UNIVERSITAT POLYTÈCNICA DE CATALUNYA

Doctoral Program:

AUTOMATICS ROBOTICS AND VISION

Research Plan:

Unified long-term 3D simultaneous localization and mapping for service robots

Jérémie Deray

Advisors: Joan Solà Ortega, Juan Andrade-Cetto
Company mentor: Luca Marchionni

Institut de Robòtica i Infomatica Industrial
IRI-UPC Barcelona, Spain

PAL Robotics, Barcelona, Spain

· July 24, 2017 ·

Contents

Acronyms

SLAM Simultaneous Localization and Mapping

V-SLAM Visual-SLAM

BoW Bag-of-Words

LRF Laser Range Finder

DoF Degree of Freedom

RGB-D Red-Green-Blue–Depth camera

IMU Inertial Measurement Unit

UAV Unmanned Aerial Vehicule

WOLF Windowed Localization Frames

HMM Hidden Markov Model

RANSAC Random Sample Consensus

GPA Generalized Procruste Analysis

NLS Nonlinear Least Square

ICP Iterative Closest Point

ICL Iterative Closest Line

EKF Extended Kalman Filter

EIF Extended Information Filter

DBN Dynamic Bayesian Networks

Chapter 1

Introduction

The thesis disclosed here is framed in the *Industrial PhD program* of the *Generalitat de Catalunya*. The work is then developed in a collaborative framework between the *Institut de Robòtica i Informàtica Industrial* from the *Universitat Politècnica de Catalunya* (*IRI CSIC-UPC*) and the company PAL Robotics. The goals of the research plan are aligned with the mid-term requirements of the company.

1.1 Overview

Since a few years we see an acceleration in the development and spreading of mobile robots evolving alongside humans both in public and private spaces. Automation has began decades ago in industries such as the automotive industry with *in-situ* robots (e.g. robotics arms). Despite a rapid growth due to the extremely structured environment of factories and warehouses limiting the uncertainty of the environment to a limited set of well established rules, integration happened almost exclusively by modifying the environment (metal cage, painted railway etc). Robots were strictly separated from workers in order to prevent any dramatic accident. The current novelty lies in the mobility aspect of the robots. With the current advances in compliance, mapping, perception, together with the interest of the public, robots are slowly but surely getting out of their metal cage.

This evolution obviously raises many different challenges but also legal and ethical concerns, we focus in this work on one of the fundamental problems in Robotics: Navigation.

1.2 Problem Statement

Although service robots begin to appear in semi-structured public spaces such as shopping malls or museums, their presence is still unusual. Most of the time a robot appears to the public, it is quickly surrounded by a crowd of people being curious leading to a partial or complete occlusion of its sensors. In this condition the robot is nearly blind and naive mapping methods fail to localize the robot. Such situation, as depicted in Figure 1.1, highlights the difficulties of dynamic environments.

Before being able to cope with the fairly unstructured and dynamic environment that homes are, they are going to be massively deployed in semi-structured public spaces (hospitals, retail stores, malls, museums ...). To do so, their reliability together with their capacity to sense and adapt to dynamic scenes

must be improved both for people-safety and long-term unwatched deployment. To adapt to dynamic environment, robots must be able to recognize places despite static and/or dynamic changes (change in room furniture, people passing by, seasonal change ...). Moreover, since spaces as those aforementioned are commonly very large, robots must then be able to create, maintain and update a fairly large map.

Another type of difficulty encountered leading to erroneous localization is related to the nature of the environment and the technology of the sensors used to apprehend it. Certain materials can absorb, refract or reflect the beam of a Laser Range Finder (LRF) (i.e. black paint, windows, mirrors), whereas entire room can lack of visual features (i.e. uniformly painted wall) making cameras almost useless for navigation purpose. This motivates the use of several different types of sensors at a time.

Figure 1.1: The REEM Robot surrounded at a fair.

1.3 Objectives

The main objective of the thesis is to investigate novel methods for multi-modal Simultaneous Localization and Mapping (SLAM). These methods must ensure the robustness over time of the overall navigation framework as they will be implemented on commercially available service robots which are deployed in real environments.

Most modern robots feature several different sensors allowing for a redundancy in the sources of displacement sensing (odometry) together with a diversity in the perceived environment's features. We aim at exploiting both these aspects in order to tend toward a robust continuous SLAM system.

Moreover, whereas SLAM research has moved toward richer sensor such as cameras or 3D LRF, many robots still use 2D LRF sensors for navigation, especially mobile bases for industrial applications. A consequence of the shift of interest to different sensors is that current solutions to planar LRF-based SLAM do not benefit from the latest advances and concepts developed for their visual counterparts. We then aim at developing algorithms that adapts some of the latest key improvements to planar LRF-based SLAM.

The main objectives are as follows:

- Develop a local optimization scheme for LRF-based SLAM which aims at refining the estimated local odometry together with the estimated local map.

- Integrate the redundancy of mobile-base odometry sources in order to bolster the SLAM framework against misestimation, algorithm divergence or critical failure.

- Improve the recognition of places using multi-sensors data to diminish the false positive recognition rate. Such recognition can be used for both re-localization and/or loop-closure.

1.4 Expected Contribution

In order to meet the above-mentioned objectives, at first a comprehensive in depth overview of the state-of-the-art must be studied and compiled, to contribute with novel solutions.

The aim is at bolstering a continuous SLAM algorithm allowing robots to constantly adapt to dynamic environments during long deployment.

To that end, we expect to provide means for exploiting robots sensors complementarity and redundancy both for improving and make robust the estimation of the trajectory; and the recognition of past visited places. Key concepts and algorithms drawing inspiration from those encountered in Visual-SLAM (V-SLAM) are going to be formalized, developed and evaluated for LRF-based SLAM.

1.5 Resources

PAL Robotics

Founded in 2004, PAL Robotics is a worldwide leading company in biped humanoid and service robots based in Barcelona. Aiming at enhancing people's quality of life, PAL's team is composed of passionate engineers that creates research platforms as well as service robots for tasks such as inventory making.

Since 2004 and the release of first version of the REEM-A robot, PAL Robotics developed several robotic platforms including:

- REEM-A – officially released in 2005 it won the following year the walking challenge of the RoboCup.
- REEM-B – the strongest robot of its time since it could carry a load about 20% of its own weight.
- REEM-H1 – the first wheel-based mobile humanoid robot of the company.
- REEM – the second wheel-based mobile humanoid robot and one of the current platforms.
- REEM-C – a human-size biped humanoid robot.
- Pmb2 – a mobile base platform developed targeting both industry and research needs.
- TiaGo – Take it & Go. A mobile manipulator that adapts to research needs.
- StockBot – a service robot that ease inventory making by means of RFID technology.
- Talos – a human-size biped humanoid robot and one of the most advanced platform in the world. It is the last born robot of PAL family.

The software developed during the thesis will mostly be tested and used both on the TiaGo and StockBot. Moreover the company provides a workstation connected to their internal network and a complete access to their software infrastructure.

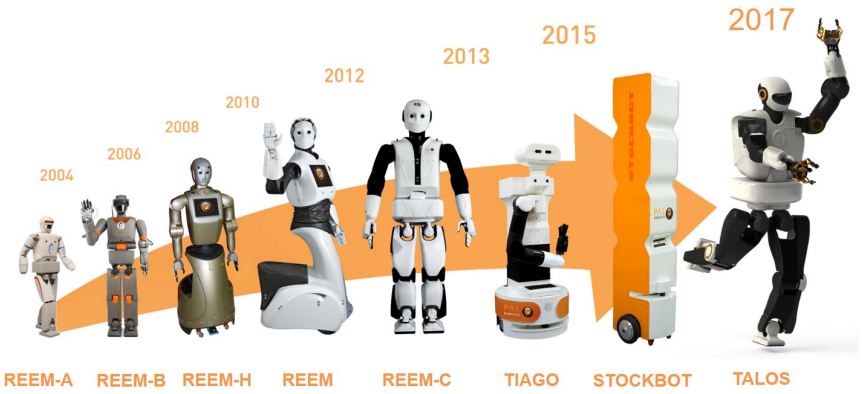


Figure 1.2: PAL's robot family.

Institut de Robòtica i Infomatica Industrial

The university provides access to their under-development SLAM framework. This library, named Windowed Localization Frames (WOLF) aims at solving localization problems in mobile robotics, such as SLAM, map-based localization, or visual odometry. It is mainly a structure for having the data accessible and organized, plus some functionality for managing this data

1.6 Plan Outline

The remainder of the report is structured as follows:

- Chapter 2 gives an overview of the state-of-the-art.
- Chapter 3 describes the preliminary work and characterizes a work-plan.
- Chapter 4 draws a time frame planning.

Chapter 2

State-of-the-Art

2.1 The SLAM Problem

For a mobile robot, SLAM is the process of concurrently building a representation of the environment, a map, and estimating its own localization in this map. The acronym actually encompasses three main complementary problematics:

- *Mapping* - the problem of the representation of the environment as the robot perceives it from its sensors readings.
- *Localization* - the problem of the localization of the robot within the aforementioned representation of the environment.
- *Planning* - the problem of finding a feasible trajectory between at least two configurations in a map.

These complementary problems are extensively active research areas and are fundamental in the sense that many high level tasks depend on them. How could a robot bring us a drink, clean a room or be a guide in a museum if it does not know its environment, what is its state in this environment nor how to move in it ?

Each one of the aforementioned problems can be sub-divided further into specific sub-problems such as the distinction between Localization and Re-Localization.

Addressing both Mapping and Localization together leads to the so called SLAM algorithm.

Tremendous efforts and progress characterized the past few years of the SLAM community, especially in its branch employing cameras, also called V-SLAM. Nowadays state-of-the-art algorithms are able to accurately localize a robot online and produce rich representation of the environment, by means of a textured point cloud for instance. Despite those impressive results and an inside joke stating that SLAM is solved, the overall SLAM framework is still very challenged by the possible combinations of:

- Robots
  - dynamics : mobile-base, biped, Unmanned Aerial Vehicle, Autonomous Underwater Vehicle ...

  - available sensors : rotary/linear encoders, RGB/D/event-camera, LRF, Inertial Measurement Unit (IMU), radars ...
  - computation resources : one/many cores, cloud

- *Environments*
  - indoor : warehouse, museum, private house ...
  - outdoor : city, wild, underwater, space ...

- Task driven specifications
  - precision of the localization and/or map
  - size of the map
  - communication resources

But also by the *Curse of parameters*. [1]

SLAM is a family of algorithms which encompasses different approaches (e.g. Filtering-based, Optimization-based) and as many different sub-problems as aforementioned scenarios.

They are usually presented as composed by two main components, a *front-end* and a *back-end*. The front-end manages the sensors raw data, extracts, interprets and organizes information in order to build a mathematical representation of the problem which can then be solved by the back-end.

Previous to detailing further, we operate a slight distinction of this representation to propose one that better fits the reality of the implementation of SLAM algorithms. Indeed, the front-end is composed of two main sub-modules which operate at different pace. The first sub-module aims at tracking the robot current state in the concurrently built map representation, hence it must be able to operate at sensor-frame performing short-term data association. The second module on the other hand performs long-term data association, trying to recognize places visited by the robot in the past history of the environment exploration. This operation is usually time and computation expensive therefore must not prevent the tracking from operating in real-time. It can be summarized as follows:

1. *Core module* - building the actual estimation problem and eventually solving it.
2. *Odometry module* - tracking the sensor/robot motion and selecting information to be added to the overall problem.
3. *Loop-closure/re-localization module* - detecting loop closures and re-localizing the sensor/robot.

Most modern SLAM algorithms such as [1, 2] rely on the parallelism of these three modules.

[1] Analogous to the *Curse of dimensionality*, it encompasses the problems of having highly parametrizable algorithms leading to a necessary expert fine tuning for a particular use, environment, scenario etc.

2.2 SLAM Formulation

The SLAM problem has known several formulations and solutions over the years. In this section are given references to some of these formulations and further details the optimization-based formulation which has become the *de-facto* standard SLAM formulation. The reader can find exhaustive and historical reviews of SLAM in [3, 4, 5, 6, 7].

Notation

The SLAM problem is best formulated in terms of probabilities, thereafter the following notation are used:
at given time t,

- x_t - A state vector representing the robot position and orientation - e.g. $x_t = [x, y, \theta] \in \mathbb{R}^2$. It may include extra variables such as the robot velocity.
  $X_T = x_0, x_1, ..., x_T$ The history of robot poses for $t \in [T_0, T]$.
- u_t - A control signal executed at $t-1$ inducing the robot motion to x_t
  $U_T = u_0, u_1, ..., u_T$ The history of input commands.
- l_n - A state vector representing a landmark position and orientation.
  $L = l_0, l_1, ..., l_N$ The set of all landmarks.
  $L_t = l_{t0}, l_{t1}, ..., l_{tn}$ The set of all landmarks observed at time t.
- $z_{t,n}$ - A measurement of landmark l_n at x_t.
  $Z_M = z_{0,N}, z_{1,N}, ..., z_{M,N}$ The set of all observations.

The robot state vector update is a Markov process depending only on the robot previous robot state and the input control command:

$$P(x_t|x_{t-1}, u_t) \Leftrightarrow x_t = f(x_{t-1}, u_t, v_t), \qquad v_k \sim \mathcal{N}(0, \Sigma_v) \quad . \tag{2.1}$$

where f is usually non-linear and models the robot kinematics and v_t is a perturbation considered Gaussian with zero-mean and covariance Σ_v.

The observation model describe the probability of making an observation z_t knowing the robot and landmarks poses:

$$P(z_t|x_t, L) \Leftrightarrow z_t = h(x_t, L) + w_t, \qquad w_k \sim \mathcal{N}(0, \Sigma_h) \quad . \tag{2.2}$$

where h is usually non-linear and models the geometry of the observation and w_t an additive noise considered Gaussian with zero-mean and covariance Σ_h.

The complete probabilistic SLAM model, that is, at time t, the joint posterior density of the landmarks and the robot pose given the history of input controls commands and the observations is then:

$$P(X_T, L|Z_t, U_T, x_0) \ . \tag{2.3}$$

This formulation, exemplified in Fig 2.1, is known as *Full* SLAM as it estimates the whole history of the robot and landmarks poses as opposed to early solution that only kept few landmarks and the current

robot poses [8]. In the case one estimates only the robot poses history by marginalizing the landmarks, such formulation is know as *Pose* SLAM [9, 10, 11, 12].

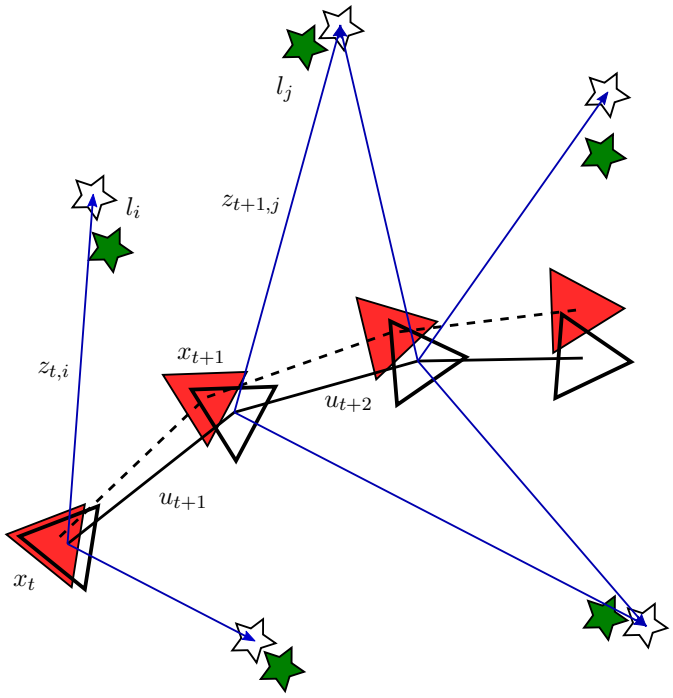


Figure 2.1: The SLAM structure. The hollow figures correspond to the truth whereas filled correspond to the estimates.

2.2.1 Core

During what [7] refers to as *the classical age*, early solution to SLAM used filtering-based algorithms to solve the problem such as Extended Kalman Filter (EKF) [13, 14, 15], particle filters [16, 17] or later on Extended Information Filter (EIF) methods were proposed [9, 12].
Nowadays, the *de-facto* standard SLAM formulations is known as *Graph*-based SLAM.

Graph-based SLAM

The SLAM problem may also be solved through non-linear sparse optimization. In this case it is expressed graphically, the robot and landmarks locations are nodes of a graph, tied together by edges representing their relative relationship - a motion (eq. 2.1) or an observation (eq. 2.2). Such representation is depicted in Figure 2.2.

Factor Graph As mentioned in the previous paragraph, the SLAM-graph is only constituted of two types of nodes: state nodes connected to a small subset of other state nodes through constraint nodes. Such bipartite-graph representation, as depicted in Fig. 2.2 is called the *Factor* graph.

Without further derivation and recalling that the noises in Eq 2.1 and Eq. 2.2 are Gaussian noises,

the SLAM problem eq. 2.3 can be written in a quadratic form,

$$\log P(X_T, L|Z_t, U_T) = \underbrace{\sum_t [x_t - f(x_{t-1}, u_t)]^T \ \Omega_f [x_t - f(x_{t-1}, u_t)]}_{\text{motions}} \ + \\ \underbrace{\sum_t [z_t - h(x_t, L_t)]^T \Omega_h [z_t - h(x_t, L_t)]}_{\text{observations}} \ + \ const \ . \tag{2.4}$$

From the probabilities in Eq 2.1 and Eq. 2.2 the following factors Φ are derived:

$$\begin{aligned} \Phi_t = P(x_t|x_{t-1}, u_t) \quad &\propto \exp\left(-\frac{1}{2}[x_t - f(x_{t-1}, u_t)]^T \Omega_f [x_t - f(x_{t-1}, u_t)]\right) \ . \\ \Phi_n = P(z_t|x_t, L_t) \quad &\propto \exp\left(-\frac{1}{2}[z_t - h(x_t, L_t)]^T \Omega_h [z_t - h(x_t, L_t)]\right) \ . \end{aligned} \tag{2.5}$$

where $\Omega_f = \Sigma_f^{-1}$ and $\Omega_h = \Sigma_h^{-1}$ are the information matrices of the observed data. Eq. 2.5 leads to a unique form of the error formulation:

$$\begin{aligned} e_k(x_{t-1}, x_t) &= f(x_{t-1}, u_t) - x_t \ . \\ e_k(x_t, l_n) &= h(x_t, l_n) - z_n \ . \end{aligned} \tag{2.6}$$

$$\Phi_k = \exp(-0.5 \ e_k^T \Omega_k e_k) \ . \tag{2.7}$$

as exemplified in the Fig. 2.2-*Right*. Finally, the SLAM problem is reduces to solving the equation:

$$x^* = \underset{x}{\operatorname{argmin}} \sum_{k=1}^{K} e_k(x_i, x_j)^T \Omega_k e(x_i, x_j) \ . \tag{2.8}$$

where the summed terms are of the form of the Mahalanobis distance.

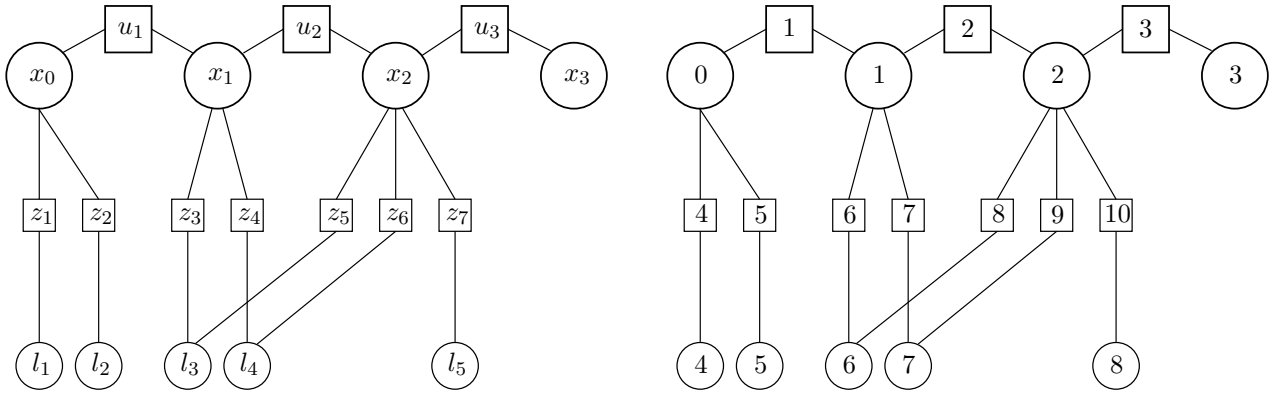


Figure 2.2: A factor graph representation of the SLAM structure of Figure. 2.1. *Left*: Nodes representing known data have been replaced by factors (*squares*) which depends on the unknown variables or states (*circles*). *Right*: The same graph. Unknown states are labeled with a single index $i \in [0, 7]$ and factors are labeled with a single $k \in [1, 10]$.

Solving the SLAM problem As mentioned in sec. 2.2.1, the SLAM problem can be solved by means of iterative non-linear optimization. Rather than detailing the optimization algorithms, which is beyond the scope of this report, here are provided some references shall the reader requires further insights.

Since the SLAM's support matrix is sparse, the methods employed in the literature to solve the problem aims at taking benefits for the sparsity of the problem. The first approach is based on a sparse QR factorization as in [18] and was further developed to an incremental versions in [19, 20]. The same objectives can be fulfilled with the Cholesky factorization [21, 22], reaching similar levels of performance.

2.2.2 Odometry

A key issue in robotics navigation is the ability of the robot to estimate its current pose in an unknown environment, the so called self-localization. The odometry module, generically expressed by Equation 2.1, aims at providing an estimate of the robot state at a given time with respect to the previous one. It may estimates the robot displacement from an input control command and/or from its sensors readings.

Additionally to the integration of the motion, one can also integrate the uncertainty associated. It is done by linearizing the motion model (f in Equation 2.1) and integration a Gaussian estimate of the state $x \sim \mathcal{N}(\hat{x}, \Sigma)$ as follows:

$$\hat{x} \leftarrow f(\hat{x}, u, 0) \ . \tag{2.9}$$

$$\Sigma_s \leftarrow J_s \Sigma_s J_s^T + J_v \Sigma_v J_v^T \ . \tag{2.10}$$

Where J_s, J_v are respectively the Jacobians of $f()$ with respect to x and the perturbation v and Σ_s the covariance matrix x.

Differential Drive Odometry

Odometry is a composition of estimated relative transforms - local pose increments. Given the control signal $u = [\delta p, \delta\theta] \in \mathbb{R}^3$, Equation 2.9 is such that:

$$p \leftarrow p + R\{\theta\}(\delta p + \delta p_t) \ . \tag{2.11}$$

$$\theta \leftarrow \theta \ + \delta\theta \ + \delta\theta_i \ . \tag{2.12}$$

where $R\{\theta\}$ is a $2D$ rotation matrix associated with the angle θ. The Equation 2.11 correspond the a composition of $2D$ rigid-body transforms between the robot previous pose and the odometry increment.

The differential drive model is derived for a robot such that it has two actuated wheels, one on each side of its base, with its origin frame located at the center of the wheels axis.

The motion is usually measured by means of wheel encoders reporting incremental wheel angles - $\delta\psi_L$,$\delta\psi_R$ - every time step δt. In this model the robot is parametrized by three parameters, its wheels radii r_L,r_R and the length of the axis joining both joints d.

Assuming δt to be small, the increment angle $\delta\theta$ is small too, then the motion increment u_t is so that:

$$\delta x = \frac{r_L * \delta\psi_L + r_R * \delta\psi_R}{2} \ . \tag{2.13}$$

$$\delta y = 0 \ . \tag{2.14}$$

$$\delta\theta = \frac{r_L * \delta\psi_L - r_R * \delta\psi_R}{d} \ . \tag{2.15}$$

The uncertainty covariance Σ_f used for uncertainty integration in Equation 2.10 is obtained from the uncertainty of the wheel angle measurements as follows:

$$\Sigma_f = J_f \Sigma_\psi J_f^T \ . \tag{2.16}$$

with Σ_ψ the wheel measurement covariance:

$$\Sigma_\psi = \begin{bmatrix} \sigma_{\psi_l}^2 + \alpha^2 & 0 \\ 0 & \sigma_{\psi_r}^2 + \alpha^2 \end{bmatrix}, \qquad \sigma_{\psi_l} = k_l * v, \sigma_{\psi_r} = k_r * \omega, \alpha = (\mu_l + \mu_r) * 0.5 \ . \tag{2.17}$$

where k_r and k_l are wheels intrinsics parameters, α acts as an offset equal to half the wheels encoders resolution μ_l and μ_r.

In case $\delta\theta$ is **not small** the contribution of the rotation angle on the transversal translation δy must be taken into account. Yielding the following:

$$\delta x = \frac{r_L * \delta\psi_L + r_R * \delta\psi_R}{2} \frac{\sin(\delta\theta)}{\delta\theta} \ . \tag{2.18}$$

$$\delta y = \frac{r_L * \delta\psi_L + r_R * \delta\psi_R}{2} \frac{1 - \cos(\delta\theta)}{\delta\theta} \ . \tag{2.19}$$

$$\delta\theta = \frac{r_L * \delta\psi_L - r_R * \delta\psi_R}{d} \ . \tag{2.20}$$

Twist Control Model A differential-drive model is typically controlled via linear and angular velocities in the robot frame also called twist $[v, \omega]$. Assuming constant velocity inputs over $[t, t_{+1}]$, the robot moves along an arc of circle of radius $r = v_t/\omega_t$. The twist control is expressed in term of motion increment u_t such as:

$$\delta x = r \sin(\delta\theta) \ . \tag{2.21}$$

$$\delta y = -r(\cos(\delta\theta) - 1) \ . \tag{2.22}$$

$$\delta\theta = \omega\delta t \ . \tag{2.23}$$

with the associated Jacobians:

$$J_m = \begin{bmatrix} \frac{\sin(\delta\theta)}{\omega} & \frac{v}{\omega}(\cos(\delta\theta) - \frac{\sin(\delta\theta)}{\omega}) \\ \frac{1-cos(\delta\theta)}{\omega} & \frac{v}{\omega}(\sin(\delta\theta) + \frac{1-\cos(\delta\theta)}{\omega}) \\ 0 & 1 \end{bmatrix} \ . \tag{2.24}$$

$$J_s = \begin{bmatrix} 1 & 0 & \frac{v}{\omega}(\cos(\delta\theta) - 1) \\ 0 & 1 & \frac{v}{\omega}(\sin(\delta\theta)) \\ 0 & 0 & 1 \end{bmatrix} . \tag{2.25}$$

where J_m is the Jacobian w.r.t the measurement and J_s the Jacobian w.r.t the robot state.

In a similar manner as previously, in case the robot follows a straight trajectory requires special consideration. Since $\omega = 0$, the arc circle radius is so that:

$$r = \frac{v}{\omega} = \frac{v}{0} \to \infty$$

Indeed, a line is a degenerate case of a circle which radius $\to \infty$. This case is handled by means of an approximate integration using a second order Runge-Kutta integration:

$$\delta x = v\delta t * \cos(\omega\delta t * 0.5) . \tag{2.26}$$

$$\delta y = v\delta t * \sin(\omega\delta t * 0.5) . \tag{2.27}$$

$$\delta\theta = \omega\delta t . \tag{2.28}$$

with the associated Jacobians:

$$J_m = \begin{bmatrix} \cos(\omega * 0.5) & -0.5 * v * \sin(\omega * 0.5) \\ \sin(\omega * 0.5) & 0.5 * v * \cos(\omega * 0.5) \\ 0 & 1 \end{bmatrix} . \tag{2.29}$$

$$J_p = \begin{bmatrix} 1 & 0 & -0.5 * v * \sin(\omega * 0.5) \\ 0 & 1 & 0.5 * v * \cos(\omega * 0.5) \\ 0 & 0 & 1 \end{bmatrix} . \tag{2.30}$$

where J_m is the Jacobian w.r.t the measurement and J_s the Jacobian w.r.t the robot state.

LRF-based Odometry

This estimation can be addressed by tracking a sensor pose directly from its readings. Laser-scan based odometry is the process of estimating the robot trajectory from consecutive LRF readings. Given two consecutive readings, the odometry algorithm is two folds; first compute a scan-to-scan data association, then estimate the relative transform that aligns the associated data. Doing so over time allows one to compute pair-wise relative transforms which once integrated provides an estimate of the robot trajectory.

In the literature this problem has been addressed in many ways, such as algorithm similar to the Iterative Closest Point (ICP) [23]. Other methods constitute direct variations of the original ICP formulation [24] such as [25], which uses a custom metric in place of the Euclidean distance to lessen the increase of distance due to the sensor rotation. Other variants include Iterative Closest Line (ICL) proposed in [26] whose error function relies on a point-to-line distance rather than point-to-point as in the classical ICP. Other LRF-based odometry estimation algorithms include feature-based data association [27] which extracts features from the range data. Doing so they discard the need for an iterative process inherent

of ICP-based methods. Rather than considering the range-reading in the Cartesian plane, [28, 29] perform the data association in polar coordinates. More recent methods align consecutive scans by means of correlation [30]. The scans are projected on 2D occupancy grids, which are then correlated. Recently, a formulation of the Optical-Flow has been successfully proposed for 2D laser scan [31].

2.2.3 Loop-Closure

Loop closure detection is an essential module of any SLAM system. It is the process of (re)identifying a place from the corpus of places visited in the past. Unlink the odometry module mentioned in Section 2.1 which estimates the robot pose in short terms, the loop closure module estimates the robot pose against the global map therefore comparing the current pose against the history of poses.

This module reduces the uncertainty in the estimated map that accumulates during open loop mapping. It is exemplified in Figure 2.3. The robot estimated trajectory highlighted with dashes drift over time. When revisiting a place, the module identify it as being part of its corpus and estimates the relative transform from the current pose to the identified one (the red segment in Figure 2.3). Doing so it creates and adds a constraint to the problem allowing to correct the accumulated drift.

This is closely related to the place recognition problem up to the difference that in the case of place recognition, one only seek for a topological match hence do not necessarily compute the relative transform between the two matched places.

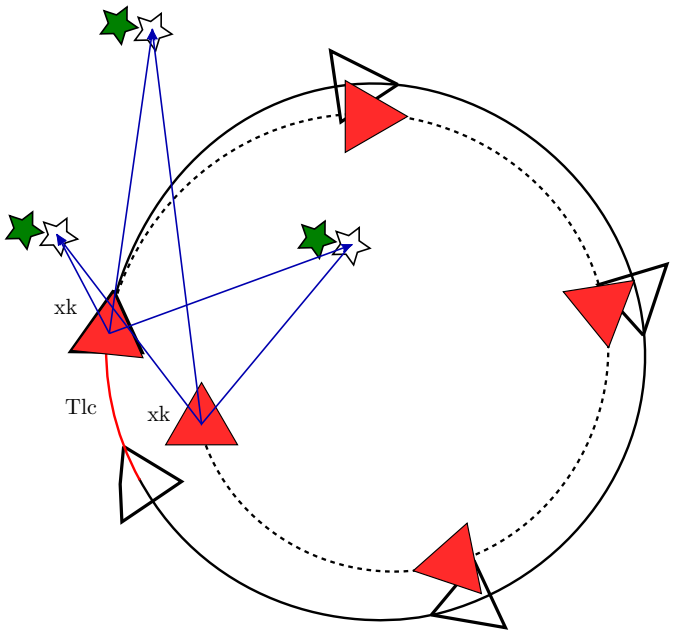


Figure 2.3: A loop closure.

Loop closure detection has been tackled with geometric methods (see *e.g.* [32]), correlation methods (see *e.g.* [33]), or with appearance-based methods. Appearance can be considered either globally [34, 35, 36, 37] or as a set of local distinctive features [38, 39, 11] possibly extracted from different sensor modalities [40]. After the initial work of the computer vision community on the use of Bag-of-Words (BoW) for object recognition [41, 42, 43], the SLAM community found in BoW an efficient manner to query large corpus of places visited by a robot while mapping [44, 45], hence its amenity for the solution of the

loop closure problem. More recently, state of the art visual SLAM algorithms have relied on BoW for their loop closure and re-localization modules. ORB-SLAM [2] for instance uses DBoW2 [46], whereas LSD-SLAM [1] relies on FAB-MAP [47].

The reader can find an recent and complete survey of visual place recognition in [48].

Chapter 3

Preliminary Work

3.1 Preliminary Work

The preliminary work presented here focus on two of the three SLAM module presented in Section 2.1 namely the odometry module and the loop-closure module. We first aim at bringing to LRF-based SLAM some of the latest concepts developed in V-SLAM, such as local windowed optimization for both trajectory and map refinement. In a second time we will explore the benefits of fusing sensors to tend to a robust continous SLAM. Finally our work will focuses on strengthening the accuracy in place recognition.

3.1.1 Odometry

As mentioned in Section 2.1, V-SLAM received a great deal of attention and great advances where made over the past few years. Following the strategy developed in [49], state of the art algorithms such as [1, 2] decoupled the odometry estimation from the overall SLAM problem (eventually splitting it in three modules - see Section 2.1). This particular sub-problem is called visual odometry [50, 51].

LRF-based odometry did not benefited from the novel approach of V-SLAM such as windowed local optimization [1, 2, 52].

LRF-based Odometry as a Local Pose Optimization

We propose to investigate the benefit of performing a local optimization around the current robot pose in order to refine the estimated odometry.

Whereas most LRF odometry perform a scan-to-scan matching in order to track the estimated odometry we propose to fit the current scan to N previously pre-registered scans and estimate $N-1$ transforms all together. This is known as the global registration problem.

Global registration of multiple point-clouds has been extensively investigated eventually leading to the formalization of Generalized Procruste Analysis (GPA) [53, 54]. GPA computes the best set of transformations (e.g. Euclidean, Similarity, Affine) relating matched shape data. It can be optimized following either the *Reference-space model* (eq. 3.1) or more often following the *Data-space model* (eq. 3.2) [55]. The latter alternates the computation of a reference shape (or control shape) from the matched shape

data and the computation of transformations relating each date to the reference.

$$E_R(T, M) = \sum \|T_i \cdot D_i - M\| \ . \tag{3.1}$$

$$E_D(T, M) = \sum \|D_i - T_i^{-1} \cdot M\| \ . \tag{3.2}$$

With M the reference shape, D_i the ith data shape and T_i the rigid transform that maps D_i onto M. The GPA formulation is well suited to the iterative nature of ICP and [56] proposes an embedding of GPA *Reference-space model* within the iterative process of ICP. It has to be noted that in Euclidean transformation case, both model's cost are identical [55].

However, since the *Data-space model* formulates the sensor noise as occurring in the observations we choose this formulation to later include the sensor model parameters as variables of the optimization process. For that reason we use Nonlinear Least Square (NLS) on the $SE(2)$ manifold to solve 3.2.

As aforementioned, the GPA algorithm is composed of three parts. First the data association aims at finding the correspondences between two scans to be registered. This is done by mean of a nearest neighbor search for which we plan on using the metric-based point-to-point. In the plan, a sensor pose is represented by an Euclidean transformation defined by a vector $q = (x, y, \theta)$. In the plan the metric-based point-to-point measure [25] defines the norm of q as follows:

$$\|q\| = \sqrt{x^2 + y^2 + L^2\theta^2} \ . \tag{3.3}$$

With L a positive real number homogeneous to a length.
Given two points $p_1 = (p_{1x}, p_{1y})$ and $p_2 = (p_{2x}, p_{2y})$ the distance between p_1 and p_2 derived from 3.3 is as follows:

$$d_mb(p_1, p_2) = \sqrt{\delta_x^2 + \delta_y^2 - \frac{(\delta_x p_{1y} - \delta_y p_{1x})^2}{p_{1x}^2 + p_{1y}^2 + L^2}} \ . \tag{3.4}$$

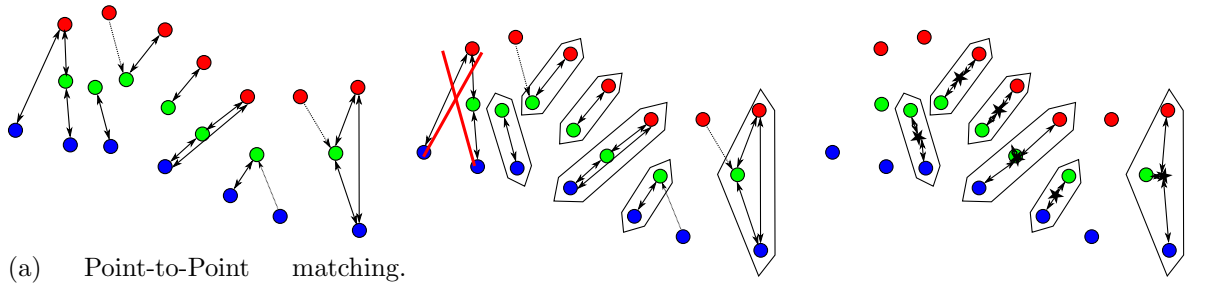
where $\delta_x = p_{2x} - p_{1x}$ and $\delta_y = p_{2y} - p_{1y}$.
This distance implies that the iso-distance curves relative to $d_mb(p_1, p_2)$ are ellipses centered on p_1 thus L acts as weighting factor between translation and rotation. Without further development, Equation 3.4 3D counter part is detailed in [57].

The second part of the GPA algorithm is the computation of the reference shape. Following the work of [56] the point-to-point correspondences are established between every combination of pairs of scans by means of a nearest neighbor search using the distance metric in Equation 3.3. Correspondences are pruned in order to keep only those that are mutually matching in a given pair of scans (Figure 3.1a). From the pruned correspondences we seek for independent sets of matched points as shown in Figure 3.1b. To establish independent sets one may look at the problem as a sparsely connected graph where every points of every scans represents a node and mutual matches represent edges. Independent sets can then be seen as graph cliques, thus been established using algorithm for finding all maximal cliques in an undirected graph [58, 59]. One may also approximates cliques as strongly connected components for less computationally expensive algorithms such as [60]. Finally, for each independent set a centroid is computed and new matches from each point of a set to their corresponding centroids are computed (Figure 3.1c). The set of all centroids constitutes the reference shape.

The third part of the GPA algorithm is then solving Equation 3.2 given the reference shape computed at the previous step.

Notice that whereas the data association is performed using Equation 3.4, in the optimization process we use the stricter Euclidean distance leading to a mixed-ICP such as proposed in [61].

The overall algorithm is summarized in Algorithm 1.



(a) Point-to-Point matching. Dashed arrows mark non-mutual matching.

(b) Selected independent sets.

(c) Centroids correspondences.

Figure 3.1: Independents sets computation steps highlight between three scans.

input : currentScan, the currently evaluated scan.
input : keyScans, N key-scans.
output : T, estimated robot pose.
parameter: maxIteration, maximum number of iterations

do

  for $(i \leftarrow 1,\ j \leftarrow 1)$ **to** $N,\ i \neq j$ **do**
    matches $[i, j] \leftarrow$ `FindNearestNeighbor(keyScans` $[i]$, keyScans $[j]$`)`;
  end

  [controlScan, controlMatches] $\leftarrow$ `ComputeControlScan(matches)`;

  $[T_1, ..., T_{n-1}] \leftarrow$ `EstimatesRTS(controlScan,` controlMatches`)`;

  converged $\leftarrow$ `HasConverged(`$[T_1, ..., T_{n-1}]$`)`;

while *not* converged *or* maxIteration;

if `IsKeyScan(`T_{n-1}`)` **then**
  `AddKeyScan(currentScan)`;
end

return currentKeyPose $\cdot T_{n-1}$;

Algorithm 1: Pose optimization LRF odometry

LRF-based Odometry as a Joint Optimization of Laser Readings and Robot Poses

The algorithm presented in Section 3.1.1 can be further extended toward a complete local optimization. We propose to include in the optimization problem the refinement of the laser readings. By doing so, we expect to to compensate for the sensor noise which in turn will lead to refined readings hence clearer occupancy-grid - in the sense of reducing its entropy. We plan on following the line of work presented in [62]. In this work the authors model the relation between the environment and the sensor readings using

a surface-based sensor model. Such model is twofold. First the environment is assumed to be man-made hence that it is mainly composed of smooth surface locally approximated by a tangent line segment. This tangent is approximated by the smallest eigenvector of a covariance matrix centered on the laser reading of interest and capturing its neighborhood. Secondly, the sensor model is so that each laser beam is considered to have a conic shape. Such assumption implies that the beam do not hit the surface in a single point but rather on an elliptical surface. The ellipse's shape is then driven by three parameters, the laser aperture together with the distance separating the sensor to the surface it hits and the incidence angle to the surface. The distance is therefore averaged over large region, leading to less accurate measurements. Unlink [62] whom performs a global optimization off-line after an initial optimization over the Pose-graph, we aim at performing the joint optimization online within the local optimization window used to compute the multi-scan ICP.

3.1.2 Loop-Closure

The loop closure module is probably the stricter module in terms of precision of the overall SLAM framework. Indeed wrong loop-closure may have a catastrophic effect on the problem estimation while outlier loop-closures are difficult to identify and filter-out. Its precision must then been strengthen targeting $99 + \%$ precision while maintaining a satisfactory recall so that as many loops as possible are closed. [1].

Feature Comparison for BoW-based Visual Place Recognition

As mentioned in Section 2.2.3 modern V-SLAM rely on the BoW scheme for their loop-closure module.

In the BoW framework, the objective is to find a document in a database with the largest similarity score to a query document. For that end, it includes two distinct elements. First, a *vocabulary*, $W = \{w_1, ..., w_k\}$, composed of cluster centers or *words*, w_k, representing the feature space. The vocabulary of words is built offline from a dataset unrelated to the later use by mean of a hierarchical k-means [43, 63]. The second element consists of a *database* composed of *documents*, $D = \{d_1, ..., d_N\}$, where each document d_j represents the BoW associated to a sensor reading at a known pose of the robot in the current map. That is, the set of local features in the vocabulary detected in a given sensor reading and their local coordinates.

The database keeps a record of each word occurrence in every document by means of two frequency scores. The *term frequency* (tf) refers to how frequent a single word is within a document, and the *inverse document frequency* (idf) refers to how frequent is a single word in the whole database. Given a word w_i in document d_j, these frequencies are computed as follows:

$$tf_{ij} = \frac{n_{ij}}{\sum_i n_{ij}} \ , \tag{3.5}$$

$$idf_i = \log\left(\frac{|D|}{\sum_j |n_{ij} > 0|}\right) \ , \tag{3.6}$$

where n_{ij} is the occurrence of the word i in document j, $|D|$ the size of the database and $|n_{ij} > 0|$ evaluates to 1 if w_i occurs in d_j and 0 otherwise. The *weight* of every word w_i in each document d_j is

[1] With $precision = \frac{true\ positive}{true\ positive\ +\ false\ positive}$ and $recall = \frac{true\ positive}{true\ positive\ +\ false\ negative}$

given by its *tf-idf* score, which is computed with

$$x_{ij} = tf_{ij} \cdot idf_i \ . \tag{3.7}$$

A document is characterized by its *signature*, a vector containing its *tf-idf* weights, $sig_j = [x_{j1}, x_{j2}, ..., x_{jk}]^T$. The document comparison is performed by computing the cosine similarity of their signatures:

$$sim_{lm} = \frac{sig_l^T \, sig_m}{\|sig_l\| \|sig_m\|} \ . \tag{3.8}$$

Given a new sensor reading (a query), its feature descriptors are extracted, quantized into words, and its signature compared to those of every document in the database; the N most similar documents are returned by the BoW scheme.

Finally, a consistency check needs to be made to assert which if any of the returned documents is a good match to the query sensor reading. In the case of visual place recognition, the consistency check can be for instance the estimation of an Essential matrix, a trifocal tensor [64] or a PnP projection [65].

Following the work of [66] we first aimed at producing an extended comparison of visual local features for BoW-based place recognition. The comparison helps to better understand how the choice of a given features influence the recognition performance, together with the execution time. The comparison is perform against 15 meaningful combinations of the point-based local feature algorithms available in the OpenCV library [67]. Table 3.1 summarizes the available algorithms while Figure 3.2 lists the tested combination. The experiments were conducted on the Kitti dataset [68] and can be summarized as follows:

- A large and random image dataset is used to train the BoW vocabulary for each feature type.
- The Kitti sequences are pre-processed in order to select a subset of images that are going to compose the BoW databases.
- Each sequence is processed by the BoW framework for each feature type. The BoW geometrical check if perform by estimating an Essential matrix in a Random Sample Consensus (RANSAC) scheme. Given the images position ground-truth are provided by the dataset, the precision and recall scores are computed for each feature type.

Detectors	orb	sift	surf	KAZE	akaze	brisk	mser	fast	ed	agast
Descriptors	orb	sift	surf	KAZE	akaze	brisk	daisy	latch	rootsift	brief

Table 3.1: Local features detectors and descriptors available in OpenCV.

All user-defined parameters of the RANSAC scheme were set so that the precision scores are above 99% as one would expect in a SLAM loop-closure context. Early results show that despite its common usage, the ORB feature may not the best suited for this task. The combination of the AGAST detector and the BRIEF descriptor increases the recall by $\sim 10\%$ for a very similar cost in terms of execution time. These results are shown in Figure 3.2. Notice that the four combinations of detector and descriptor performing better than AGAST+BRIEF are an order of magnitude slower.

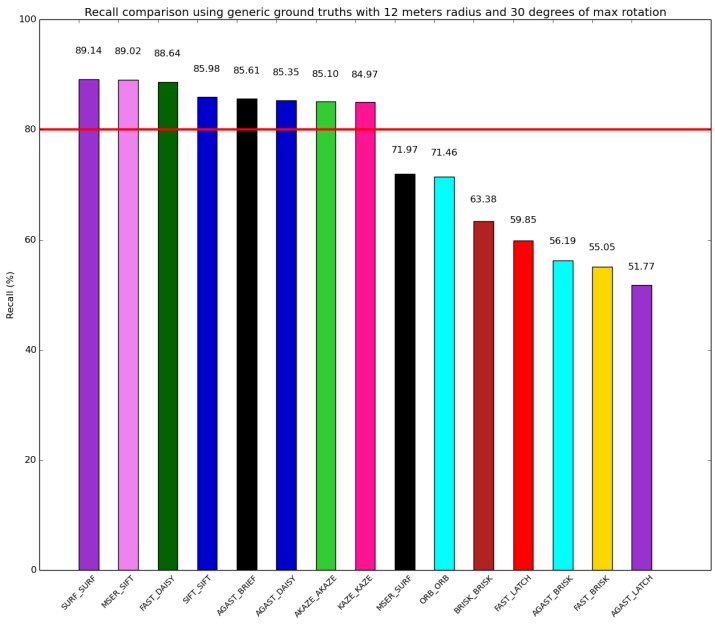


Figure 3.2: Comparison of OpenCV features for BoW-based place recognition recall at 99+% precision.

A related media is available online:

TiaGo visual place recognition. : `https://www.youtube.com/watch?v=jDFwzBNhNek`

BoW-based Loop-Closure with 2D Laser Scan

Unlike V-SLAM, the following focuses on the creation of a BoW for the treatment of 2D laser range data. There is little published work on appearance-based place recognition using 2D laser scans, possibly due to the fact that reliable feature detectors/descriptors were developed later than their image based counterparts. The local feature Fast Laser Interest Point Transform (FLIRT) is robust to scale and orientation changes [69] and thus allows a direct application of BoW for the problem of place recognition [70]. As for global descriptors, the Geometrical Landmark Relations (GLARE) [71] encodes the geometrical relations of FLIRT corners in an histogram of relative distance over relative orientation. Extending GLARE, the Geometrical Surface Relations [72] descriptor considers every reading of the 2D laser scan rather than extracted corners.

In contrast with other sensing modalities, 2D LRF data present a natural (counter-)clockwise ordering of its local features which can be easily exploited to reinforce the computation of scan similarity. We hence draw some empirical observations regarding this ordering, and use them in an algorithm that computes the best feature correspondence assignment between two scans.

The observations are the following, given local features extracted from a 2D scan (quantized into words) they are ordered clock-wise in a sequence. This ordering must remain the same for a given scene observed from slightly different viewpoints. As the viewpoint change increases, features can disappear, shift their location in the sequence or reorder.

Feature Sequence Encoding as a Hidden Markov Model Feature matching is done directly on words. So, a given descriptor quantized into a particular word w, can only match features also quantized as w and in no case could match another word in the vocabulary. This is exemplified in Fig. 3.3 (top).The problem of scan alignment is then analogous to finding the path that maximizes the sequence of feature matches in a Hidden Markov Model (HMM). Consider the query laser scan l_i and its extracted words $w_{1i}, \ldots, w_{Ni}$ as the set of states S_N in the model. Consider also the candidate match l_j with its words $w_{1j}, \ldots, w_{Mj}$ as a set of observations O_M. We can define our HMM such that:

- We have equal initial probabilities $\delta_{s_n} = \frac{1}{N}$.
- The transition from one state to another solely goes forward with respect to the clockwise ordering of the states. Self transitions have a lower probability to enforce the importance of alignments $\phi_{s_n|s_n} = \frac{0.5}{F}$, $\phi_{s_n|s_{n+x}} = \frac{1+\frac{0.5}{F-1}}{F}$, and $\phi_{s_n|s_{n-x}} = 0$, where F is the number of states following the currently evaluated state in the ordered sequence.
- The output probability is defined such that a word mismatch has null probability whereas a word match has equal probability. Hence our emission probabilities are $\theta_{s_n|o_m} = \frac{1}{C}$ for a match, and $\theta_{s_n|o_m} = 0$ for a mismatch, where C is the number of matches of the currently evaluated word.

Fig. 3.3 (middle) gives an unnormalized representation of the HMM produced by the matching of words in scans l_i and l_j. Black downward pointing arrows indicate feasible transitions, and red upward pointing arrows indicate non-feasible transitions. Each cell is then filled by the product $\phi_{s_n-x|s_n} \cdot \theta_{s_n|o_m}$, where s_n is the currently evaluated state, s_{n-x} is the previous most likely state and $\theta_{s_n|o_m}$ the output probability. Columns are filled recursively based on the previous iteration.

Unlike [73], the HMM is built based on the inner ordering of two independent set of features extracted from raw sensor readings, whereas [73] builds a HMM based on the inner ordering of two independent sequences of key-frames, hence not impacting the frame-to-frame similarity measure. Once the HMM is built, the goal is then to find a sequence of states that maximizes the probability of a path across it.

The Viterbi Algorithm In order to find the most probable path at a reasonable cost in terms of computation, we propose the use of the Viterbi algorithm [74]. This dynamic programming algorithm searches recursively for the most likely sequence of states given a sequence of events, by computing for each observation the partial probability with respect to the previous state that optimally induced the current state. Such sequence is called the Viterbi path. It is commonly used in speech recognition, speech synthesis and decoding [75, 76].

Crossing edges in Fig. 3.3 (top) highlight mismatches. These might occur either because different features are quantized to the same word (e.g., words C and F), or because the feature is on a moving object. The work in [70] does not discard such mismatches while constructing the offset histogram, and thus they can not be taken into account to compute a consistent relative transform. Thanks to the constraint of forward state transition, the Viterbi algorithm naturally discards such crossing edges. Note that in our example, crossing edges for the sequence C-D-E can be resolved in two different ways, either by removing the match C and keeping D and E, or removing the latter keeping only C. The Viterbi algorithm maximizes the sequence of states in the Viterbi path and hence it would prefer, in this case, to keep matches D and E and discard C.

Scoring Once the Viterbi path is obtained, the candidate is scored based on three criteria:

- the number of correct matches that have not been discarded by the Viterbi algorithm,
- the number of sequences of consecutive words that have a correct match, and
- the distribution of matches in the laser scans. The wider the better.

The second point is similar to the concept of phrases in [70], where a phrase represents a sequence of consecutive words, analogous to a n-grams model.

Considering such sequences and weighting them according to their length we can add an extra layer of constraints to our geometric check. These criteria evaluate respectively to: $score_{jk} = \frac{|M|}{|C|}$, where $|M|$ is the number of correct matches and $|C|$ the number of features in the candidate scan; $weight_{jk} = \frac{|CM|}{|C|}$, where $|CM|$ is the number of sequences of consecutive correct matches, e.g., sequences *A-B* & *D-E* in Fig. 3.3 (top); and $ratio_{jk} = \frac{Id_r - Id_l}{|C|}$, where Id_r and Id_l are the indices of the rightmost- and leftmost- correct matches in the Viterbi path, respectively. These three criteria are aggregated into a final geometric score,

$$g_{jk} = \frac{score_{jk} + weight_{jk}}{2} \cdot ratio_{jk} \ . \tag{3.9}$$

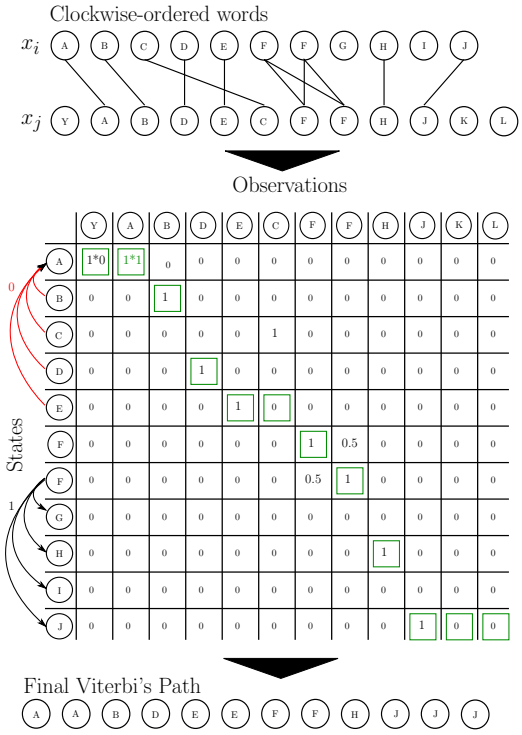


Figure 3.3: Top: Clockwise ordered words of two scans and their matches. Middle: The resulting hidden Markov model. Each cell represents the product $\phi_{s_n - x|s_n} \cdot \theta_{s_n|o_m}$ e.g cells *Y-A* & *A-A*. Green squared cells represent the best path across the complete graph. Bottom: The final sequence of states given the observations O_n

While querying the BoW database, both the *tf-idf*-based similarity and the geometric score in (3.9) are computed for each document in the database. The two are then aggregated into a single similarity term,

$$sg_{jk} = sim_{jk} \cdot g_{jk} \ . \tag{3.10}$$

This aggregated similarity term is then used to rank BoW candidates instead of the *tf-idf*-based similarity.

Pose-Graph Database Augmentation In this section we detail a topological augmentation of the BoW database. By *augmentation* we refer to the fact of benefiting from common features in adjacent poses in the pose-graph of our map for the computation of the *tf-idf* weights. Since the pose-graph is computed by our SLAM front end, our database augmentation involves no computation overhead.

In pose-graph SLAM, every node holds a robot pose and a sensor measurement, and every edge between two nodes represents a spatial constraint –a relative transform– usually computed from the sensor measurements. The most likely map is obtained by jointly optimizing for all pose constraints in the graph.

Database augmentation taking the form of a similarity graph has been proposed in [77] and [78] for the task of image recognition. Graph edges are created by matching image features and asserting an affine transform between images through RANSAC. Direct edges represent document adjacencies; documents connected to an adjacent document then represents 2-adjacencies, and so on. The set E_j of adjacencies of document d_j is used to emphasize the *tf* weight of the document,

$$m_{ij} = n_{ij} + \sum_{k \in E_j} n_{ik} \ , \tag{3.11}$$

$$atf_{ij} = \frac{m_{ij}}{\sum_i m_{ij}} \ . \tag{3.12}$$

These normalized scores (3.12) constitute the *adjacency tf* used as a direct drop-off replacement for (3.5) in (3.7), so that the *tf-idf* weight becomes

$$x_{ij} = atf_{ij} \cdot idf_{ij} \ . \tag{3.13}$$

While for object recognition the database augmentation is based on object appearance similarity, in the case of place recognition within a SLAM framework the topological distribution of the places matters. Since an edge in a pose-graph SLAM is computed from sensor readings and represents a spatial constraint, it embeds both the appearance-based similarity required by the BoW scheme (consecutive nodes share some common features) and the topological information that we want to emphasize by the database augmentation.

Whereas object recognition usually considers a pre-trained database for which an offline database augmentation can be computed [77, 78], in the case of place recognition within a SLAM framework the database together with its augmentation are constructed online. Using the SLAM pose-graph built online by another module of the SLAM framework allows for a database augmentation at no extra cost.

Finally, [78] identifies useful features (features belonging to a transformation inlier set) from the document adjacencies and discards the others. Since we build the database online, we keep all of them, as they can become useful later on during mapping.

Implementation The overall aforementioned algorithm have been developed in C++ in such way that it is agnostic to the SLAM front-end and limits its interaction with the core module to:

- receiving new key-frames' raw data sensor and its direct adjacency with the previous key-frame.

- informing of loop-closure detections, in the form of pose-graph constraints.

Further Work The overall framework being developed for the use of 2D LRF we now aim at extending its use to other sensors. Targeting first camera but also 3D LRF, this leads to the the development of a multi-modal place recognition using several sensors at once, each sensor compensating for each other's drawbacks.

Moreover, whereas BoW relies on low-level features extracted from the sensor readings (e.g. corners), our second line of development aims at using higher-level features such as texture or objects.

3.2 Publications

The preliminary work presented in Section 3.1.2 has given rise to the following publication.

"Word Ordering and Document Adjacency for Large Loop Closure Detection in 2D Laser Maps"
J. Deray, J. Solà, J. Andrade-Cetto. IEEE Robotics and Automation Letters, vol. PP, no. 99, pp 1-1, 2017. [79]

Abstract – We address in this paper the problem of loop closure detection for laser-based simultaneous localization and mapping (SLAM) of very large areas. Consistent with the state of the art, the map is encoded as a graph of poses, and to cope with very large mapping capabilities, loop closures are asserted by comparing the features extracted from a query laser scan against a previously acquired corpus of scan features using a bag-of-words (BoW) scheme. Two contributions are here presented. First, to benefit from the graph topology, feature frequency scores in the BoW are computed not only for each individual scan but also from neighboring scans in the SLAM graph. This has the effect of enforcing neighbor relational information during document matching. Secondly, a weak geometric check that takes into account feature ordering and occlusions is introduced that substantially improves loop closure detection performance. The two contributions are evaluated both separately and jointly on four common SLAM datasets, and are shown to improve the state-of-the-art performance both in terms of precision and recall in most of the cases. Moreover, our current implementation is designed to work at nearly frame rate, allowing loop closure query resolution at nearly 22 Hz for the best case scenario and 2 Hz for the worst case scenario. Media related to the publication are available online:

Many BoW-based Loop Closure with Laser Scan only. : `https://www.youtube.com/watch?v=EZoCmsXixp0`
BoW-based Loop Closure with Laser Scan only. : `https://www.youtube.com/watch?v=O4xRdzUYAQs`

Chapter 4

Research Plan Summary

4.1 Work Plan

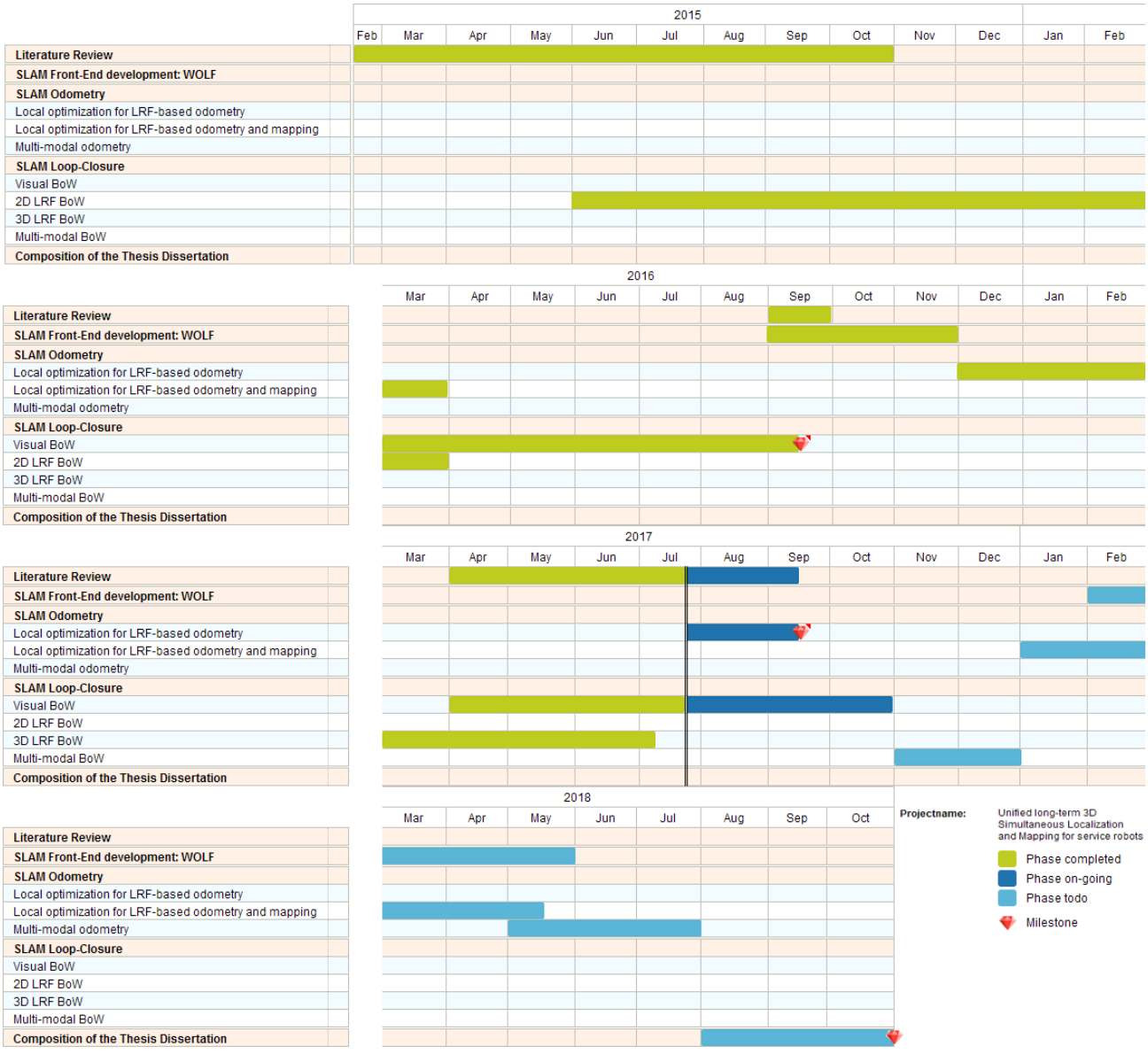


Figure 4.1: Work plan of the proposed research.

Bibliography

[1] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. 13th Eur. Conf. Comput. Vis*, vol. 8689 of *Lect. Notes Comput. Sci.*, (Zurich), pp. 834–849, 2014.

[2] R. Mur-Artal, J. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[3] H. F. Durrant-Whyte and T. Bailey, "Simultaneous Localisation and Mapping (SLAM): Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[4] T. Bailey and H. F. Durrant-Whyte, "Simultaneous Localisation and Mapping (SLAM): Part II," *Robotics and Autonomous Systems (RAS)*, vol. 13, no. 3, pp. 108–117, 2006.

[5] G. Dissanayake, S. Huang, Z. Wang, and R. Ranasinghe, "A review of recent developments in Simultaneous Localization and Mapping," in *International Conference on Industrial and Information Systems*, pp. 477–482, IEEE, 2011.

[6] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, p. 1729881416669482, 2016.

[7] 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: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, pp. 1309–1332, Dec 2016.

[8] G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem," *IEEE Transactions Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.

[9] R. M. Eustice, H. Singh, and J. J. Leonard, "Exactly sparse delayed-state filters for view-based slam," *IEEE Transactions on Robotics*, vol. 22, pp. 1100–1114, Dec 2006.

[10] J. Nieto, T. Bailey, and E. Nebot, *Scan-SLAM: Combining EKF-SLAM and Scan Correlation*, pp. 167–178. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[11] K. Konolige and M. Agrawal, "FrameSLAM: From bundle adjustment to real-time visual mapping," *IEEE Trans. Robotics*, vol. 24, pp. 1066–1077, Oct 2008.

[12] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-Based Compact Pose SLAM," *IEEE Transactions on Robotics (TRO)*, vol. 26, no. 1, pp. 78–93, 2010.

[13] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," in *Proceedings of the 4th International Symposium on Robotics Research*, (Cambridge, MA, USA), pp. 467–474, MIT Press, 1988.

[14] P. Moutarlier and R. Chatila, "Stochastic multisensory data fusion for mobile robot location and environment modeling," in *5th Int. Symposium on Robotics Research*, vol. 1, Tokyo, 1989.

[15] J. J. Leonard, H. F. Durrant-Whyte, and I. J. Cox, "Dynamic map building for an autonomous mobile robot," *The International Journal of Robotics Research (IJRR)*, vol. 11, no. 4, pp. 286–289, 1992.

[16] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *Eighteenth National Conference on Artificial Intelligence*, (Menlo Park, CA, USA), pp. 593–598, American Association for Artificial Intelligence, 2002.

[17] J. L. Blanco, J. A. Fernández-Madrigal, and J. Gonzalez, "A Novel Measure of Uncertainty for Mobile Robot SLAM with Rao-Blackwellized Particle Filters," *The International Journal of Robotics Research (IJRR)*, vol. 27, no. 1, pp. 73–89, 2008.

[18] F. Dellaert and M. Kaess, "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing," *The International Journal of Robotics Research (IJRR)*, vol. 25, no. 12, pp. 1181–1203, 2006.

[19] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental Smoothing and Mapping," *IEEE Transactions on Robotics (TRO)*, vol. 24, no. 6, pp. 1365–1378, 2008.

[20] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree," *The International Journal of Robotics Research (IJRR)*, vol. 31, pp. 217–236, 2012.

[21] L. Polok, V. Ila, M. Solony, P. Smrz, and P. Zemcík, "Incremental block cholesky factorization for nonlinear least squares in robotics," in *Robotics: Science and Systems*, 2013.

[22] V. Ila, L. Polok, M. Solony, and P. Svoboda, "Highly efficient compact pose SLAM with SLAM++," *CoRR*, vol. abs/1608.03037, 2016.

[23] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, no. 3, pp. 249–275, 1997.

[24] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, pp. 239–256, Feb. 1992.

[25] J. Minguez, F. Lamiraux, and L. Montesano, "Metric-based scan matching algorithms for mobile robot displacement estimation," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3557–3563, April 2005.

[26] A. Censi, "An ICP variant using a point-to-line metric," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (Pasadena, CA), May 2008.

[27] A. A. Aghamohammadi, H. D. Taghirad, A. H. Tamjidi, and E. Mihankhah, "Feature-based laser scan matching for accurate and high speed mobile robot localization.," in *EMCR*, 2007.

[28] A. Diosi and L. Kleeman, "Fast laser scan matching using polar coordinates," *The International Journal of Robotics Research*, vol. 26, no. 10, pp. 1125–1153, 2007.

[29] G. Huo, L. Zhao, K. Wang, R. Li, and J. Li, "Polar metric-weighted norm-based scan matching for robot pose estimation," *Discrete Dynamics in Nature and Society*, vol. 2016, 2016.

[30] E. B. Olson, "Real-time correlative scan matching," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 4387–4393, IEEE, 2009.

[31] M. Jaimez, J. G. Monroy, and J. González-Jiménez, "Planar odometry from a radial laser scanner. a range flow-based approach," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4479–4485, 2016.

[32] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, pp. 34–46, Feb 2007.

[33] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1271–1278, 2016.

[34] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, 2001.

[35] N. Sünderhauf and P. Protzel, "BRIEF-Gist - closing the loop by simple means," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, (San Francisco), pp. 1234–1241, Sep. 2011.

[36] C. McManus, B. Upcroft, and P. Newmann, "Scene signatures: Localised and point-less features for localisation," in *Robotics: Science and Systems*, (Berkeley), Jul. 2014.

[37] H. Friedrich, D. Dederscheck, K. Krajsek, and R. Mester, "View-based robot localization using spherical harmonics: Concept and first experimental results," in *Pattern Recognition*, vol. 4713 of *Lect. Notes Comput. Sci.*, (Heidelberg), pp. 21–31, Sep. 2007.

[38] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *Int. J. Robotics Res.*, vol. 21, pp. 735–758, 2002.

[39] P. Newman and K. L. Ho, "SLAM loop closing with visually salient features," in *Proc. IEEE Int. Conf. Robotics Autom.*, (Barcelona), pp. 635–642, Apr. 2005.

[40] J. Collier, S. Se, and V. Kotamraju, "Multi-sensor appearance-based place recognition," in *Proc. Int. Conf. Comput. and Robot Vis.*, (Regina), pp. 128–135, May 2013.

[41] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. IEEE Int. Conf. Comput. Vis.*, (Nice), pp. 1470–1477 vol.2, Oct. 2003.

[42] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Proc. ECCV Workshop Stat. Learn. Comput. Vis.*, (Prague), pp. 1–22, 2004.

[43] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. 20th IEEE Conf. Comput. Vis. Pattern Recognit.*, (New York), pp. 2161–2168, Jun. 2006.

[44] J. Callmer, K. Granström, J. Nieto, and F. Ramos, "Tree of words for visual loop closure detection in urban SLAM," in *Proc. Australasian Conf. Robotics Autom.*, (Canberra), pp. 1–8, 2008.

[45] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based maps," *Int. J. Robotics Res.*, vol. 29, no. 8, pp. 941–957, 2010.

[46] D. Galvez-Lopez and J. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robotics*, vol. 28, pp. 1188–1197, Oct. 2012.

[47] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth, "OpenFABMAP: An open source toolbox for appearance-based loop closure detection," in *Proc. IEEE Int. Conf. Robotics Autom.*, (Saint Paul), pp. 4730–4735, May 2012.

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

[49] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE/ACM Int. Sym. Mixed and Augmented Reality*, (Nara), pp. 1–10, Nov. 2007.

[50] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]. Part I: The First 30 Years and Fundamentals," *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[51] F. Fraundorfer and D. Scaramuzza, "Visual Odometry. Part II: Matching, Robustness, Optimization, and Applications," *IEEE Robotics and Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[52] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," in *arXiv:1607.02565*, July 2016.

[53] J. Gower, "Generalized procrustes analysis," *Psychometrika*, vol. 40, pp. 33–51, March 1975.

[54] A. Beinat and F. Crosilla, "Generalized procrustes analysis for size and shape 3-d object reconstruction," *Optical*, pp. 345–353, 2001.

[55] A. Bartoli, D. Pizarro, and M. Loog, "Stratified generalized procrustes analysis," *International journal of computer vision*, vol. 101, no. 2, pp. 227–253, 2013.

[56] R. Toldo, A. Beinat, and F. Crosilla, "Global registration of multiple point clouds embedding the generalized procrustes analysis into an icp framework," in *3DPVT 2010 Conference*, 2010.

[57] L. Armesto, J. Minguez, and L. Montesano, "A generalization of the metric-based iterative closest point technique for 3d scan matching," in *2010 IEEE International Conference on Robotics and Automation*, pp. 1367–1372, May 2010.

[58] C. Bron and J. Kerbosch, "Algorithm 457: Finding all cliques of an undirected graph," *Commun. ACM*, vol. 16, pp. 575–577, Sept. 1973.

[59] D. Eppstein, M. Löffler, and D. Strash, "Listing all maximal cliques in sparse graphs in near-optimal time," *CoRR*, vol. abs/1006.5440, 2010.

[60] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[61] L. Armesto, J. Minguez, and L. Montesano, "A generalization of the metric-based iterative closest point technique for 3d scan matching," in *2010 IEEE International Conference on Robotics and Automation*, pp. 1367–1372, May 2010.

[62] M. Ruhnke, R. Kümmerle, G. Grisetti, and W. Burgard, "Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2812–2817, May 2011.

[63] A. Gersho and R. M. Gray, *Vector quantization and signal compression*, vol. 159 of *The Springer International Series in Engineering and Computer Science*. Springer, 1992.

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

[65] 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, p. 155, Jul 2008.

[66] D. Gálvez-López and J. D. Tardós, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Transactions on Robotics (TRO)*, vol. 28, pp. 1188–1197, October 2012.

[67] G. Bradski, "Open source computer vision library," *Dr. Dobb's Journal of Software Tools*, 2000.

[68] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[69] G. Tipaldi and K. Arras, "FLIRT - interest regions for 2D range data," in *Proc. IEEE Int. Conf. Robotics Autom.*, (Anchorage), pp. 3616–3622, May 2010.

[70] G. D. Tipaldi, L. Spinello, and W. Burgard, "Geometrical FLIRT phrases for large scale place recognition in 2D range data," in *Proc. IEEE Int. Conf. Robotics Autom.*, (Karlsruhe), pp. 2693–2698, May 2013.

[71] M. Himstedt, J. Frost, S. Hellbach, H. J. Bohme, and E. Maehle, "Large scale place recognition in 2D LIDAR scans using geometrical landmark relations," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, (Chicago), pp. 5030–5035, Sep. 2014.

[72] M. Himstedt and E. Maehle, "Geometry matters: Place recognition in 2D range scans using geometrical surface relations," in *Proc. Eur. Conf. Mobile Robots*, (Lincoln), Sep. 2015.

[73] P. Hansen and B. Browning, "Visual place recognition using HMM sequence matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, (Chicago), pp. 4549–4555, Sep. 2014.

[74] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. 13, pp. 260–269, Apr. 1967.

[75] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, pp. 257–286, Feb. 1989.

[76] R. Shinghal and G. T. Toussaint, "Experiments in text recognition with the modified Viterbi algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 1, pp. 184–193, Apr. 1979.

[77] J. Philbin and A. Zisserman, "Object mining using a matching graph on very large image collections," in *Proc. 6th Indian Conf. Comput. Vis. Graphics Image Process.*, (Bhubanswar), pp. 738–745, Dec. 2008.

[78] P. Turcot and D. Lowe, "Better matching with fewer features: The selection of useful features in large database recognition problems," in *Proc. ICCV Workshop Emergent Issues Large Amount. Vis. Data*, (Kyoto), pp. 2109–2116, Oct. 2009.

[79] J. Deray, J. Sola, and J. Andrade-Cetto, "Word ordering and document adjacency for large loop closure detection in 2d laser maps," *IEEE Robotics and Automation Letters*, vol. PP, no. 99, pp. 1–1, 2017.