 5.3). The distributions over states and measurements are both modeled as Gaussians, $\mathcal{N}(\mathbf{x}_i^t, \Sigma_i^t)$ and $\mathcal{N}(\mathbf{z}_i^t, \mathbf{S}_i^t)$ respectively.

5.4 Algorithm

In this section, we describe the architecture of our approach that is illustrated in Fig. 5.4. The front-end has two components for processing the images: a detector and a visual tracker. The detector can identify all objects in an image and their class labels, but is slower and only generates axis-aligned bounding boxes. The tracker is faster and produces angled bounding boxes, but requires an accurate prior image location of the object. The back-end relies on a class-specific UKF associated to each tracked object. It takes in angled bounding box measurements from the front-end and produces pose estimates. The decision to use the tracker vs. detection is determined by the measures of uncertainty of each object's bounding box generated by the back-end's UKFs.

5.4.1 Object detection

Upon receiving an image I, the angled bounding boxes must be extracted. If no objects have previously been located, or if re-detection has been triggered, a pre-trained, out-of-the-box object detector outputs the axis-aligned bounding box and class label for each visible object $\mathbb{D}(I) \rightarrow \{(\tilde{\mathbf{z}}_i, c_i) | O_i \in \mathbf{O}\}$.

Detection will find all objects in the scene, even those already being tracked. After acquiring $\tilde{\mathbf{z}}_i \forall i$, each bounding box is either associated with an existing tracked object or classified as new through the data-association process described in Sec. 5.4.4. Knowing the correspondences allows proper routing of measurements to the appropriate existing UKFs in the back-end for previously tracked objects. For each object O_i classified as new or needing re-detection, the tracking network is used to initialize the visual tracker (Sec. 5.4.2) using the detected box and produces an angled bounding box: $\mathbb{T}(I^t, \tilde{\mathbf{z}}_i^t) \rightarrow \mathbf{z}_i^t$. This, along with the class label c_i from the detector, serves to initialize the UKF (Sec. 5.4.3). Because the filter requires an initial pose estimate, the initial 6D pose is roughly approximated assuming the width, height, and roll are aligned with the 2D bounding box. This is a poor assumption in general, but the robustness of the filter will compensate for the initialization error. Since the class of the object is also identified, class-based rules for initialization can also be used (e.g. mugs are usually upright).

5.4.2 Visual tracking

Once an object has been detected and the associated tracker and UKF are initialized, MSL-RAPTOR's front-end only relies on the tracker to provide measurements to the UKF. The tracker is an object-agnostic method, which produces angled bounding box measurements

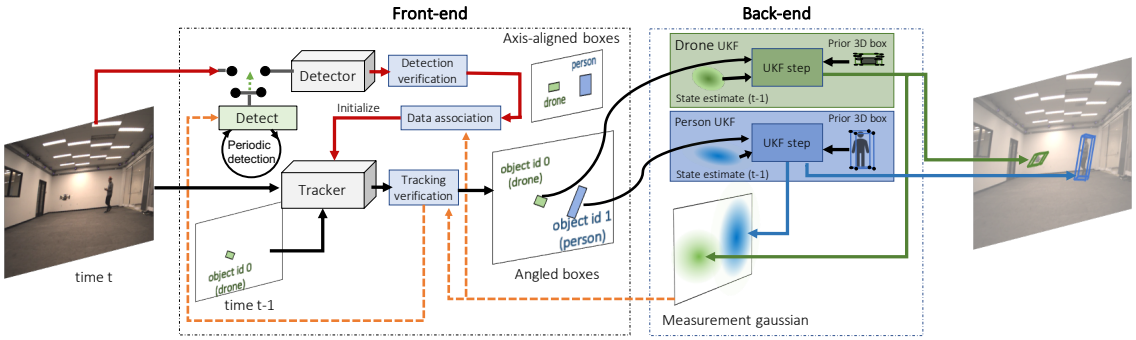

Figure 5.4: Overview of MSL-RAPTOR. The front-end takes in monocular images and produces angled-bounding box measurements of the tracked objects, associated to their class information. The back-end makes use of those measurements for a class-specific unscented Kalman filter, which generates pose estimates as well as measurement distribution approximations. The latter are re-used by the front-end to evaluate the performance of the visual tracking.

given the previous bounding box and a new image $\mathbb{T}(I^t, \mathbf{z}_i^{t-1}) \rightarrow \mathbf{z}_i^t$. Unlike the detector, the tracker only considers one object O_i at a time, but may be called for each tracked object. When a new image is available, the tracker generates angled bounding boxes that serve as observations for the UKFs predicting the objects' poses (Sec. 5.4.3).

Aside from the speed benefit, we use a visual tracking method that produces angled bounding boxes that are more informative measurements than the axis-aligned boxes. While the angle is still not sufficient to uniquely determine an object's pose, it reduces the set of poses that would result in the given measurement which improves tracking performance. Another benefit of visual tracking is the implicit correspondence between measurements and objects. Except for occlusions (handled separately), the output of the tracker called with input $\mathbf{z}_i^{t-1}$ is the bounding box of the same object at the current time $\mathbf{z}_i^t$.

5.4.3 Unscented Kalman Filter

To estimate pose from a series of angled bounding box measurements, we use a class-dependent unscented Kalman filter for each tracked object O_i. Probabilistic filters like the UKF consist of a prediction and update step. During prediction, the state is propagated forward using the object dynamics, and the expected measurement from the resulting anticipated state is calculated. In the update step, the difference between the predicted and actual measurement is used to adjust the state estimate and uncertainty accordingly. A typical output of a UKF is the mean and covariance of the state, but as part of the update process, the measurement mean and covariance ($\hat{\mathbf{z}}_i$ and $\hat{\mathbf{S}}_i$) are also calculated. These are passed

to the front-end for use in data association and determining whether to trigger re-detection (Sec. 5.4.4).

The prediction step of the UKF relies on a dynamics model, which can depend on the object tracked. The MSL-RAPTOR framework gives access to a class label for each object, allowing to use class-specific dynamics if available, or a simple constant velocity model if not (5.1). For systems without significant inertial effects, this is a good approximation, otherwise, the robustness of the filter is relied upon to correct for the modeling error.

To predict the expected measurement associated to a pose, the knowledge of the 3D bounding box of the tracked object is used. The vertices of this box $\mathbf{B}_i$ are projected into the image plane using the known camera intrinsic matrix K and the estimated $\hat{\mathbf{T}}_i$ (the object's pose in the camera frame) [109]. The minimum area rectangle that encloses these projected pixels is the predicted measurement $\pi_{\mathbf{K}}(\mathbf{B}_i, \hat{\mathbf{T}}_i) \rightarrow \hat{\mathbf{z}}_i$.

A UKF is chosen over alternatives such as an EKF or PF due to its efficient handling of the non-linearities inherent in the system. The PF's natural ability to track multiple-hypothesis potentially could allow it to better determine poses that best explain the series of bounding box measurements, but empirical testing showed the computational requirements of this method to be too high to run onboard.

5.4.4 Updating the tracked objects

Occlusions, objects leaving the field of view, and poor measurements (e.g. due to light reflections) can all cause tracking failures that lead to inaccurate bounding boxes. No assumptions are made regarding these scenarios, so the front-end must be able to handle these situations.

One solution is to rely on detection for situations where an object has become unoccluded or has entered the image. To compromise between speed and the ability to identify new objects, it is periodically triggered, which has the advantage of compensating for any drift or errors that might occur in the visual tracking. The front-end's verification mechanisms can also trigger detection to correct for uncertain measurements.

Bounding Box Verification

Detected measurements Detection returns both the bounding box and the class label of the object. This can be used to filter detections by class and to set class-specific rejection conditions for the measurement that can help prevent false-positive matches. For example, an aspect ratio condition can be set so that a bounding box $\tilde{\mathbf{z}}_i$ is valid only if it is in a valid

class-specific aspect-ratio range:

$$\Gamma_{\min}(c_i) < \tilde{w}_i/\tilde{h}_i < \Gamma_{\max}(c_i),$$

where the thresholds $\Gamma_{\min}(c_i)$ and $\Gamma_{\max}(c_i)$ can be determined from the typical geometry of the class. Additionally, a threshold on the minimum distance $d_{\min}$ to the edge of the image can prevent bad measurements due to truncated bounding boxes:

$$\begin{aligned} d_{\min} < \tilde{x} - \tilde{w}/2 < \tilde{x} + \tilde{w}/2 < W - d_{\min} \\ d_{\min} < \tilde{y} - \tilde{h}/2 < \tilde{y} + \tilde{h}/2 < H - d_{\min}. \end{aligned}$$

Tracker measurements At each iteration of the UKF for object O_i, anticipated measurements are predicted for a range of perturbations about the current state estimate. From this set of possible measurements, a mean $\hat{\mathbf{z}}_i$ and covariance $\hat{\mathbf{S}}_i$ representing the distribution of observations is calculated. These probabilistic metrics can be used to evaluate the likelihood of new measurements. If a mismatch between the state estimate of the back-end and the new measurement is detected, this could indicate drift in the tracker or some other failure, and detection is triggered.

To verify that the most recent tracking measurement $\mathbf{z}_i$ is acceptable, we compute the Mahalanobis distance as

$$\mathbb{M}(\mathbf{z}_i, \hat{\mathbf{z}}_i, \hat{\mathbf{S}}_i) \doteq \sqrt{(\mathbf{z}_i - \hat{\mathbf{z}}_i)^T \hat{\mathbf{S}}_i^{-1} (\mathbf{z}_i - \hat{\mathbf{z}}_i)} < \gamma. \tag{5.2}$$

Additional checks are performed to detect if the tracked object has reached the edge of the field of view using the z-test with the diagonal values of $\hat{\mathbf{S}}_i$ ($\hat{\mathbf{S}}_{x,x}$, $\hat{\mathbf{S}}_{y,y}$, $\hat{\mathbf{S}}_{w,w}$, and $\hat{\mathbf{S}}_{h,h}$) for each edge of the bounding box:

$$\begin{aligned} (0 - (x_i - w_i/2))/\sqrt{\hat{\mathbf{S}}_{x,x} + \hat{\mathbf{S}}_{w,w}/4} > -\zeta \\ (0 - (y_i - h_i/2))/\sqrt{\hat{\mathbf{S}}_{y,y} + \hat{\mathbf{S}}_{h,h}/4} > -\zeta \\ (W - (x_i + w_i/2))/\sqrt{\hat{\mathbf{S}}_{x,x} + \hat{\mathbf{S}}_{w,w}/4} < \zeta \\ (H - (y_i + h_i/2))/\sqrt{\hat{\mathbf{S}}_{y,y} + \hat{\mathbf{S}}_{h,h}/4} < \zeta. \end{aligned}$$

The thresholds γ and ζ can be chosen from the quantiles of the chi-squared distribution and normal distribution, respectively. When the test fails, the system switches into detection mode to either confirm it should stop tracking the object, or to correct its pose estimate and

tracked bounding box.

Data-association

After a detection, it is unknown which of the bounding boxes correspond to previously tracked objects or newly visible objects. To associate these observations to previously tracked objects, we use the Mahalanobis distance (5.2). For each measurement $\mathbf{z}_j^t$ and measurement distribution $\hat{\mathbf{z}}_i^{t-1}$ of tracked object of the same class $c_i^{t-1} = c_j^t$, the Mahalanobis distance $\mathbb{M}(\mathbf{z}_j^t, \hat{\mathbf{z}}_i^{t-1}, \hat{S}_i^{t-1})$ is evaluated. The lowest of these distances that meets a minimum threshold chosen with a quantile of the chi-squared distribution is accepted as a valid correspondence. If no distances are accepted, it is handled as a new detection (Sec. 5.4.1).

5.5 Experimental results

5.5.1 Implementation choices

For our experiments we implement MSL-RAPTOR with YOLOv3 [49] from Ultralytics [110] for our front-end's object detector. We chose YOLOv3 due to its ubiquity, speed, and demonstrated ability to detect a range of objects. To support the objects used in our experiments, we retrain it on the COCO dataset mixed with a custom dataset containing images of a flying quadrotor, which leads to 81 supported object categories. For our visual tracker, we rely on the implementation of SiamMask [100] provided by the authors, which is object-agnostic and therefore does not require retraining. We chose SiamMask for its speed, ability to produce angled bounding boxes, and impressive tracking performance. We use the provided weights trained on the VOT dataset [111].

5.5.2 Evaluation on the NOCS-REAL275 dataset

Experiment

Environment and scenarios We begin by evaluating MSL-RAPTOR on the NOCS-REAL275 dataset [94] to answer a fundamental question about the method: can the algorithm track pose accurately for a range of objects? In all, the dataset contains seven training videos and six testing videos from a camera moving in different scenes. The scenes contain objects with class labels in {`bowl`, `mug`, `can`, `bottle`, `laptop`, and `camera`}. We tune our class-specific UKF parameters using the training set and present the results evaluated on the testing set. Similarly to the baseline methods, we do not penalize orientation errors about axes of symmetry for the `can`, `bowl` and `bottle` classes.

Implementation details Our re-detections and verification mechanisms presented in 5.4.4 serve to make the tracking robust to occlusions and to handle objects exiting and entering the frame. However, sequences of the dataset are relatively short and mostly maintain all objects in the field of view. Moreover, the datasets used to train our detector $\mathbb{D}$ did not contain instances of all the classes appearing in the scenes, and the training set from NOCS-REAL275 would not be general enough to train the detector to properly detect new instances of objects in the testing set. For these reasons, we disable detection and verification mechanisms for evaluation on this dataset and rely only on visual tracking in the front-end. This is a conservative approach as violations of these assumptions will reduce our performance.

To replace MSL-RAPTOR's filter initialization, which is based on the disabled detection, the same method is used as the baselines [106, 112, 113]. For all objects $O_i \in \bar{\mathbf{O}}$, random uniform translation noise of up to 4cm is added to the ground truth initial pose $\mathbf{T}_i^0$. From this initialized pose, the first bounding box is calculated using the UKF's measurement prediction function $\pi_{\mathbf{K}}(\mathbf{B}, \mathbf{T}_i^0) \rightarrow \tilde{\mathbf{z}}_i^0$. We add uniformly sampled noise of up to 5% of the width and height of the box to its row and width, and its column and height respectively. With this first bounding box, the visual tracker $\mathbb{T}$ will generate future measurements.

For objects in the dataset with shapes not well approximated by a 3D bounding box (bowl, laptop, mug), we add additional vertices to restrict the volume. For example, 12 points are used instead of 8 for the laptop, resulting in an "L" shape when viewed from the side. For mugs and bowls, extra points allow a tighter fit around curved rims. Using these refined volumes reduces the mismatch between predicted and observed measurements in the UKF, which can lead to incorrect adjustments of the pose estimate. This effect is greatest when objects are close to the camera, as in the NOCS dataset.

Finally, as the baseline methods are evaluated using desktop GPUs, for this set of experiments only we use a desktop computer with an Nvidia RTX 2060 GPU.

Baselines We compare our results with those reported by Wang *et al.* over the same dataset [106] for four methods, **all of which use RGB-D input**:

- NOCS [94], a recent 6D pose estimation solution relying on pixel correspondences to a canonical representation.
- ICP [114], the Open3D implementation of the iterative closest point algorithm.
- KeypointNet [115], a category-based 3D keypoint generator.
- 6-PACK [106], a state-of-the-art method using learned 3D keypoints with an anchoring

mechanism incorporating temporal information for improved tracking.

Results

The average rotation and translation errors over each class are presented in Tab. 5.1 and the accuracy visualized in Fig. 5.5. Despite relying only on monocular images, MSL-RAPTOR achieves comparable performance to state-of-the-art methods that use RBG-D.

As is expected for a monocular camera, depth estimates are less accurate than translations perpendicular to the agent's camera axis (i.e. parallel to the image plane). This can be seen in Fig. 5.5c where these are broken out into separate errors. This is a problem that comes with the use of monocular images rather than RGB-D data.

5.5.3 Baseline evaluation in an aerial robotics scenario

To demonstrate that MSL-RAPTOR provides robust onboard robotic perception, we implemented the algorithm on a drone while tracking another drone. Additional data was collected from this system, and processed offline to allow comparison to baseline methods and for parameter tuning. By design, MSL-RAPTOR is lightweight, relying on the probabilistic output of the back-end to avoid slower front-end calculations. To make our system perform even faster we integrated TensorRT into our code which streamlines our tracking network by reducing the computational cost of each visual tracking call.

Experiment

Hardware The ego quadrotor uses the DJI F330 frame and is equipped with off-the-shelf and custom made components as shown in Fig. 5.6. The quadrotor is equipped with a perception-specific module that contains a global shutter RGB camera (480×640 resolution) and a TX2 for real-time image processing. The ego pose is fused with the onboard IMU using the Pixhawk flight controller's EKF filter for full ego state estimation.

Environment and scenarios The experiments were performed in Stanford University's Boeing Autonomous Flight Laboratory that is equipped with an Optitrack motion capture system for obtaining ground truth poses of the ego drone and tracked objects. Scenarios were tested where the ego drone moves about the experimental space (causing the background to change substantially), a challenge which demonstrates the robustness of the system. The flights went from slow to aggressive, and the tracked bodies could exit the field of view and return, and were occasionally occluded by other objects. As would be expected in collision

Table 5.1: Results on NOCS-REAL275. Rotation errors are reported in degrees and translation errors in centimeters.

Method		NOCS	ICP	Keypoint Net	6-PACK	Ours
Modality		RGB-D	RGB-D	RGB-D	RGB-D	**Mono.**
Bottle	R_{err}	25.6	48.0	28.5	15.6	18.7
	t_{err}	1.2	4.7	8.2	1.7	9.8
Bowl	R_{err}	4.7	19.0	9.8	5.2	16.2
	t_{err}	3.1	12.2	8.5	5.6	4.0
Camera	R_{err}	33.8	80.5	45.2	35.7	23.4
	t_{err}	14.4	15.7	9.5	4.0	8.5
Can	R_{err}	16.9	47.1	28.8	13.9	17.6
	t_{err}	4.0	9.4	13.1	4.8	9.0
Laptop	R_{err}	8.6	37.7	6.5	4.7	17.4
	t_{err}	2.4	9.2	4.4	2.5	11.6
Mug	R_{err}	31.5	56.3	61.2	21.3	35.3
	t_{err}	4.0	9.2	6.7	2.3	7.7
Overall	R_{err}	20.2	48.1	30.0	16.0	21.8
	t_{err}	4.9	10.5	8.4	3.5	8.2

avoidance situations, the distances between the ego and tracked drone routinely reached 8 meters (several times further than scenarios usually covered by pose estimation datasets). This provides important validation of the use of this method for planning and decision making onboard agile robots. From our collected data, we use 5500 images over several runs to form a training set, and use 5000 different images to form a testing set.

Implementation details We do not assume knowledge about the control input of the tracked drones, and the basic dynamics in (5.1) are assumed. Since we are looking at tracking the quad at up to 10 meters distance from the ego drone, we treat the tracked quad as symmetric about the vertical axis considering the yaw angle does not have an appreciable effect on our measurements, and does not provide useful information about its behavior. For a fair comparison, the same assumption is made for the baseline method. With a higher resolution camera or for tracing at closer ranges this constraint could be removed.

Baseline While most state-of-the-art methods struggle to achieve real-time speeds on desktop GPUs, SSP reports inference-only speeds of 50 Hz. This, in addition to its impressive accuracy, makes it the only candidate likely to be competitive when running onboard. SSP takes in an image and outputs the estimated locations of the projected 3D bounding box of the object on the image plane. Its network is based on YOLO's architecture and is modified to produce the 3D bounding box projections instead of a 2D bounding box. The relative rigid body transformation is then calculated from the projected vertex locations and the 3D bounding box geometry using a PnP algorithm.

SSP requires the full object pose for each image when training. Outside of laboratory environments with specialized equipment, this is more challenging to acquire than the 2D bounding boxes used by MSL-RAPTOR. Moreover, SSP is presented as an instance-based method, meaning it must be trained on specific objects for which it will infer pose. For this reason, we did not evaluate it on the NOCS dataset that contains unseen objects instances in the testing set.

We train SSP using a desktop GPU, and present results for both SSP and MSL-RAPTOR evaluated on an Nvidia Jetson TX2.

Results

Fig. 5.7 highlights the improvement in accuracy MSL-RAPTOR offers compared to SSP. The median translation and rotation errors are 0.82 m and 8.59 degrees for MSL-RAPTOR compared to 2.45 m and 11.26 degrees for SSP (66% less translation error and 23% less rotation error). When tracking a flying drone, MSL-RAPTOR runs three times faster than SSP with average speeds of 0.115 s (8.7 Hz) and 0.335 s (2.98 Hz) respectively.

SSP directly estimates the locations of the 3D bounding box projections on the image. However, in images where the drone is far away or moving quickly (causing blur), these projected points are warped and scaled (see Fig. 5.8). The PnP algorithm will find a rigid body transformation that best matches these projections to the actual 3D bounding box and this directly leads to the vast depth errors as seen in Fig. 5.7c. Since the in-plane translation is affected by the average of these points, and the rotation by the relative positioning, there is a much less stark difference in performance for the in-plane translation and rotation errors.

A key feature of MSL-RAPTOR is that it can incorporate temporal information in a probabilistically rigorous manner via the UKFs and their coupling with the front-end. The uncertainty estimates are used to improve the measurements by the front-end and update the pose estimates. SSP does not have these capabilities, leading to many estimates with large errors that result in the apparent asymptotes in Fig. 5.7.

5.6 Conclusion

In this paper, we propose MSL-RAPTOR, an efficient monocular multi-object relative pose tracking algorithm that combines the image processing capabilities of modern neural networks with the robustness advantages of classical probabilistic filters. The front-end extracts class labels and angled bounding boxes, which are used as observations by the back-end's UKFs. The back-end estimates the state and returns measurement uncertainty parameters to the

front-end for methodically determining when to trigger the slower, but more accurate, re-detection. When not detecting, the faster visual tracking method is used. The algorithm runs 3 times faster than the quickest state-of-the-art method, while still out-performing it in terms of precision in onboard robotic perception scenarios. Furthermore, despite not using depth measurements, comparable performance to state-of-the-art results is achieved on the NOCS-REAL275 dataset.

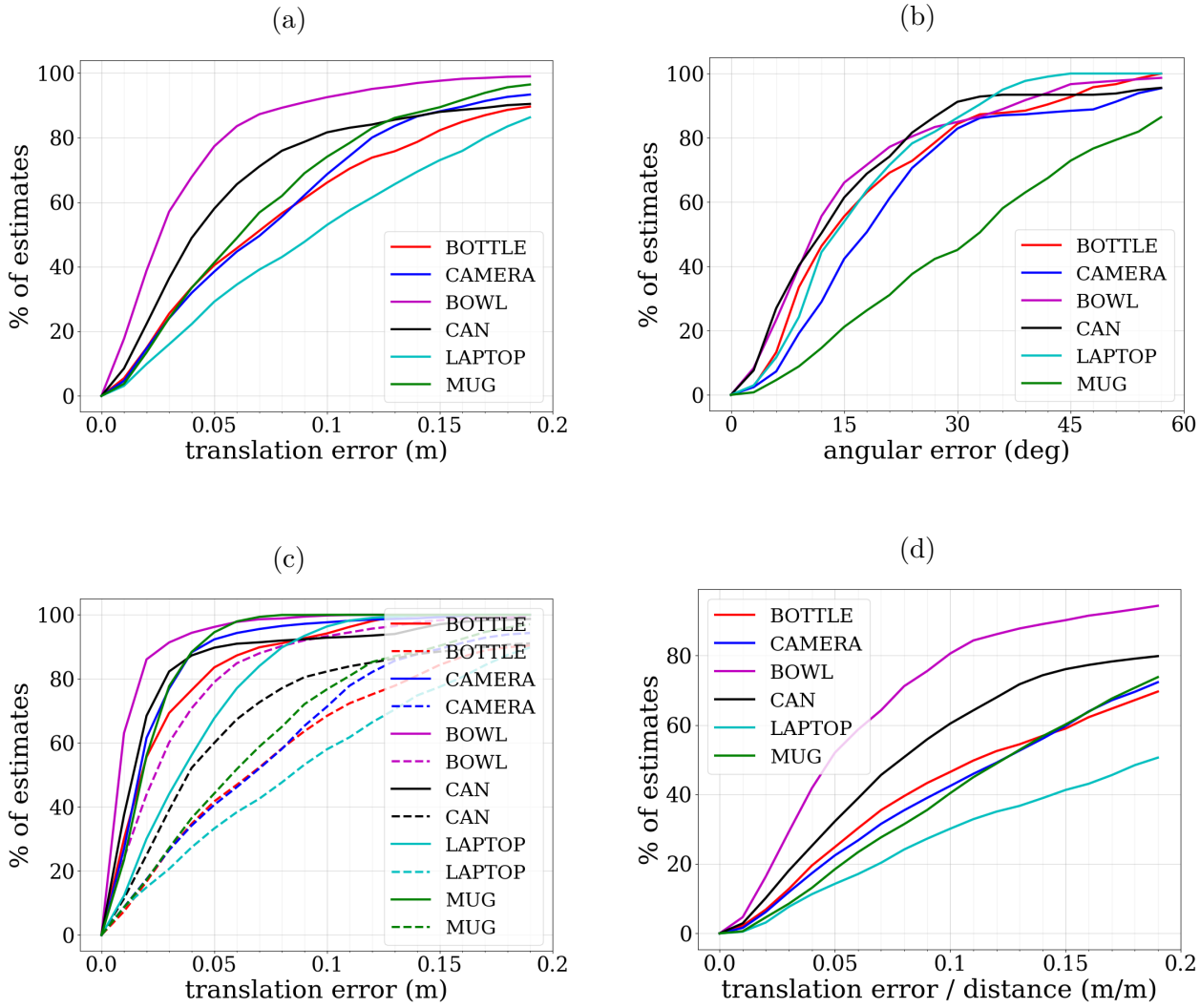

Figure 5.5: Empirical cumulative distribution function on errors:(a) translation only; (b) rotation only; (c) translation separated into in-plane error (solid line) and depth error (dashed line); (d) translation error per distance to obstacle.

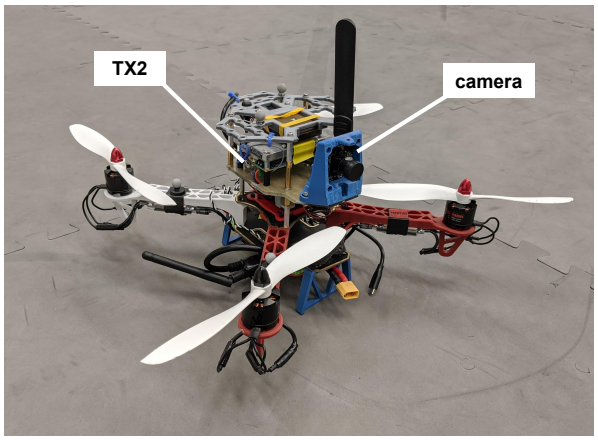

Figure 5.6: Quadrotor used for tracking objects in robotic perception scenarios.

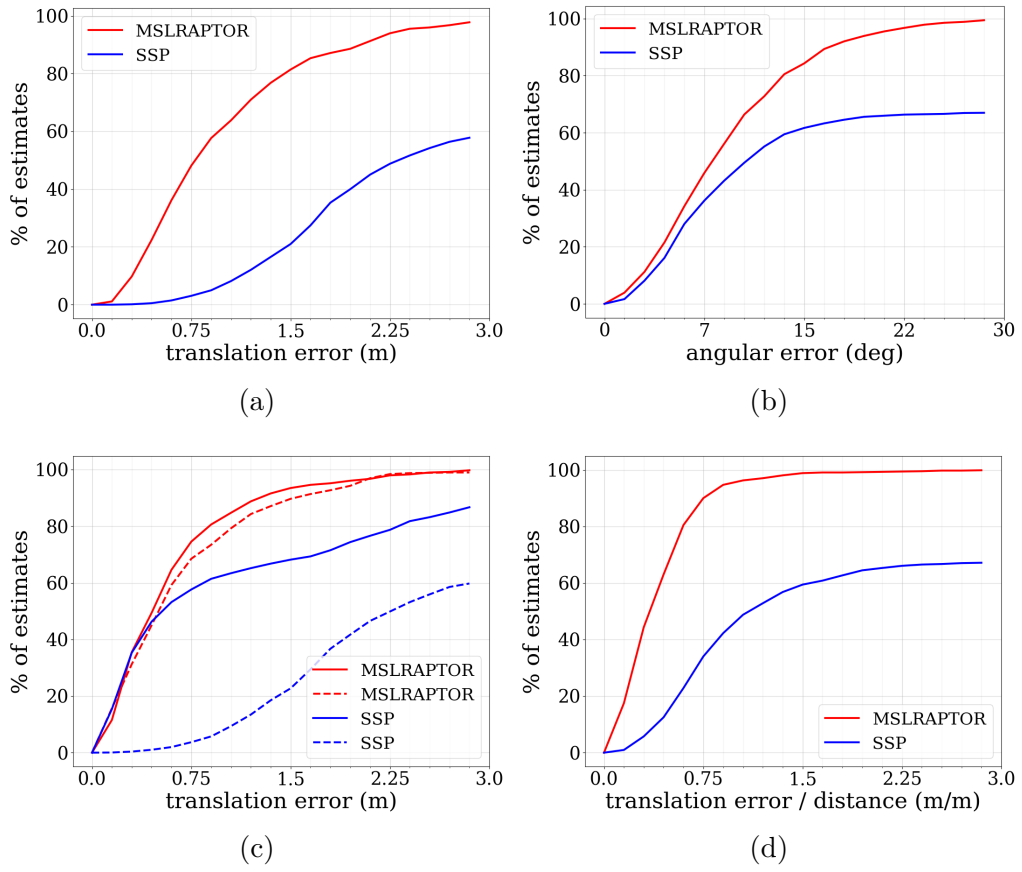

Figure 5.7: Empirical cumulative distribution function on errors: (a) translation only; (b) rotation only; (c) translation in-plane error (solid line) and depth error (dashed line); (d) translation error per distance to obstacle.

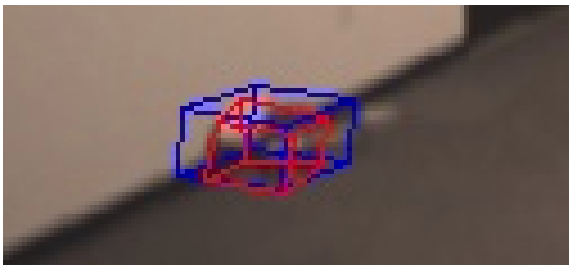

Figure 5.8: SSP's prediction for the projected 3D bounding box (red), and the ground truth bounding box (blue). Note that due to the fast motion of the drone, the blurry image has resulted in a distorted prediction.

CHAPTER 6 GENERAL DISCUSSION

In this chapter, we jointly discuss the works presented in Chapters 4 and 5 and their place in practical multi-robot inter-robot loop closure detections.

CAPRICORN, presented in Chapter 4, proposes a decentralized place recognition mechanism that can serve for indirect inter-robot loop closure detections. On the other-hand, MSL-RAPTOR, presented in Chapter 5 proposes a method for relative pose estimation which can serve for direct inter-robot loop closure detections.

6.1 Hardware requirements

Considering the wide presence of cameras on robots, both methods have been designed to rely on vision sensors. However, CAPRICORN makes use of additional depth information provided by RGB-D cameras to build a constellation, while MSL-RAPTOR only requires monocular images. The constellation presented in CAPRICORN must be 3D, but the sensory input to create it can easily be changed. In fact, the main contribution relies on the use of the representation rather than in how it is built. Instead, making use of motion from sequences of monocular images has been explored to estimate depth data [116], and it is easy to imagine adapting CAPRICORN to a monocular camera sensor if needed. Moreover, MSL-RAPTOR can be used with monocular images to estimate the relative pose of other objects than robots of a team. As such, it could provide positions of objects using only monocular images, generating the information necessary to generate points of the constellation and remove the need for depth data.

A key aspect associated with robots is their mobility, and consequently their need to rely on embedded computing devices. These constraints have been considered in the design of both CAPRICORN and MSL-RAPTOR, and have been tested on-board a flying drone in the latter. The most computationally intensive part is the deep learning-based image processing which serves as a base for both solutions. They rely on pre-trained networks and do not require online re-training, but only inference. Embedded computers such as the Nvidia Jetson TX2 or the Nvidia Jetson Xavier allow to take advantage of GPUs on mobile systems and provide satisfying computation speed for inference when using small enough neural networks. Both approaches are thus fit to be used onboard real robotics systems using vision and embedded computers.

6.2 Contexts of usage

While both CAPRICORN and MSL-RAPTOR serve to estimate relative poses between different robots of a team, they can do so in different contexts. CAPRICORN relies on the fact that the team is fully connected, and robots can all communicate with each other. It can be adapted to other mechanisms with different communication requirements, but will still require data exchanges to recognize places and find loop closures. MSL-RAPTOR, instead, allows each robot to independently obtain such estimates and does not make assumptions about communication possibilities. A limitation associated with direct estimation methods such as MSL-RAPTOR is the fact that robots need to be in line-of-sight of each other to generate such loop closures. Indirect methods such as CAPRICORN can make use of the full history of trajectories of both robots to find places visited by several members of the team, and do not require two robots to be in proximity of each other at any point in time. This gives the potential for many more inter-robot loop closures to be found. In a fully connected robotic team, these methods can be used in a complementary manner: combining a direct estimate to an indirect one will generate more inter-robot loop closure measurements, and improve the final SLAM estimates of the system.

6.3 Adaptability to new scenarios

It is primordial that the proposed methods be adaptable to a variety of robot types and environments for them to be reusable, and useful to the robotics community.

CAPRICORN does not need adjustments when robots of the team change, but must be adapted to the environment it has to work in since it relies on a pre-trained object detector to create the constellation. These are the subject of extensive research, and many datasets are provided with labeled data to train object detectors on a wide range of objects. The only measures to take to reuse CAPRICORN in a new environment are to ensure that a method to segment the images of that environment into semantic entities is available. This means that CAPRICORN could be used outdoors in a forest if given an appropriate way to identify trees. If pre-trained weights for semantic entities of an environment are not available, training appropriate weights would require to obtain labeled data in the form of bounding boxes around semantic elements of interest.

MSL-RAPTOR does not need any adjustments to new environments but instead must be adapted to the robots whose relative pose must be estimated. Similarly to CAPRICORN, the only module in the front-end concerned is the object detector which finds the robot in the images. For common robot types, many images are available online, and it is reasonable

to assume that datasets can be curated to train object detectors on general classes of robots. For unusual looking robots, it might be required to retrain the object detector to adjust to its specific appearance, as was done with the drone in the experiments. This would require to collect data in which the robot is placed in front of many backgrounds and to provide bounding boxes around the robot. Alternatively, the learned object detector could be changed for a hand-made computer vision algorithm making use of the specific visual cues from the robot. In the back-end, an approximation of the shape of the robot is necessary but can be assumed based on the type of robot, and should be known before-hand when deploying a robotic team for multi-robot SLAM. This approximation can be very rough, but the more precise it is, the more accurate the pose estimates will be. Moreover, the UKF must be tuned according to the dynamics of the robots and the noise of the bounding boxes generated by the front-end.

In both cases, while the methods might require adjustments depending on the environment and robots in the team, these are facilitated by the datasets available online and do not require expensive data labeling.

CHAPTER 7 CONCLUSION AND RECOMMENDATIONS

This thesis has presented two vision-based approaches to provide inter-robot loop closure measurements for multi-robot SLAM. First, an indirect approach is proposed in the form of a decentralized place recognition solution which makes use of a compact descriptive semantic representation of the environment. Secondly, a direct approach is presented and proposes a general relative pose tracking solution that can be applied to robots when they appear in the field-of-view of an image. These two approaches complement each other and are suited to physical computation and communication constraints that appear in a multi-robot systems setup.

7.1 Summary of Work

The indirect approach, CAPRICORN, describes places using constellations, sets of 3D points with a semantic label. We present a semantic descriptor based on what classes of objects appear in a scene, as well as a geometric descriptor describing the geometric configuration of the scene. These provide robust and compact descriptions that can be used to recognize places. Additionally, we adapt a decentralized place recognition mechanism to use constellations and show that places can be recognized over a full robotic team in a scalable way.

The direct solution, MSL-RAPTOR, is proposed as a general relative pose estimation tool that can be applied to other robots for inter-robot loop closure detections. It relies on visual tracking to feed 2D bounding box measurements to an unscented Kalman filter which produces pose estimates. Compared to other pose estimation methods, it is easily adaptable to new robots and well suited for on-board computation.

7.2 Limitations

While the proposed approaches reach the research objectives proposed, they present limitations. CAPRICORN is subject to perceptual aliasing as several places in our man-made environments tend to contain similar configuration of objects. For example, desks usually contain similar objects such as a screen, a keyboard, and a mouse, and constellations for different desks could end up resembling each other closely. Moreover, the decentralization mechanism proposed does not balance well the communication and computation loads among robots of the team and assumes full connectivity, which is not always possible.

MSL-RAPTOR suffers from the ambiguity that multiple poses can generate the same 2D

bounding box, which can generate errors in the pose estimates if poorly initialized. Moreover, some objects can exhibit distinctive shapes in their core that could help estimate their pose, but MSL-RAPTOR only makes use of the outer shape of objects. Finally, the level of accuracy of MSL-RAPTOR depends on the quality of the approximation of the tracked object's shape but is usually not as precise as indirect relative pose estimation methods.

7.3 Future Work

These limitations guide possibilities of future works to extend and improve the proposed approaches, and create better tools to perform inter-robot loop closure detection.

A straightforward approach to limit perceptual aliasing with constellations of objects is to rely on more descriptive classes. These could include visual aspects such as color for example which would help discriminate objects of the same class. An important extension for CAPRICORN would be to develop a method to build constellations consistently, in such a way that the point associated with an object remains consistent when generating the constellation from multiple point-of-views. Doing so could make the geometry of constellations representing the same scene consistent enough to not only perform place recognition, but also relative pose estimation directly based on the constellations rather than on visual features. This would reduce greatly the most costly aspect of communication for inter-robot loop closure detections. Interestingly, in addition to finding direct loop closures, MSL-RAPTOR can also estimate the pose of objects in the environment. Hence, using MSL-RAPTOR to provide the information needed to generate a constellation would allow to remove the need for depth data, and could provide better consistency in the position of objects when seen from multiple viewpoints Moreover, MSL-RAPTOR would be able to provide the orientation of objects, for which CAPRICORN could be adjusted to use and become more discriminative and robust to perceptual aliasing.

Constellations also allow for priors to be incorporated, based on our knowledge of the environment, and these could contribute to better decentralization mechanisms. For example, scene categorization could be performed by looking at objects composing a constellation and serve as a filter to check for place recognition matches. Priors could also prevent relying on uncertain semantic entities that may not be static and should not be relied upon for example.

MSL-RAPTOR could also be improved by designing a more complex pose estimate initialization solution. The current initialization relies on strong approximations but is critical to the pose tracking performance. Incorporating other works on pose estimation from a single frame, even if more computationally intense, could allow for better initialization and thus

better pose estimates during tracking. In addition, MSL-RAPTOR could benefit from using more accurate models of the objects to track, as well as a more detailed segmentation of the objects in the images. The angled bounding box that is currently used does not fit well all objects in general, and more flexible shapes in the measurement could allow for better performance on a wider set of objects to track.

Finally, an interesting next step would be to incorporate and test both methods in DOOR-SLAM [7], another work to which contributions were made as part of this Master's program. CAPRICORN and MSL-RAPTOR could serve as additional, complementary inter-robot loop closure mechanisms to the current one used in DOOR-SLAM. This would allow DOOR-SLAM to adapt and combine the front-end inter-robot loop closure methods to the communication abilities of the team. Without communication, MSL-RAPTOR could be used. With limited communication, DOOR-SLAM's current distributed front-end is an appropriate choice. In fully connected teams, CAPRICORN would be most efficient. Experimenting with CAPRICORN and MSL-RAPTOR as part of a full deployable SLAM system in the real-world would also allow for solid proof of their relevance. They would benefit from DOOR-SLAM's robust back-end to avoid failure against perceptual aliasing which can occur with CAPRICORN, or outlier estimates provided by MSL-RAPTOR which can occur due to the observation ambiguity when poorly initialized.

REFERENCES

[1] "The Fourth Industrial Revolution: what it means and how to respond," library Catalog: www.weforum.org. [Online]. Available: https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/

[2] "When Robots Ring the Bell - The New York Times." [Online]. Available: https://www.nytimes.com/2018/11/07/business/robotics-automation-productivity-jobs.html

[3] "The Future of Jobs Report," World Economic Forum, Geneva, Switzerland, Tech. Rep., 2018.

[4] R. Bruemmer, M. G. U. March 10, and 2018, "Driverless cars are coming. Is Montreal ready? | Montreal Gazette," Mar. 2018, library Catalog: montrealgazette.com. [Online]. Available: https://montrealgazette.com/news/local-news/driverless-cars-are-coming-is-montreal-ready

[5] "Six Ways Drones Are Revolutionizing Agriculture - MIT Technology Review." [Online]. Available: https://www.technologyreview.com/s/601935/six-ways-drones-are-revolutionizing-agriculture/

[6] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016, arXiv: 1606.05830.

[7] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656–1663, Apr. 2020.

[8] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: http://link.springer.com/10.1023/B:VISI.0000029664.99615.94

[9] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, vol. 3951, pp. 404–417.

[10] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, vol. 3951, pp. 430–443, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/11744023_34

[11] D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 6314, pp. 778–792, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-642-15561-1_56

[12] Sivic and Zisserman, "Video Google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*. Nice, France: IEEE, 2003, pp. 1470–1477 vol.2.

[13] Fei-Fei Li and P. Perona, "A Bayesian Hierarchical Model for Learning Natural Scene Categories," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. San Diego, CA, USA: IEEE, 2005, pp. 524–531.

[14] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[15] A. Oliva and A. Torralba, "Chapter 2 Building the gist of a scene: the role of global image features in recognition," in *Progress in Brain Research*. Elsevier, 2006, vol. 155, pp. 23–36.

[16] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, Jun. 2005, pp. 886–893 vol. 1, iSSN: 1063-6919.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Dec. 2015. [Online]. Available: https://doi.org/10.1007/s11263-015-0816-y

[19] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition," in *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. Columbus, OH, USA: IEEE, Jun. 2014, pp. 512–519. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6910029

[20] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, Jun. 2014, pp. 1717–1724. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909618

[21] N. Suenderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of ConvNet features for place recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 4297–4304.

[22] Z. Chen, O. Lam, A. Jacobson, and M. Milford, "Convolutional neural network-based place recognition," pp. 1–8, 2014. [Online]. Available: https://eprints.qut.edu.au/79662/

[23] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[24] D. Tardioli, E. Montijano, and A. R. Mosteo, "Visual data association in narrow-bandwidth networks," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Hamburg, Germany: IEEE, Sep. 2015, pp. 2572–2577. [Online]. Available: http://ieeexplore.ieee.org/document/7353727/

[25] T. Cieslewski and D. Scaramuzza, "Efficient Decentralized Visual Place Recognition Using a Distributed Inverted Index," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 640–647, Apr. 2017.

[26] ——, "Efficient decentralized visual place recognition from full-image descriptors," in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. Los Angeles, CA: IEEE, Dec. 2017, pp. 78–82.

[27] Y. Tian, K. Khosoussi, M. Giamou, J. P. How, and J. Kelly, "Near-Optimal Budgeted Data Exchange for Distributed Loop Closure Detection," *arXiv:1806.00188 [cs]*, Jun. 2018, arXiv: 1806.00188. [Online]. Available: http://arxiv.org/abs/1806.00188

[28] Y. Tian, K. Khosoussi, and J. P. How, "A Resource-Aware Approach to Collaborative Loop Closure Detection with Provable Performance Guarantees," *arXiv:1907.04904 [cs]*, Jul. 2019, arXiv: 1907.04904. [Online]. Available: http://arxiv.org/abs/1907.04904

[29] M. Giamou, K. Khosoussi, and J. P. How, "Talk Resource-Efficiently to Me: Optimal Communication Planning for Distributed Loop Closure Detection," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD: IEEE, May 2018, pp. 3841–3848. [Online]. Available: https://ieeexplore.ieee.org/document/8460783/

[30] X. Zhou and S. Roumeliotis, "Multi-robot SLAM with Unknown Initial Correspondence: The Robot Rendezvous Case," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Beijing, China: IEEE, Oct. 2006, pp. 1785–1792. [Online]. Available: http://ieeexplore.ieee.org/document/4058636/

[31] S. Roumeliotis and G. Bekey, "Distributed multirobot localization," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, Oct. 2002. [Online]. Available: http://ieeexplore.ieee.org/document/1067998/

[32] L. Paull, G. Huang, M. Seto, and J. J. Leonard, "Communication-constrained multi-AUV cooperative SLAM," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015, pp. 509–516. [Online]. Available: http://ieeexplore.ieee.org/document/7139227/

[33] Been Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller, "Multiple relative pose graphs for robust cooperative mapping," in *2010 IEEE International Conference on Robotics and Automation*. Anchorage, AK: IEEE, May 2010, pp. 3185–3192. [Online]. Available: http://ieeexplore.ieee.org/document/5509154/

[34] C. Sahin, G. Garcia-Hernando, J. Sock, and T.-K. Kim, "A review on object pose recovery: from 3d bounding box detectors to full 6d pose estimators," *arXiv preprint arXiv:2001.10609*, 2020.

[35] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing Images Using the Hausdorff Distance," *Ieee Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, pp. 850–863, 1993.

[36] C. Olson and D. Huttenlocher, "Automatic target recognition by matching oriented edge pixels," *IEEE Transactions on Image Processing*, vol. 6, no. 1, pp. 103–113, Jan. 1997, conference Name: IEEE Transactions on Image Processing.

[37] C. Steger, "Similarity Measures for Occlusion, Clutter, and Illumination Invariant Object Recognition," in *Pattern Recognition*, G. Goos, J. Hartmanis, J. van Leeuwen, B. Radig, and S. Florczyk, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, vol. 2191, pp. 148–154, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/3-540-45404-7_20

[38] ——, "Occlusion, Clutter, and Illumination Invariant Object Recognition," p. 6.

[39] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient Response Maps for Real-Time Detection of Textureless Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 876–888, May 2012. [Online]. Available: http://ieeexplore.ieee.org/document/6042881/

[40] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *2011 International Conference on Computer Vision*. Barcelona, Spain: IEEE, Nov. 2011, pp. 858–865. [Online]. Available: http://ieeexplore.ieee.org/document/6126326/

[41] V. Lepetit, J. Pilet, and P. Fua, "Point matching as a classification problem for fast and robust object pose estimation," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2. Washington, DC, USA: IEEE, 2004, pp. 244–250. [Online]. Available: http://ieeexplore.ieee.org/document/1315170/

[42] Y. Amit and D. Geman, "Shape Quantization and Recognition with Randomized Trees," *Neural Computation*, vol. 9, no. 7, pp. 1545–1588, Oct. 1997. [Online]. Available: http://www.mitpressjournals.org/doi/10.1162/neco.1997.9.7.1545

[43] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1465–1479, Sep. 2006. [Online]. Available: http://ieeexplore.ieee.org/document/1661548/

[44] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1521–1529.

[45] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes," in *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation, Jun. 2018. [Online]. Available: http://www.roboticsproceedings.org/rss14/p19.pdf

[46] T.-T. Do, T. Pham, M. Cai, and I. Reid, "Real-time monocular object instance 6d pose estimation," in *British Machine Vision Conference (BMVC)*, vol. 1, no. 2, 2018, p. 6.

[47] M. Rad and V. Lepetit, "Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3828–3836.

[48] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 292–301.

[49] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[50] S. Saeedi, M. Trentini, M. Seto, and H. Li, "Multiple-Robot Simultaneous Localization and Mapping: A Review: Multiple-Robot Simultaneous Localization and Mapping," *Journal of Field Robotics*, vol. 33, no. 1, pp. 3–46, Jan. 2016.

[51] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, Dec. 2011.

[52] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular SLAM with multiple Micro Aerial Vehicles," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Tokyo: IEEE, Nov. 2013, pp. 3962–3970.

[53] L. Riazuelo, J. Civera, and J. Montiel, "C2tam: A Cloud framework for cooperative tracking and mapping," *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 401–413, Apr. 2014.

[54] J. G. Morrison, D. Gálvez-López, and G. Sibley, "MOARSLAM: Multiple Operator Augmented RSLAM," in *Distributed Autonomous Robotic Systems*, N.-Y. Chong and Y.-J. Cho, Eds. Tokyo: Springer Japan, 2016, vol. 112, pp. 119–132.

[55] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-Efficient Decentralized Visual SLAM," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 2466–2473.

[56] R. Finman, T. Whelan, L. Paull, and J. Leonard, "Physical Words for Place Recognition in RGB-D Maps," in *International Conference on Robotics and Automation Workshop on Place Recognition in Changing Environments*, 2014.

[57] K. M. Frey, T. J. Steiner, and J. P. How, "Efficient constellation-based map-merging for semantic slam," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 1302–1308.

[58] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, Z. Liu, H. I. Christensen, and F. Dellaert, "Multi Robot Object-Based SLAM," in *2016 International Symposium on Experimental Robotics*, D. Kulić, Y. Nakamura, O. Khatib, and G. Venture, Eds. Cham: Springer International Publishing, 2017, vol. 1, pp. 729–741.

[59] B. Ramtoula, R. De Azambuja, and G. Beltrame, "Data-efficient decentralized place recognition with 3d constellations of objects [extended abstract]," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. New Brunswick, NJ: IEEE, Aug. 2019.

[60] S. Lowry, N. Suenderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual Place Recognition: A Survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, Feb. 2016.

[61] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2. San Francisco, CA, USA: IEEE, 2000, pp. 1023–1029.

[62] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece: IEEE, 1999, pp. 1150–1157 vol.2.

[63] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 779–788.

[64] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, Jun. 2015, pp. 3431–3440.

[65] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[66] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *Computer Vision – ECCV 2016*. Cham: Springer International Publishing, 2016, vol. 9905, pp. 21–37.

[67] Z. Chen, F. Maffra, I. Sa, and M. Chli, "Only look once, mining distinctive landmarks from ConvNet for visual place recognition," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vancouver, BC: IEEE, Sep. 2017, pp. 9–16.

[68] S. Garg, N. Suenderhauf, and M. Milford, "Don't Look Back: Robustifying Place Categorization for Viewpoint- and Condition-Invariant Place Recognition," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 3645–3652.

[69] S. Garg, N. Suenderhauf, and M. Milfort, "Lost? appearance-invariant place recognition for opposite viewpoints using visual semantics," *Proceedings of Robotics: Science and Systems XIV*, 2018.

[70] N. Suenderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford, "Place Recognition with ConvNet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free," in *Robotics: Science and Systems XI*. Robotics: Science and Systems Foundation, Jul. 2015.

[71] T. Naseer, G. L. Oliveira, T. Brox, and W. Burgard, "Semantics-aware visual localization under challenging perceptual conditions," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. Singapore, Singapore: IEEE, May 2017, pp. 2614–2620.

[72] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 1352–1359.

[73] S. Choudhary, A. J. Trevor, H. I. Christensen, and F. Dellaert, "SLAM with object discovery, modeling and mapping," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 1018–1025.

[74] J. Mccormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger, "Fusion++: Volumetric Object-Level SLAM," in *2018 International Conference on 3D Vision (3DV)*, Sep. 2018, pp. 32–41.

[75] L. Nicholson, M. Milford, and N. Sünderhauf, "QuadricSLAM: Dual Quadrics From Object Detections as Landmarks in Object-Oriented SLAM," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, Jan. 2019.

[76] M. Hosseinzadeh, Y. Latif, T. Pham, N. Suenderhauf, and I. Reid, "Structure Aware SLAM using Quadrics and Planes," *arXiv:1804.09111 [cs]*, Apr. 2018, arXiv: 1804.09111.

[77] M. Hosseinzadeh, K. Li, Y. Latif, and I. Reid, "Real-Time Monocular Object-Model Aware Sparse SLAM," *arXiv:1809.09149 [cs]*, Sep. 2018, arXiv: 1809.09149.

[78] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard, "Efficient incremental map segmentation in dense RGB-D maps," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong, China: IEEE, May 2014, pp. 5488–5494.

[79] A. Gawel, C. D. Don, R. Siegwart, J. Nieto, and C. Cadena, "X-View: Graph-Based Semantic Multi-View Localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1687–1694, Jul. 2018.

[80] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 5220–5227.

[81] E. Montijano, R. Aragues, and C. Sagüés, "Distributed Data Association in Robotic Networks With Cameras and Limited Communications," *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1408–1423, Dec. 2013.

[82] A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli, "The Sensor-based Random Graph Method for Cooperative Robot Exploration," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 163–175, Apr. 2009.

[83] R. Arandjelović and A. Zisserman, "Visual vocabulary with a semantic twist," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 178–195.

[84] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767 [cs]*, Apr. 2018, arXiv: 1804.02767.

[85] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vilamoura-Algarve, Portugal: IEEE, Oct. 2012, pp. 573–580.

[86] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common Objects in Context," *arXiv:1405.0312 [cs]*, May 2014, arXiv: 1405.0312.

[87] P. Lajoie, S. Hu, G. Beltrame, and L. Carlone, "Modeling Perceptual Aliasing in SLAM via Discrete–Continuous Graphical Models," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1232–1239, Apr. 2019.

[88] J. Wu, H. I. Christensen, and J. M. Rehg, "Visual Place Categorization: Problem, Dataset, and Algorithm," in *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, ser. IROS'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 4763–4770, event-place: St. Louis, MO, USA.

[89] R. Spica, D. Falanga, E. Cristofalo, E. Montijano, D. Scaramuzza, and M. Schwager, "A real-time game theoretic planner for autonomous two-player drone racing," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[90] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects." in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 87. PMLR, 2018, pp. 306–316.

[91] M. Wang, Z. Wang, S. Paudel, and M. Schwager, "Safe Distributed Lane Change Maneuvers for Multiple Autonomous Vehicles Using Buffered Input Cells," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, May 2018.

[92] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, March 2000.

[93] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.

[94] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2642–2651.

[95] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.

[96] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[97] T. Nguyen, S. S. Shivakumar, I. D. Miller, J. Keller, E. S. Lee, A. Zhou, T. Özaslan, G. Loianno, J. H. Harwood, J. Wozencraft, C. J. Taylor, and V. Kumar, "Mavnet: An effective semantic segmentation micro-network for mav-based tasks," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3908–3915, Oct 2019.

[98] C. Xie, Y. Xiang, Z. Harchaoui, and D. Fox, "Object discovery in videos as foreground motion clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9994–10 003.

[99] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*. Springer, 2016, pp. 850–865.

[100] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 1328–1338.

[101] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "SiamRPN++: Evolution of Siamese Visual Tracking With Very Deep Networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, Jun. 2019, pp. 4277–4286.

[102] A. Sauer, E. Aljalbout, and S. Haddadin, "Tracking holistic object representations," in *British Machine Vision Conference (BMVC)*, 2019.

[103] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[104] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[105] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "Poserbpf: A rao-blackwellized particle filter for6d object pose estimation," in *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.

[106] C. Wang, R. Martín-Martín, D. Xu, J. Lv, C. Lu, L. Fei-Fei, S. Savarese, and Y. Zhu, "6-pack: Category-level 6d pose tracker with anchor-based keypoints," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, May 2020.

[107] T. Schmidt, R. Newcombe, and D. Fox, "Dart: Dense articulated real-time tracking," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[108] E. Kraft, "A quaternion-based unscented kalman filter for orientation tracking," in *Proceedings of the Sixth International Conference of Information Fusion*, vol. 1, 2003, pp. 47–54.

[109] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[110] G. Jocher, guigarfr, perry0418, Ttayu, J. Veitch-Michaelis, G. Bianconi, F. Baltacı, D. Suess, WannaSeaU, and IlyaOvodov, "ultralytics/yolov3: Rectangular Inference, Conv2d + Batchnorm2d Layer Fusion," Apr. 2019.

[111] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin, "A novel performance evaluation methodology for single-target trackers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2137–2155, Nov 2016.

[112] J. Issac, M. Wuthrich, C. G. Cifuentes, J. Bohg, S. Trimpe, and S. Schaal, "Depth-based object tracking using a robust gaussian filter," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016.

[113] M. Wuthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic object tracking using a range camera," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013.

[114] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv preprint arXiv:1801.09847*, 2018.

[115] S. Suwajanakorn, N. Snavely, J. J. Tompson, and M. Norouzi, "Discovery of latent 3d keypoints via end-to-end geometric reasoning," in *Advances in Neural Information Processing Systems*, 2018, pp. 2059–2070.

[116] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT: IEEE, Jun. 2018, pp. 5667–5675. [Online]. Available: https://ieeexplore.ieee.org/document/8578692/